UNIVERSIDAD AUTÓNOMA METROPOLITANA
Casa abierta al tiempo Azcapotzalco
DIVISIÓN DE CIENCIAS Y ARTES PARA EL DISEÑO
Especialización, Maestría y Doctorado en Diseño

PROPUESTA DE DISEÑO MODULAR PARA LA CONFIGURACÓN DE UN ENTORNO VIRTUAL DE ESEÑANZA-APRENDIZAJE CON TUTORÍA INTELIGENTE
Prototipo DECANO

Martha Gutiérrez Miranda
Tesis para optar por el grado de Doctor en Diseño
Línea de Investigación: Nuevas Tecnologías

Miembros del Jurado:

Dra. María Aguirre Tamez
Directora de la tesis

Dra. Ana Lilia Laureano Cruces
Dr. Miguel Ángel Herrera Batista
Dr. Iván Garmendia Ramírez
Dra. Marcela Esperanza Buitrón de la Torre

México D.F.
Agosto de 2014

AGRADECIMIENTOS

Quiero agradecerle en primer instancia a la Dra. María Aguirre Tamez, el esfuerzo tan grande que implicó acompañarme en esta aventura. Sé que no fue nada fácil, porque fui una alumna un tanto cuanto complicada, pero yo espero que al final el resultado haya cumplido las espectativas. Gracias por la paciencia y el tiempo dedicado a esta tesis.

De igual forma agradezco a la Dra. Ana Lilia Laureano Cruces, que desde que se planteó el proyecto aceptó leer este documento.

También doy las gracias a la Dra. Marcela Esperanza Buitrón de la Torre, mi compañera y amiga, que se ofreció a apoyarme siempre y al Dr. Miguel Ángel Herrera Batista, a quien admiro por su trabajo, que en el camino ha sido mi asesor permanente y que colaboró en la revisión de este texto.

Asi mismo, mi agradecimiento para el Dr. Iván Garmendia Ramírez quien aceptó revisar este documento y me apoyó en la etapa final tan complicada. La experiencia de todos y cada uno de ellos enriquecieron sustantiva el resultado.

De igual forma, quiero agradecer y hacer un gran reconocimiento al Laboratorio de Informática de UAQ, a todos los colaboradores y equipo de trabajo que en los últimos años facilitaron el desarrollo de la propuesta y me ayudaron a que el DECANO saliera a la luz, especialmente al Ingeniero Eduardo Trejo que desarrolló la primera fase de programación. De igual manera, agradezco a los alumnos y profesores, que con mucho entusiasmo se prestaron para la evaluación del sistema y a quienes todavía siguen trabajando para mejorar y completar la propuesta.

Y para finalizar, agradezco a la Universiad Autónoma Metropolitana, especialmente al Arq. Eduardo Kotasek González, por todas la facilidades para la realización de esta investigación, también agradezco a quienes tuvieron que ser partícipes y se convirtieron en grandes apoyos, especialmente al personal de Alper Consultores, las secretarias y personal adiministrativo del Departatmento de Procesos y Técnicas de Realización de la UAM y a los alumnos de la UAQ que me apoyaron durante el proceso.

Dedicatoria

"Nuestra recompensa se encuentra en el esfuerzo y no en el resultado.
Un esfuerzo total, es una victoria completa"
Mahatma Gandhi

Este documento y todo lo que hay detrás de él, jamás se hubieran podido materializar sin el apoyo incondicional de mi familia, que más de una ocasión tuvo que estar al pie del cañón con palabras de aliento, acompañamiento, porras y todo lo que se hacía necesario para que siguiera adelante. Valoro en mucho sus consejos, tiempo, comprensión y ayuda siempre generosa.

Dedico este esfuerzo a todos ellos, porque han sido mi apoyo más grande e incondicional.

Especialmente quiero reconocer a Valentina y Edgar, porque en el camino les robé muchas horas de compartir en familia por estar detrás de la computadora. Gracias porque fueron mi gran impulso y porque se mantuvieron siempre cerca y siempre amorosos conmigo, para ellos sea este logro. Gracias por estar conmigo, por compartir la vida, por hacerme sentir siempre especial.

Dedico también esta tesis a mis padres, David y Martha, porque de ellos aprendí el valor del trabajo y la recompensa del esfuerzo, porque sin su apoyo no hubiera llegado hasta donde me encuentro.

Quiero igualmente compartir este logro con mis hermanos, familiares y amigos, porque estoy segura de que son partícipes de la alegría que siento.

Además quiero también dedicar este trabajo a quienes se han ido, pero que se que estarían muy orgullosos de ver que por fin pude concluirlo.

Esta última dedicatoria para mí es muy especial... Dedico esta tesis a la memoria del Arquitecto **Francisco Montero López,** que hace ya 11 años se nos adelantó y que por cuestiones de la vida no pudo terminar su doctorado en diseño, así que valga este esfuerzo por los dos...

RESUMEN

Un Entorno Virtual de Enseñanza-Aprendizaje con Tutoría Inteligente es un sistema que busca que la enseñanza se pueda adaptar a las características de cada estudiante y para su desarrollo, debe contar con muchos atributos como la reutilización, modificabilidad y adaptabilidad, elementos que hoy día se persiguen al diseñarlos. La gran mayoria de este tipo sistemas se construyen desde cero, puesto que no es posible reutilizar los elementos o modificar componentes, sin afectar su funcionalidad, lo que supone un gran esfuerzo en costos y tiempo.

Esta tesis presenta el proceso para el diseño y costrucción de un prototipo adaptable y modificable, considerando para ello una arquitectura modular, que toma en cuenta la instrucción indivualizada, así como la integración del ambiente virtual colaborativo. Adicionalmente, plantea el modelo de interfaz gráfica diseñado a apartir de dimensiones sintático-simbólicas, para la traducción de sus componentes y funciones. Se ha desarrollado un modelo sistmático a partir de estos dos aspectos: interfaz y arquitectura, definiendo una metodología para el diseño, construcción y desarrollo, basada en las cualidadas mencionadas, compo principal aportación. También, se presenta el marco de trabajo para la construcción de prototipos similares por usuarios no especializados, de modo que se complemente con el uso y aplicación de programas y herramientas externas con la posibilidad de ser aplicado a diversos dominios.

Esta propuesta pretende mejorar lo que comúnmente se utiliza para el diseño y construcción y además reducir el costo de desarrollo. Independientemente del dominio de instrucción que se trabaje, la metodología tiene como principales virtudes, acondicionar un Sistema de Tutoría Inteligente que se desenvuelve en un Entorno Virtual de Enseñanza-Aprendizaje colaborativo, donde la modificabilidad y adaptabilidad son las principales características, que permiten hacer de la propuesta, una opción flexible y novedosa, tanto en la arquitectura, como en el modelado de la interfaz. Todo esto como resultado no sólo de aplicar estándares y atributos de calidad, sino también de un proceso sistemático para la evaluación y definición de sus componentes. Finalmente, se definen una serie de recomendaciones, a partir tanto del análisis preliminar, como del diseño y evaluación de la célula tipo que ha sido programada con el objetivo de evaluar la propuesta y generar las conclusiones del proyecto.

ABSTRACT

A Teaching and Learning Virtual Environment with Intelligent Tutoring is a system that seeks to teaching can be adapted to the characteristics of each student and their development, must have many attributes such as reusability, changeability and adaptability, items today pursued their design. The vast majority of this type of systems are built from scratch, since it is not possible to reuse items or modify components without affecting its functionality, which means a great effort in cost and time.

This thesis presents the design process for costruction of a customizable and modifiable system, taking into consideration modular architecture that takes into account the individual instruction, and the integration of collaborative virtual environment. Additionally, the model presents the graphic users interface designed in order of sintactic-symboli values, for the translation of its components and functions.

It had been developed a systematic model from these two aspects, interface and architecture, defining a methodology for the design, construction and development, based on quality atributes. The framework for the prototype construction is also presented for the non-specialist users, so it complements the use and application of external programs and with the possibility to be applied to different domains tools.

This proposal aims to improve what is commonly used for design and construction and also reduce the cost of development. Regardless of the domain of instruction work, the methodology's main virtues put Intelligent Tutoring System that operates in a Teaching and Learning Colaborative Virtual Environment for where the modifiability and adaptability are the main features that allow the proposal a flexible and innovative system, both in architecture and in the modeling of the interface. This results not only in applying standards and quality attributes, but also a systematic process for the assessment and definition of its components. Finally, a number of recommendations are defined, both from the preliminary analysis and the design and evaluation of a cell´s system that has been programmed in order to evaluate the proposal and generate the project finding.

ÍNDICE GENERAL

ÍNDICE DE FIGURAS.

Capítulo 1

Capítulo 2

Capítulo 3

Capítulo 4

Capítulo 5

Capítulo 6

Capítulo 7

ANEXOS

INTRODUCCIÓN AL INFORME

Este informe documenta y presenta la investigación que sustenta el diseño, construcción y evaluación de un Sistema de Tutoria Inteligente (STI) desarrollado con una propuesta concreta de tutor virtual para su implementación como recurso de formación y actualización dentro de un Entorno Virtual de Enseñanza-Aprendizaje (EVEATI). Para la presentación final, se trabajó en los aspectos metodológicos de diseño propuestos por Cataldi (2004), así como en desarrollar la arquitectura basada en le trabajo de Salgueiro *y otros* (2005), en identificar modelos del estudiante como lo expresan Costa *y otros* (2005) y Cataldi y colegas (2007) y en la selección de las funciones de tutoría; asimismo, se hizo la selección del protocolo pedagógico y sus recursos igualemente considerando el trabajo de Salgueiro y sus colaboradores (2005a y 2005b) y de Cataldi y colegas (2005 y 2007), todo con el objetivo de que su comportamiento fuera similar a un tutor humano.

El modelo propuesto tomó en cuenta la base conceptual del modelo de componentes planteado por Carbonell (1970) y sus rediseños (Cataldi, 2004; Salgueiro y otros,2004; Salgueiro y otros,2005; Costa y otros2005; Laureano, 1998), así como la integración de un entorno virtual con el fin de incorporar servicios de comunicación sincrónica y asincrónica. Finalmente se diseñó una arquitectura integrada por el Ambiente Individualizado de Aprendizaje (AIA), el Entorno Virtual de Enseñanza-Aprendizaje con Tutoría Inteligente (EVEATI) y la Interfaz Aprendiz/Sistema.

A través de la interacción entre los módulos básicos y todas las modificaciones que implicó la propuesta, este STI es capaz de determinar lo que sabe el estudiante y cómo va en su progreso, por lo que la enseñanza, se puede ajustar a sus necesidades, sin la presencia de un tutor humano. De esta manera se establece la pertinencia de que el modelo final, funcione bajo dos tipos de ambientes de aprendizaje: el individualizado y el colaborativo. Los usuarios cuando lo deseen podrán pasar de un ambiente de aprendizaje a otro.

Para la propuesta definitiva, la arquitectura además de poseer los cuatro módulos genéricos tradicionales, integra elementos y módulos adicionales para mejorar la tutoría, la evaluación y los recursos multimedia para la simulación.

Este documento muestra la arquitectura diseñada que pretende resolver la problemática mencionada de los STIs en relación al máximo aprovechamiento. Se expresan las cualidades más sobresalientes aproximándonos a niveles altos de simulación y considerando la interacción, mejorando inclusive la interacción tutor-estudiante, mediante la utilización de un entorno virtual; lo que dota al sistema construido bajo esta plataforma, de muchas virtudes, entre ellas: la reutilización, modificabilidad, adaptabilidad y usabilidad con el fin de cumplir con los estándares planteados,

como son un menor tiempo-costo de desarrollo, opciones de enseñanza más adaptables a los estudiantes, monitoreo y evaluación activos y permanentes e interacción.

Bajo este esquema se ha decido denominar al prototipo DECANO (Diseño Experimental para la Capacitación y Actualización con Nuevas Tecnologías, Orientado al aprendizaje) puesto que se trata de un prototipo modelo base y con la función clara de priorizar el aprendizaje. Independientemente del dominio de instrucción, el modelo se adapta tanto a los contenidos, como a los recursos para el aprendizaje y las situaciones didácticas que se presenten. Con el fin de establecer una relación entre la propuesta de arquitectura y el modelado de la interfaz, se diseñó una "célula" del sistema, misma que se evaluó tanto por usuarios, como por expertos en programación y diseño de interfaz. El contenido de aprendizaje se tomó de los fundamentos para la formación de competencias básicas para la investigación y la organización de proyectos y prácticas de investigación, y también se incluyó un diagnóstico/sondeo sobre competencias básicas digitales, que exclusivamente sirvió para determinar el nivel de experiencia como usuarios con las TIC.

Aprovechando este contenido como competencia básica para el nivel de alumno de educación superior, se realizaron tanto pruebas diagnósticas como recursos interactivos, plantillas y por supuesto la célula funcional del sistema o más bien la célula tipo del sistema. Una de las tareas más significativas del Prototipo DECANO ha sido la definición específica de los componentes y subcomponentes de cada uno de los módulos del sistema, ya que a pesar de que basa una arquitectura genérica integrada por los cuatro módulos clásicos, se integraron cambios significativos aumentado otros módulos y resaltando la importancia de la interfaz, como uno de los ejes más importantes de la propuesta, además de la presentación de componentes particulares modelados para el propósito general del sistema.

A a lo largo de este documento se podrán observar y revisar tanto los componentes básicos de cada uno de los módulos, como las funciones más sobresalientes que efectuarán, para posteriormente explicar la manera en que se establecerá la comunicación con el EVEATI y sus características generales y finalmente describir la traducción sintáctico-simbólica que determinó el diseño de la interfaz para los componentes visuales de todos los elementos y funcionalidades.

Se trata de un modelo, que además de comportarse como un tutor a través de una interfaz que se apoya en ingeniería de software, también es capaz de ofertar las facilidades y opciones que otorga internet, así como la posibilidades del ambiente colaborativo integrado, para la interacción. El prototipo simula un centro de educación ya que dentro del planteamiento se precisa la parte administrativa y la educativa. Los componentes administrativos fueron fáciles de abordar con los métodos clásicos de ingeniería de software (consulta de material de enseñanza, matrícula

de los estudiantes, altas y bajas de los profesores y los alumnos, gestión de la certificación académica, etc). Los componentes relativos a la formación, sin embargo, resultaron bastante más complejos. Porque el tutor reproduce la toma de decisiones y esto conlleva un enfoque psicopedagógico. Por ello para diseñar y desarrollar estos componentes, se decidió recurrir a la ingeniería del conocimiento y las técnicas de inteligencia artificial. De esta forma el sistema es capaz de reproducir el comportamiento de un tutor real.

El funcionamiento completo se controla por un sistema inteligente basado en reglas de producción; el estudiante puede decidir parte de la secuencia del programa, por lo tanto, el sistema ofrece cierto control sobre su propio proceso de aprendizaje. Estas características aumentan la facilidad de uso, facilitan también su comprensión, permiten y fomentan la autonomía y resultan una opción vanguardista para la formación, considerando las limitantes más importantes de esta actividad: costo, tiempo, conocimiento actualizado, capacitación continua y permanente.

Este trabajo además de centrarse en desarrollar la estructura del sistema general que sirve como base para la creación del Sistema de Tutoría Inteligente, facilita el caminio, a quienes de forma espontánea, accidentada o bien fortuita se encuentran involucrados en este tipo de tareas. Todas estas nuevas formas de interrelación nos sitúan en el camino de la innovación y el progreso. Al final, parte de los objetivos que se buscaban era que este Entorno Virtual colaborativo, por un lado se convietiera en un espacio en el que los estudiantes pudieran actuar como si se encontrasen en un entorno o ambiente real, meta que se cumplió al realizar la evaluación con el grupo piloto; y por otro lado, que tuviera la capacidad de adaptarse y comportara las cualidades de reutilización y modificabilidad que mucho se han perseguido y considerando la experiencia, así como los resultados, también esta meta se ha alcanzado.

DESARROLLO DEL DOCUMENTO

El presente trabajo está integrado por el planteamiento general, siete capítulos y las conclusiones finales. El primero de ellos comprende el planteamiento del problema, donde se expone concretamente lo que se ha de resolver, asi como los objetivos, el método de trabajo y un acercamiento general a los posibles aportes de la tesis.

En el capítulo 2 se introducen las principales áreas de investigación relacionadas con este documento: realidad virtual, sistemas de enseñanza, agentes y arquitecturas software, arquitectura de la información y diseño de interfaz gráfica. También se realiza un estudio de algunos de los proyectos que han llevado a cabo distintos grupos de investigación dentro de estas áreas, prestando especial atención a aquellos proyectos que abarcan más de una de ellas, pues resultan de interés para analizar la forma en que han resuelto los problemas encontrados en su integración.

Con la información obtenida al analizar el estado de la cuestión, en el capítulo 3 se hace el planteamiento sobre el diseño general de la propuesta, haciendo énfasis en las principales funcionalidades. Dentro del capítulo 4 se presentan las principales consideraciones que deben tener presentes al construir el EVEATI, proponiéndose el enfoque a utilizar para su resolución y los requisitos mínimos necesarios para su construcción.

A partir de la experiencia obtenida, en el capítulo 5 se presenta la arquitectura diseñada y se documenta todo el sistema a partir de indicadores y funciones. Con el fin de que otros investigadores o grupos de investigación puedan utilizar la propuesta o bien para que se mejore el planteamiento, se exponen unas recomendaciones metodológicas que pueden ser utilizadas como guía para su construcción. Además, se proporcionan criterios de diseño cuyo objetivo es servir de ayuda en la creación de sistemas similares al tratado en el presente trabajo.

En el capítulo 6 se plantea la propuesta del modelo de interfaz gráfica de usuario, a fin de obtener una visión más completa y todas las consideraciones que fueron tomadas en cuenta. Adicionalemnte se presentan una serie de recomendaciones metodólogicas y criterios para su diseño. Esto se traduce en una lista de puntos sensibles, junto con los factores que pueden afectarles y las implicaciones que pueden tener para la interacción y la simulación general.

El capítulo 7 describe la evaluación realizada al modelo funcional de la propuesta y se hace un breve recorrido por lo que ha supuesto desarrollar el trabajo de tesis y presenta las principales aportaciones que se han producido con su elaboración. Y para terminar, se presentan tanto las conclusiones como las que, a criterio de la autora, constituyen las principales líneas de trabajo futuro que pueden seguirse desde el punto en el que se ha finalizado el trabajo.

PRINCIPALES APORTACIONES

Considerando el proceso completo de investigación y aplicación del pruebas a la “célula” prototipo que se diseñó y construyó a partir de la metodología propuesta, las principales aportaciones se realizaron sobre la arquitectura software diseñada y el modelo sintético para el diseño de la interfaz.

Con este trabajo se enlistan también una serie de recomendaciones metodológicas, obtenidas a partir del diseño de la arquitectura, y que constituyen las bases que han permitido llegar hasta la solución descrita. Estas consideraciones deben entenderse más bien como una serie de criterios de decisión que ayuden al diseño del tipo de sistemas objeto de estudio, allí donde otras metodologías no cubren estos aspectos.

Otro punto a resaltar es que desde el inicio se hizo mucho hincapié en la documentación. Resulta de especial relevancia elaborar una documentación adecuada por cada uno de los posibles involucrados, lo cual implica reflejar los aspectos que necesitan conocer cada uno de ellos con un nivel de detalle suficiente. La recomendación es no hacerla demasiado profunda para no dificultar la comprensión del documento, este punto permitió regresarnos cuando algo salió mal y proseguir sin tener que partir de cero, además nos abrió caminos y posibilidades y aunque a primera vista esto parciera constituir un elemento de retraso, por el contrario al concluir todo este proyecto, esa iniciativa fue de gran utilidad y permitió se pudieran reestructurar muchos de los puntos importantes de el presente documento.

Y aunque todo lo anterior parece motivo suficiente para leer esta tesis, o al menos para despertar el interés, considero sin lugar a dudas que la mayor aportación y el gran reto personal lo tenemos en dos proyectos de investigación primero el DECANO, por obvias razones que a medida que se vaya leyendo este documento se irán develando y la herramienta de autor TAO denominado “TAxonomía para la redacción y definición de Objetivos”, proyectos que se han convertido en dos iniciativas que se consolidarán con el desarrollo completo del sistema y todas sus herramientas que se están proponiendo adicionalmente. Además también y como iniciativa del equipo de ingenieros de la Universidad Autónoma de Querétaro que han trabajado en ambos, se decidió también incluir dos herramientas adicionales, lo que significó elaborar dos interfases basadas en un sistema icónico de metáforas que se han documentado en el capítulo 6 y que también aportan al modelo integral de la interfaz de todo el sistema.

Por si esto no fuera suficiente, además de la interfaz gráfica y las interfaces de traducción del sistema, uno de los grandes esfuerzos y lo que implicó y en su momento significó una gran inversión de tiempo, fue el modelado 3D y la animación tridimensional, para que el tutor

realmente tanto a nivel físico, símbólico, operativo y reactivo, resultara lo más parecido a lo que había imaginado. Como ya se ha mencionado y como parte de la lista de requerimientos, se documentaron desde los dibujos iniciales, hasta el último renderizado, proceso que también se precisa en las recomendaciones del modelo de interfaz con el fin de que quienes se acerquen a este trabajo puedan entender y quizá retomar el método de trabajo.

Y de manera un poco más técnica, podemos enlistar a continuación todas las respuestas que en nuestro proceso hoy consideramos se han convertido en aportaciones significativas para el sistema propuesto.

El resultado de la investigación expuesta es una arquitectura software para EVEATIs. En esta arquitectura se han definido los elementos que deben formar parte del EV, así como los que deben conformar el STI, que es uno de los elementos integrantes del sistema de aprendizaje.

Como primera aportación relevante del diseño de la arquitectura se encuentra el hecho de que incluye elementos que no contemplan otras arquitecturas de manera global, como son el planificador, el planificador de rutas, el modelo semántico del mundo virtual y el simulador, además de los componentes básicos de un STI, como son el módulo de tutoría, el experto y el de modelado de los estudiantes.

Además, se ha preparado la arquitectura para que admita la inclusión, de manera sencilla, de otros elementos de importancia, como un agente tutor virtual que controle la presencia física del tutor en el Entorno Virtual.

Otra aportación importante de esta tesis se encuentra en la aplicación crítica de métodos de diseño y evaluación de arquitecturas software a la elaboración de una arquitectura software basada en agentes.

La arquitectura desarrollada no es una arquitectura genérica que pueda ser utilizada para cualquier tipo de Entorno Virtual. Es válida para Entornos Virtuales de aprendizaje conceptual y procedimental, independientemente de su naturaleza, siempre que admitan la definición de las actividades como una serie de acciones que deben realizarse de manera secuencial.

Una cualidad sobresaliente es que permite enseñarle a varios estudiantes de forma simultánea realizando tareas complementarias en una misma actividad.

Así pues, tras la realización de este trabajo, se dispone de argumentos sólidos para afirmar que se ha desarrollado una arquitectura software basada en agentes que además, y no como consecuencia de ello, es modificable en aspectos determinados porque se ha diseñado para que así sea.

También resulta relevante que al final de los capítulos 5 y 6 se expongan una serie de recomendaciones, basadas en la evaluación y la aplicación de indicadores de calidad, que proporcionan una ayuda para comprender los fundamentos que han guiado el diseño de la arquitectura, por un lado, y el modelado de la interfaz por otro.

Todas estos puntos detallados, facilitan la adopción de la arquitectura por parte de otros grupos, lo cual puede ser una posibilidad para enriquecer o difundir esta propuesta. Esto se ha realizado a través de la elaboración de una lista de comprobación que sugiere los elementos de la arquitectura a utilizar en función de los requisitos de la aplicación a construir y la interpretación sintáctico-simbólica para la integración de las interfaces.

CAPÍTULO 1:

PLANTEAMIENTO DEL PROBLEMA

CAPÍTULO 1: Planteamiento del Problema

El objetivo del presente capítulo consiste en plantear la situación problemática de los Entornos Virtuales de Enseñanza Aprendizaje con Tutoría Inteligente. El problema a resolver es el cómo incorporar en un sistema de este tipo todas las cualidades que requieren, es decir una enseñanza activa que permita un aprendizaje dinámico, dentro de un entorno virtual donde se generen altos niveles de simulación para que el estudiante sienta que se encuentra en un entorno real, presentar al estudiante el contenido de acuerdo a su estilo de aprendizaje, asesorarlo acerca de cómo debería aprender un contenido determinado y cuáles son las habilidades esperadas, ejercer la tutoría de forma inteligente, a fin de que pueda cumplir los objetivos del tema en tiempo y forma, asistir en los procesos de trabajo colaborativo con el tutor y con los pares y efectuar diagnósticos sobre el rendimiento académico de los estudiantes y proveerles de herramientas para mejorar su producción.

En este sentido, la propuesta, debe ser lo suficientemente flexible para permitir que cada estudiante, de acuerdo a su nivel inicial y a su estilo de aprendizaje pudiera elegir “su propio” método de enseñanza. El planteamiento consiste entonces en el diseño de un modelo de arquitectura modular y la interfaz para un Entorno Virtual de Enseñanza-Aprendizaje con Tutoría Inteligente (EVEATI) apoyado en un agente pedagógico, con el fin de integrar un sistema de enseñanza capaz de adaptarse a todos estos requisitos.

Aquí se describe tanto la situación problemática que envuelve al problema de diseño, así como presenta la hipótesis general y finalmente plantea los objetivos, el tipo de investigación a desarrollar y el alcance del trabajo de la presente tesis. Como la propuesta pretende establecer un marco de trabajo, el proyecto se ha dividido en etapas, con el objetivo de que sea más fácil de comprender el proceso llevado a cabo y las aportaciones de cada una de ellas.

1.1 Situación problemática

Las tecnologías de la información y comunicación (TIC´s) han encontrado como principales aliadas a las ciencias de la computación y la informática. El campo de la educación es uno de los focos principales de atención de estas disciplinas, puesto que se han dado a la tarea de contribuir en el proceso de aprendizaje de los estudiantes, sin embargo uno de los mayores problemas en los sistemas tradicionales de aprendizaje asistidos por computadora es la dificultad de brindar enseñanza adaptada a las necesidades y características específicas de cada estudiante.

Este problema hizo que otras áreas también aportaran elementos para mejorar esa condición. Así, en el área de la inteligencia artificial se han planteado y diseñado los Sistemas de Tutoría

Inteligente (STI) los cuales se crearon con la idea de poder impartir el conocimiento, utilizando técnicas de inteligencia con el fin de percibir y adaptarse a las distintas necesidades del usuario y de esta forma poder asistir y guiar de forma más eficaz al estudiante en su proceso de aprendizaje. Un STI emula el comportamiento de un tutor humano, es una simulación de un tutor que puede adaptarse al comportamiento del estudiante, identificando la forma en que él mismo resuelve un problema a fin de poder brindarle ayuda cuando lo requiera (Cataldi, Zulma; Lage, Fernando J. ,2009; Laureano-Cruces, Terán-Gilmore, de Arriaga, El Alami, 2003; Laureano-Cruces & de Arriaga, 1998; Laureano-Cruces & de Arriaga, 2000).

Originalmente los STI fueron desarrollados con mayor abstracción a nivel de interacción y adaptación a las diferentes necesidades de los usuarios, lo que a medida resultó un gran acierto a nivel impacto en los objetivos de aprendizaje. A partir de estos resultados, se pudo constatar que estos sistemas podían comportarse no sólo como un tutor con alto nivel de inteligencia, sino que también eran capaces de comportarse como un miembro más de un grupo de estudiantes.

A medida que estos sistemas han experimentado cambios y se han vuelto más completos, incorporando no sólo funciones de ayuda o tutoría, sino agentes con comportamientos reactivos cada vez más parecidos a las respuestas de un tutor humano, se ha observado una creciente atención por estos sistemas en el sentido de mejorar su interfaz e incrementar un mayor número de posibilidades de interacción con los estudiantes.

Los roles que pueden desempeñar van desde el actuar igual como un tutor, o como un estudiante que enseña a otros estudiantes, igualmente pueden fungir como un colaborador, competidor, motivador, o crítico (Choua y otros, 2002). Hasta este punto, digamos que parcialmente cumplen con el cometido, el problema que actualmente se ha observado y que es el principal detonante de esta tesis, es que son construidos con una interfaz deficiente y aunque parece posible concluir que la computadora puede ser inteligente y que comprende todo lo que se le dice, la propuesta gráfica de la interfaz de los STI no se vuelve un elemento mediador como resulta con otros desarrollos, que ya la consideran prioritaria para establecer la relación de interacción con el usuario.

Los desarrolladores de este tipo de sistemas no han tomado en cuenta que la interfaz puede ser considerada como un entorno de simulación en el sentido de que es el lugar donde tienen representación las salidas y entradas. Su compromiso básico es la comunicación entre el sistema y el estudiante y que aunque sean el medio de salida de las acciones del STI, también tiene una responsabilidad didáctica (Laureano, Terán-Gilmore & Rodríguez, 2005; Laureano, y otros, 2009).Tomando en cuenta que ya existen estándares y normas que impactan sobre este aspecto y que se cuenta con modelos y recursos avanzados para el desarrollo de la interfaz, no se puede pasar por alto o bien obviar este aspecto.

La tutoría inteligente o asistida por las nuevas tecnologías de la información tiene como objetivo central constituir un apoyo fundamental para las actividades de aprendizaje en general y para objetivos relacionados con la tutoría en particular. Estas tecnologías se adaptan al proceder del estudiante, identificando la forma en que cada uno resuelve un problema y si es necesario, le prestan ayuda cuando comete errores.

Un tutor inteligente, como ya se ha referido, es un sistema software que utiliza técnicas de inteligencia artificial (IA) para representar el proceso de enseñanza-aprendizaje e interactúa con los estudiantes con el fin de enseñar un determinado dominio (VanLehn, K. 1988). Esta intención quedó plasmada en algunos de los más importantes sistemas tutores inteligentes como fueron: Scholar (Carbonell, 1970), Guidon (Clancey, 1987), Sophie (Brown y otros, 1975), West ((Burton, R. y Brown, J., 1981), así como en la línea de desarrollo de los agentes pedagógicos animados que nacen de los sistemas basados en conocimientos y los sistemas de interfaces inteligentes, como los mencionados anteriormente.

Estos sistemas han buscado que los estudiantes aprendan, practiquen o se entrenen en determinadas habilidades, siempre considerando el interactuar con ellos por medio de un diálogo o una intervención, simulando a un tutor o profesor, o bien en ciertas situaciones actuando como si fueran uno de sus compañeros (Johnson, Rickel y Lester, 2000). A partir de la década de los 90, se ha observado que los STI después de la concepción trimodular de Carbonell (módulos de estudiante, tutor y dominio) deben presentar adicionalmente a cada una de las funcionalidades de los módulos bien definidas propiedades para ser reutilizados desde una visión multidisciplinaria y multingüística, lo que a su vez facilitaría la implementación de sus arquitecturas para otros dominios del conocimiento.

Otro punto débil detectado en este modelo tan difundido es que muchas de sus funciones están relacionadas más con el módulo tutor y el módulo del estudiante, restando importancia a los demás módulos. Este problema requiere una definición más clara de las interfaces para diferenciar la implementación de los módulos. Por ello, se debe identificar al módulo encargado de realizar cada una de las funciones del STI a fin de que las mismas queden definidas sin solapamientos. De esta forma se obtendrán módulos completamente independientes del dominio de la aplicación que podrán intercambiarse (por ejemplo para un dominio diferente) sin necesidad de modificar el resto de la estructura del sistema tutor inteligente.

Mchos de los sistemas que se construyen carecen de una arquitectura reutilizable y no cuentan con una interfaz adecuada para desarrollar el aprendizaje colaborativo, es por este motivo que se propone en esta investigación tanto una arquitectura modular, como el desarrollo de un modelo de interfaz para un Sistema de Tutoría Inteligente de propósito general y colaborativo. Con este

proyecto se busca integrar un sistema que pueda reutilizar componentes, si tiene que crecer o migrar, acompañado de una serie de consideraciones y estándares para modelar su interfaz, priorizando la colaboración y la interacción como parte esencial de sus atributos.

Si nos remontamos a los inicios de estos sistemas, quizá se pueden apreciar las carencias de interfaz por la tecnología imperante con que estos primeros tutores fueron construidos, sin embargo en este momento en el que lo visual resulta un elemento de alto impacto, este aspecto no puede descuidarse.

Además de esta situacion, no se diseña la arquitectura, porque ya se dispone de la tradicional estadarizada de los Sistemas Inteligentes de Tutoría. No se diseñan sistemas reutilizables, porque la teoría de los Sistemas Inteligentes de Tutoría, por un lado, y de los agentes, por otro, afirman que los sistemas que se basen en ellas serán reutilizables.Y se puede decir que también, uno de los problemas a los que comúnmente deben hacer frente los equipos que desarrollan STIs que funcionan junto a un EV es que ningún miembro del equipo suele ser experto. Los Evs desarrollados suelen padecer de defectos como escenarios mal estructurados, interfaces sin correspondencia a la arquitectura o bien, desarticuladas, algoritmos poco eficientes o modelos 3D de aspecto bastante pobre. Todo ello hace que tanto el resultado final de la aplicación, como la velocidad en la ejecución hagan desmerecer la labor realizada por estos equipos.

Si reconsideramos todos estos inconvenientes, nos encontramos frente a un campo con muchas aportaciones importantes, pero que ha descuidado aspectos fundamentales y que pueden mejorar la construcción y aparienca considerablemente.

En primer lugar, la falta de componentes que puedan ser reutilizados para construir estos sistemas hace que los miembros de todas las comunidades se vean obligados a desarrollar componentes que están fuera de su campo de investigación. El resultado, por ejemplo, genera aplicaciones con excelentes capacidades de modelado del estudiante, pero con una apariencia bastante pobre y una ejecución lenta, o sistemas muy vistosos altamente cuidados en cuanto a la presentación de la interfaz gráfica, pero cuyas posibilidades son bastante limitadas.

En los casos en los que sí es posible encontrar algún elemento que se pueda reutilizar, por ejemplo, el dominio o contenido,lo habitual es experimentar serios problemas a la hora de integrarlos con la parte de desarrollo propio, si no una imposibilidad total, lo cual conduce también a fracasos o a sistemas con carencias notables.

Finalmente, en muchas ocasiones, la falta de formación y de experiencia en el desarrollo de aplicaciones informáticas y la poca o nula preocupación por desarrollar sistemas o entornos que

sean mucho más simbólicos y accesibles a mayor cantidad de usuarios, hace que gran parte de estos sistemas presente problemas de diseño, implementación y mantenimiento, por no hablar de su reutilización.

Se puede decir, por tanto, que sería “ideal” que los miembros de las distintas comunidades no tuviesen que desarrollar ellos mismos toda la infraestructura que necesitan para implementar los frutos de sus investigaciones. En su lugar, resultaría más sencillo para ellos y más fructífero para sus investigaciones que pudiesen utilizar componentes ya desarrollados que pudiesen interconectarse fácilmente entre sí, adicionalmente que partieran de metodologías de desarrollo integradas a través de principios simples de arquitectura y diseño de interfaz, lo cual haría más corto el camino hacia el resultado.

El primer punto al que hay que prestar especial atención para conseguir este objetivo se centra en el diseño de la arquitectura software de estos sistemas. Este es un tema al que el mundo del software no le está prestando la atención que requiere, y el área de los Entornos Virtuales no constituye una excepción. Y en segundo término y que generalmente queda sumamente relegado, centrarnos en presentar una propuesta sistemática que posibilite el diseño y desarrollo de la interfaz en concordancia con las necesidades de los sistemas o entornos que se estén desarrollando.

1.2 **Problema de Diseño**

El principal problema de diseño radica en plantaer un sistema lo más completo posible, por lo que en primer lugar, se deberá definir una metodología de desarrollo, correspondiente al nivel del conocimiento, en la que se identifican los elementos y actores que intervienen en el proceso de desarrollo. En segundo lugar se construirá un marco de trabajo, que se corresponde con el nivel simbólico, que se ajusta a la metodología definida y permite el diseño y desarrollo del sistema, por usuarios no especializados mediante la integración de programas preexistentes o bien con el desarrollo de herramientas de autor.

Tomando como referencia estos aspectos, si se desarrolla un Entorno Virtual de Enseñanza-Aprendizaje dotándolo de un Sistema de Tutoría Inteligente que tome en cuenta los estilos de aprendizaje de los alumnos logrando hacer que sea modificable y reutilizable su Arquitectura e Interfaz, con el fin de representar diferentes estilos dirigidos al aprendiz, el proceso de aprendizaje será más dinámico, flexible y adaptable a las necesidades de instrucción.

Adicionalmente se prentede la incorporación de un tutor reactivo con ciertas cualidades y características que permitan mejorar la respuesta y motivación de los estudiantes. Por ello, se considera que si la personificación y presentación del agente pedagógico logra ser empática y resulta útil en la tutoría o apoyo, influirá en el desempeño del estudiante y mejorará la usabilidad del sistema y su motivación hacia el dominio de aprendizaje.

Finalmente se planteará una metodología de diseño basada en estándares y normas, para la construcción de la interfaz. De tal manera que se debe buscar que la interfaz se adecúe de forma correcta a las necesidades del sistema y logre traducir cada función de manera simbólica y natural, para que el estudiante logre familiarizarse y apropiarse rápidamente del entorno virtual, mejorando su desempeño y lográndose el aprendizaje esperado.

1.3 **Supuestos de diseño**

El STI basado en un modelo de arquitectura modular flexible, adaptable y modificable diseñado con una interfaz gráfica que logre interpretar de forma sintáctico-simbólica los componentes y sus funciones, permitirá el desarrollo de la dialéctica enseñanza-aprendizaje.

Adicionalmente, con el diseño y construcción del sistema, considerando los estilos de aprendizaje, se fomentará la independencia y autonomía del alumno quien será capaz de decidir el orden de acceso a cualquier información de la base de conocimiento sin restricción de la estructura inicial impuesta por el diseñador de la base de conocimiento o el tutor, lo que nos coloca en un modelo constructivista de formación.

Al establecer los fundamentos para diseñar una interfaz consistente, usable y de fácil manejo se permitirá que el usuario mejore la interacción, puesto que se ofrece una interfaz más adecuada a sus características, habilidades o preferencias, considerando como prioridad un alto grado de usabilidad en cada una de las posibles plataformas donde puede hacer uso de la aplicación, y en cada uno de los entornos donde dicha interacción se pueda llevar a cabo.

Esta intefaz, soportada por un sistema inteligente que se fundamenta en la gestión de ideas y que de manera atractiva pretende capturar la atención del usuario a partir de su presentación, se complementará con el diseño de un agente tutor, dotado de cualidades tanto físicas, como funcionales, que facilitarán la adaptación del estudiante, al sentir empatía y motivación hacia el dominio de conocimiento, por el acompañamiento y apoyo del agente.

De esta forma el sistema completo, habilitado dentro de un entorno virtual, considerando prioritaria la estructura de la interfaz y la arquitectura, si se usan para la construcción del conocimiento, pueden completar la necesidad de entornos de aprendizaje “a medida” para las crecientes demandas de la sociedad de la información.

Y finalmente el diseñar y desarrollar el modelo metodológico tanto para la arquitectura, así como para la interfaz, a partir de indicadores y estándares, facilitará el crecimiento y construcción de otros sistemas similares con la posibilidad de reutilizar y modificar componentes y funcionalidades

1.4 Preguntas de investigación

En este sentido, se parte de las siguientes preguntas de investigación:

¿ Qué características debe tener el sistema para integrar todas las funcionalidades y cumplir con una tutoría inteligente adecuada para cada estudiante?

¿ Cuáles son los requerimientos necesarios para diseñar tanto la arquitectura, como la interfaz del EVEATI?

¿Cuál sería la metodología adecuada para determinar las funciones básicas y establecer sus requerimientos?

¿De qué manera podrían desarrollarse los componentes gráficos de la interfaz para la interacción entre el agente, los estudiantes y el tutor?

¿Cómo podrían definirse los componentes para que cumplan con los principios de reutilización, modificabilidad, flexibilidad, adaptabilidad y usabilidad?

1.5 Objetivos

1.5.1 Objetivo General

Proponer una una metodología para la elaboración de un STI con una estructura modular, flexible y modificable, donde cada módulo además de desempeñar una función específica dentro de la arquitectura, esté interrelacionado y en sincronía con los demás, pero considerando también la posibilidad de disponerse independientemente, sumando a este desarrollo estándares gráficos,

normas de usabilidad para integrar un diseño de la interfaz que permitan mejorar las condiciones generales de colaboración y de aprendizaje asistidos por un agente pedagógico.

1.5.2 **Objetivos Particulares**

Con el fin de producir una herramienta que facilite su adaptación a un entorno dinámico que permita propiciar el aprendizaje, se propone desarrollar un STI con características reactivas considerando la propuesta que hacen Laureano y De Arriaga (2000) y Laureano,Terán-Gilmore & Rodríguez (2005) y que incluye el componente interactivo de comunicación integradora asociada a las nuevas tecnologías dentro de su interfaz. Y se parte para ello de los siguientes objetivos específicos:

Diseñar una célula tipo representativa del sistema general con el fin de modelar tanto la ingeniería del sistema a partir de la arquitectura planteada, como el diseño de la interfaz, para evaluar si cumple con los atributos propuestos.

Diseñar, programar e implementar un tutor inteligente con características reactivas, parra adaptar a un grado considerable de tópicos y lo suficientemente robusto para poderlo adecuar a futuras necesidades, apegándonos a estándares para su diseño y construcción

Desarrollar tanto el modelo de arquitectura de construcción del EVEATI, como el modelo de interfaz a partir de atributos como la modificabilidad, adaptabilidad y flexibilidad.

Proponer el diseño de los componentes gráficos de interfaz del sistema a partir de los fundamentos del modelo de interacción natural (IN) y considerando para su presentación los criterios de interfaz multimodal con el uso de agentes animados, la experiencia de usuario y la traducción simbólica de funciones.

Evaular las metodología de diseño y construcción del sistema, con el fin de establecer los lineamientos y recomendaciones mínimas para la elaboración de otros entorno o bien para el desarrollo de otros dominios de aprendizaje.

Documentar todos los procesos llevados a cabo para la propuesta, así como el desarrollo y evaluación de la célula básica del prototipo, con el fin de establecer dos modelos metodológicos de recomendaciones tanto para el diseño de la arquitectura, como para el diseño de interfaz.

1.6 **Método de trabajo**

Esta investigación se llevó a cabo por etapas, considerando que debían sincronizarse los campos de la ingeniería software, la inteligencia artificial, el diseño de interfaz gráfica y la ingeniería de requerimientos para la documentación general. Por estas razones, a continuación se describen cada una de esas etapas del desarrollo y al final se presenta un diagrama de como se sincroniza el proceso completo (Figura 1.2).

En una primera etapa se realizó fundamentalmente investigación de carácter documental sobre las diferentes áreas del conocimientos relacionadas con los entornos virtuales de aprendizaje, educación a distancia, ingeniería software e inteligencia artificial e interfaz gráfica de usuario, con el fin de establecer los ejes temáticos más sobresalientes. Todos estos conceptos y elementos, integraron el estado del Arte y sirvieron como marco teórico-conceptual que fundamenta la propuesta, pero adicionalmente también esta investigación se incorporó en la contextualizción inicial de cada uno de los capítulos. Con el fin de que esta etapa estuviera más completa, muchos de los conceptos claver se revisaron a partir del método de cocitación de autor.

Posteriormente y como segunda etapa, se realizó un análisis sobre las diferentes teorías del aprendizaje, los modelos de instrucción y los modelos de tutoría virtual, para en consecuencia, definir las herramientas y recursos tecnológicos que serían empleados en el sistema propuesto, y adicionalmente se realizó un proceso semejante que inició con la revisión de casos y análisis de métodos para la propuesta de interfaz, que se verá reflejada en el planteamiento del modelado de la misma. Para este punto y conjuntamente con el equipo de desarrollo, se tomarán decisiones relacionadas con la Ingeniería de requerimientos y los modelos de ingeniería y construcción que fueron propuestos como metodología para la Arquitectura del Sistema.

Una vez definidos los parámetros básicos de estructura y contenido, así como los elementos sintéticos para el modelado de la interfaz de usuario y el modelado del tutor y habiéndose establecido los lineamientos para el soporte tecnológico de la aplicación, en la tercera etapa se plantearon los criterios para la definición de los contenidos de los módulos del sistema, así como los mecanismos para su evaluación y la gestión del agente tutor y los agentes de apoyo.

La cuarta etapa implicó el desarrollo de una «célula» modelo, como ejemplo general del sistema y para poder llevar a cabo la fase de evaluación, que fue la quinta etapa del proceso. Como parte de las aportaciones y también dentro de esta etapa, la autora propone un sistema de diseño para los íconos y símbolos de la interfaz, documentado ampliamente en el capítulo 6. Este sistema

de diseño permite identificar una serie de consideraciones preliminares, basadas no sólo en estándares para el la interfaz gráfica, sino también para la representación sintáctico-simbólica y la traducción funcional del sistema software a elementos visuales que pueden ser más fácilmente comprendidos por el usuario.

Para mostrar la validez de este método se realizó la evaluación de la célula prototipo tanto con usuarios, como con expertos en el área de interfaz a manera de evaluación heurística y una evaluación basada en atributos de calidad, realizada por los expertos en ingeniería. Este proceso implicó la documentación del proceso y la aplicación de pruebas, con el fin de definir los aspectos del sistema de forma completa y valorar su implementación.

Y finalmente, en la etapa seis, se plantearon los criterios para el análisis y gestión de modelos de aplicación con el apoyo de la tecnología desarrollada considerando los factores pedagógicos, instruccionales, tecnológicos y de interfaz que puedan ser utilizados en posteriores proyectos de la misma naturaleza.

La propuesta por tanto incluye una serie de estrategias de instrucción, consideraciones de estilos de aprendizaje y gestión de los modelos del alumno y de los dominios implementados que deben ser vistos como un ejemplo. Cualquiera de estos módulos podría ser sustituido por otro que se ajuste a las especificaciones de la metodología. Este enfoque permite la construcción de distintos marcos de trabajo basados en la misma metodología de desarrollo. La figura 1.1 refiere todo el proceso.

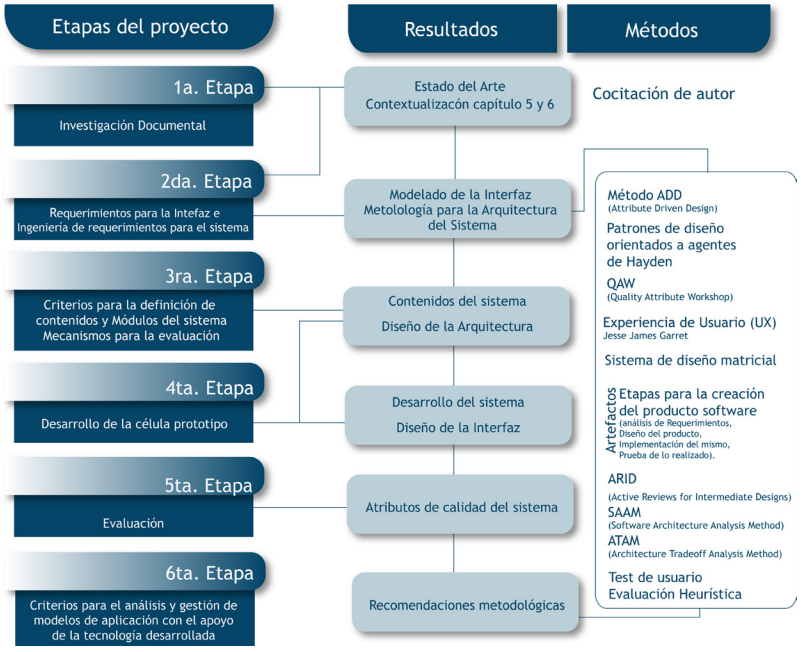


Figura 1.1: Etapas del Proceso completo.

Todo el proceso llevado a cabo concluye con una serie de consideraciones y puntos sensibles que posibilitan su valoración como metodología integral de trabajo en el desarrollo de propuestas similares. Y principalmente todo este proceso lo que permite es valorar la importancia no sólo intedisciplinaria del proyecto, sino también de los múltiples métodos y recursos, ahora necesarios para integrar un sistema lo más completo posible.

CAPÍTULO 2:

ESTADO DEL ARTE

CAPÍTULO II. Estado del Arte.

Los primeros pasos de la enseñanza asistida por computadora se dieron en los años 50, y su finalidad era hacer uso de la informática para instruir a los alumnos de una forma personalizada. Los primeros sistemas se basaron en la "instrucción programada" y originaron la denominada *Computer Assisted Instruction* (CAI) o *Computer-Based Training* (CBT) que aún existe hoy. La clave para su funcionamiento es la división del material docente en pequeñas partes, de tal forma que cada una de ellas requiera la intervención del alumno. En función de la respuesta del estudiante, el sistema continuará con una u otra parte, de modo que la computadora es capaz de adaptar el curso a las necesidades del alumno, intuidas a partir de sus respuestas.

Adicionalmente, la enseñanza a distancia nació como una solución para la instrucción en zonas donde la población estaba muy dispersa. Desde el principio demostró tener problemas particulares, debido a la poca comunicación entre los profesores y los alumnos. Esto originó la creación de material específico para este tipo de cursos, de modo que se comenzaron a incuir test, cuestiones y actividades en un intento de crear una mínima interacción entre el alumno y el material, más allá de la simple lectura. Con la unión de todas estas técnicas, se trataba de conseguir la motivación constante del estudiante, al hacerle sentirse partícipe en el descubrimiento de cada nuevo concepto.

La llegada de Internet y la WWW a los hogares ha brindado la posibilidad de mejorar la interacción entre estudiantes y profesores en este tipo particular de enseñanza. Gracias a Internet, el material (ya sea temario a estudiar o nuevos ejercicios y exámenes para autoevaluación) puede ser accesible electrónicamente, por lo que su actualización y mejora es mucho más sencilla y barata. Además, ese material puede complementar a los libros, añadiendo información multimedia como sonido o vídeos. A parte del incremento en la cantidad de información relacionada con la asignatura, Internet abre las puertas a la llamada *tutoría telemática*, gracias a las nuevas formas de comunicación que pueden utilizarse entre tutor y alumno, como son los chats y los foros de discusión. Estos dos últimos servicios pueden ser utilizados también entre alumnos, de modo que se puede crear entre ellos una colaboración difícil de conseguir anteriormente en la enseñanza a distancia, permitiendo, por ejemplo, realizar trabajos en grupo sin necesidad de salir de casa.

Para que todo esto se convierta en algo habitual, los profesores deben contar con herramientas que faciliten la tarea. Con esta finalidad han aparecido en el mercado una serie de entornos integrados denominados IDLE (Integrated Distributed Learning Environments) que simplifican la creación y gestión de contenido educativo, así como su publicación en Internet para hacerlo

accesible a los alumnos a través de navegadores Web comunes. Además facilitan la creación de foros, la gestión del control de acceso, el soporte de servicios de conferencia electrónica y muchos servicios más.

Sin embargo estos sistemas suelen poner más énfasis en facilitar la gestión que en la creación de material educativo, cuya confección está mucho más ligada a la presentación Web que a la elaboración de conocimiento estructurado que pueda ser reutilizado fácilmente. Eso ha llevado a algunos investigadores a buscar alternativas, a menudo basadas en programas independientes que no utilizan navegadores Web, sino que se ejecutan en la computadora del alumno para proporcionar modos de interacción más elaborados y que utilizan Internet para acceder a información adicional. También hay intentos de facilitar la construcción de contenidos educativos sobre Internet en los que el usuario es ayudado por agentes pedagógicos animados.

Los Entornos Virtuales de Enseñanza Aprendizaje (EVEA) son un tipo de aplicación que requiere combinar tecnologías pertenecientes a distintas áreas de investigación para su desarrollo. Como no dejan de ser una aplicación de la informática a la enseñanza, para completar efectivamente sus funciones han tenido que sumar uno de sus componentes fundamentales, o sea, los Sistemas de Tutoría Inteligente (STI), que es el elemento que aporta el componente pedagógico al sistema. Los STI son programas cuya misión es transmitir a un alumno el conocimiento y habilidades relacionadas con determinada asignatura y que tienen como objetivo principal, realizar una enseñanza personalizada, adaptada a las necesidades del alumno, intentando emular la forma en que lo haría un tutor humano.

El desarrollo de los STI es una tarea compleja que requiere la participación de profesionales de áreas muy diversas como la Inteligencia Artificial, la Ingeniería del Software, la Pedagogía, la Psicología Cognitiva y los campos propios del dominio de enseñanza y adicionalmente, como se verá en este proyecto, la intervención de expertos en diseño de interfaz.

En la actualidad, este campo ha evolucionado hacia una nueva generación de STI, los Entornos Inteligentes de Aprendizaje para la Web (WILE, Web-based Intelligent Learning Environments), sobre los que versa esta tesis. Por ello se ha considerado de interés acercarnos a sus orígenes realizando la revisión de algunos de los STI desarrollados en las últimas décadas, como ejemplo de los principios y técnicas sobre los que se sustentan estos nuevos entornos de aprendizaje inteligentes.

En este tema se introduce el concepto de STI, el estudio de su arquitectura profundizando en cada uno de sus módulos, la descripción y clasificación de algunos de los STI más significativos por sus aportaciones a esta disciplina, así como las primeras aproximaciones a los elementos

referenciales simbólicos que integran las interfases gráficas de usuario y para complementar al finalizar el documento de tesis se incluyen dos anexos, el correspondiente a la descripción detallada de los ejemplos y el relacionado a las normas y estándares para el modelado de la interfaz.

Como segunda pieza se encuentran los Entornos Virtuales (EV), aplicaciones tridimensionales que reproducen el entorno donde tiene lugar el entrenamiento o la formación, y que es uno de los elementos diferenciadores respecto de otras aplicaciones de la informática a la enseñanza.

A la combinación de ambos tipos de aplicaciones es a lo que se le ha dado el nombre de Entornos Virtuales de Enseñanza Aprendizaje (EVEATI) con Tutoría Inteligente, que son Entornos Virtuales en los que un usuario puede practicar actividades que tendrá que llevar a cabo en un entorno real, pero con la particularidad de que contará con la ayuda de un tutor inteligente para guiarle en el proceso de aprendizaje.

Una tercera pieza que puede unirse a este conjunto de componentes son los llamados agentes pedagógicos, que son una representación física del tutor dentro del EVEATI. Esta representación física resulta de utilidad en muchas ocasiones, pues a la capacidad de ofrecer explicaciones y responder preguntas se une la posibilidad de demostrar cómo se realiza una acción y la de guiar al usuario por el entorno y se mejora cuando la presentación física y tridimensional del tutor lo hace un elemento empático y motivante para el estudiante.

A lo largo de este capítulo se presenta una descripción de estas tecnologías, lo que ayudará a tener una visión que delimite el alcance de los sistemas estudiados en la sección de trabajo relacionado. En primer lugar se describe la evolución que han experimentado los Sistemas de Tutoría Inteligente, seguida de una breve historia de los sistemas de Realidad Virtual. Posteriormente se exponen los conceptos del paradigma de la orientación a agentes que se han considerado más relevantes para este trabajo, para dar paso a un análisis del las arquitecturas software, que hasta el momento no han tenido cabida en el desarrollo de Entornos Virtuales y cuya aplicación en esta área constituye una de las principales aportaciones de esta tesis.

Finalmente, se describe la manera en que se ha llevado a cabo la aplicación de la Realidad Virtual a la enseñanza y a los agentes pedagógicos. Una vez que se cuente con una visión de las tecnologías involucradas en el desarrollo de estos sistemas, se presenta una sección de trabajo relacionado en la que se describen los principales sistemas existentes en la literatura relativos al objeto de estudio, prestando especial atención a su desarrollo, arquitectura y metodología de construcción, así como las consideraciones generales relativas al modelado de la interfaz gráfica y la metodología de integración propuesta.

Una vez hechas estas precisiones, también se expone lo relacionado a los mecanismos de interacción hombre-computadora, para definir y resaltar uno de los atributos que deberá integrar el sistema que sin duda es la interacción sumada a los niveles de representación y simulación que buscará alcanzar la interfaz. Por lo que se incluyen también definiciones, dimensiones, elementos y los principios y normas ya reconocidos para su modelado.

Y esta sección se complementa con los anexos, sobre el análisis de los sistemas descritos, identificando así las características comunes a todos ellos que son de interés para la elaboración de esta tesis. Tras este análisis, se realiza una descripción de la experiencia previa en la exploración de sistemas que combinan EVs y tutores inteligentes, pues constituye la motivación que ha dado pie a este trabajo.

Lo que se busca al final de este capítulo es ofrecer una amplia visión de todo aquello que se ha revisado, documentado y que como parte del análisis ha servido de partida para la propuesta de esta tesis.

2.1 Sistemas de Tutoría Inteligente

La historia de los Sistemas de Tutoría Inteligente comienza junto con la de la Inteligencia Artificial entre finales de los años 50 y principios de los 60, cuando investigadores como Turing, Minsky o Newell[1] plantean que las computadoras pueden llegar a pensar de manera similar a la de los humanos.

Este planteamiento encontró su mejor nicho en años posteriores a la Segunda Guerra Mundial, cuando muchos de los soldados que volvieron del frente recibieron ayuda para realizar estudios universitarios. Se requirió la preparación de muchos de ellos para alcanzar el nivel mínimo exigido para poder acceder a la universidad, lo que a su vez obligó a buscar maneras de mejorar la eficiencia en la enseñanza. Para ello, se pensó en el uso de computadoras para satisfacer las necesidades de la educación.

En los años 60 se crearon algunas aplicaciones de enseñanza asistida por computadora (CAI, Computer Assisted Instruction) de tipo generativo (Uhr, 1969). Estos programas creaban conjuntos

1 A partir del famoso artículo de Alan Turing, “Computer Machinery and Intelligence (1950) se establece el test de Turing como forma de determinar el carácter inteligente o no del comportamiento de una máquina. En 1955, Allen Newell, J.C. Shaw y Herbert Simon crearon lo que posiblemente fue le primer lenguaje especializado de la inteligencia artificial llamado IPL-II. Y finalmente Marvin Lee Minsky es considerado como el precursos del análisis de redes neuronales artificiales.

de problemas diseñados para incrementar las habilidades de los estudiantes en determinados dominios.

Básicamente, estas aplicaciones presentaban un problema, recibían la respuesta del estudiante y rellenaban una tabla con los resultados obtenidos. En términos pedagógicos, estos sistemas no establecían ninguna diferencia entre enseñanza y aprendizaje, y asumían que si se le presentaba una información al alumno, éste la aprendería.

Entre finales de los 60 y principios de los 70 los investigadores empezaron a considerar al estudiante como un factor más dentro del sistema (Suppes, 1981), y comenzaron a aparecer aplicaciones que presentaban información distinta al usuario en función de sus respuestas anteriores. Para ello, los programadores tenían que saber por adelantado las posibles respuestas de los estudiantes, para así poder decidir qué hacer en cada caso.

En el año 1970 se publica el que es considerado por muchos investigadores como el primer artículo sobre Sistemas Inteligentes de Tutoría (SIT) (Carbonell, 1970). En él se describe un sistema denominado SCHOLAR (Fig. 2.1), un conjunto de programas basados en el diálogo del estudiante con la computadora, que utiliza una red semántica para representar el conocimiento y para organizar el diálogo con el estudiante, en lugar de utilizar ejercicios, respuestas y baterías de errores predefinidas.

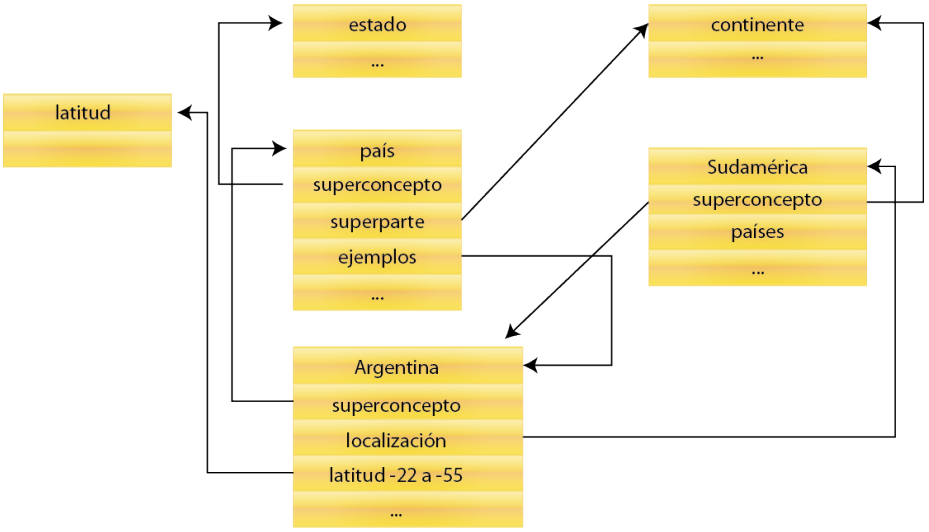


Figura 2.1. Parte de la Red Semántica de SCHOLAR, creada por Carbonell.

En esta época, la influencia de Piaget comienza a hacerse notar, y la teoría del constructivismo se abre camino como base para los métodos de enseñanza/aprendizaje (Piaget, 1954).

A comienzos de los 80, Sleeman y Brown realizan una revisión del estado de la tecnología en el área de los Sistemas Inteligentes de Tutoría (Sleeman y Brown, 1982), en la que se utiliza por primera vez este término para distinguir los sistemas no deterministas de los sistemas anteriores, a los que siguen catalogando como CAI. Se asume ya de manera implícita que el mecanismo de aprendizaje es el de "aprender haciendo" (*learning by doing*). Es también aquí donde por primera vez aparece el concepto de Modelo de Estudiante para describir la representación del alumno dentro del sistema, que clasifican en tres tipos: superposición, en la que el modelo del estudiante es un subconjunto del conocimiento experto; diferencial, que modela el conocimiento del estudiante como subconjunto del experto y resalta las diferencias entre el conocimiento del estudiante y el del experto; y perturbación, que representa los desafíos del estudiante respecto al conocimiento del experto. Estos modelos contrastan con modelos previos, como el propuesto por Brown y Burton (1978), que identifica errores concretos que pueden cometer los alumnos. Este modelo se encuentra implementado en el sistema DEBUGGY (Burton, 1982).

A mediados de los 80 el constructivismo se convertiría en la corriente dominante en la psicología educativa, y una nueva revisión del estado de la tecnología realizado por Wenger muestra lo mucho que han cambiado las cosas en los pocos años transcurridos desde la publicación del trabajo de Sleeman y Brown (Wenger, 1987). En este trabajo, Wenger presenta un modelo de Sistema de Tutoría Inteligente que consta de cuatro módulos: conocimiento pedagógico, conocimiento del dominio, modelo del estudiante e interfaz, que conforman la estructura que hoy en día se conoce como la arquitectura clásica de los STI.

El conocimiento del dominio es un modelo dinámico de la materia a enseñar, que contiene tanto la representación del conocimiento como la manera de razonar sobre él, lo que permite generar distintas soluciones a un problema en lugar de manejar una solución prediseñada.

El modelo del estudiante es la principal característica que permite distinguir los STI de la Enseñanza Asistida por Computadora, ya que permite adaptar la enseñanza a cada estudiante.

Según Wenger, los modelos de estudiante deben recopilar datos acerca de sus acciones, utilizarlos para construir un modelo del conocimiento del estudiante, así como de su forma de aprendizaje, y hacer uso de estos modelos para diagnosticar qué sabe el estudiante y así poder decidir la mejor estrategia pedagógica con la que debe continuarse la enseñanza.

En cuanto al conocimiento pedagógico, debe servir para realizar el diagnóstico pedagógico del alumno y para dirigir el proceso de enseñanza, para lo cual se requiere que observe el comportamiento del alumno y se adapte a él.

Finalmente, la interfaz permite la comunicación entre el alumno y el sistema, que según Wenger debe promover la eliminación de las ambigüedades en las respuestas de los estudiantes, ya que esto facilita el resto de tareas que debe llevar a cabo el STI.

Desde el trabajo publicado por Wenger hasta hoy, estos Sistemas han experimentado notables avances en todas sus áreas, relacionadas tanto con el aprendizaje como con la construcción de los mismos. Sin embargo, para el objetivo de esta tesis, el interés se centra en la utilización de los STIs junto con entornos virtuales para realizar procesos de enseñanza y facilitar la instrucción, motivo por el cual se cierra aquí esta breve introducción a los Sistemas Inteligentes. No obstante, en las siguientes secciones se continúa con el estudio de los STI, aunque en un terreno más cercano al ámbito del presente trabajo, como es el desarrollo de Entornos Virtuales de Enseñanza-Aprendizaje con Tutoría Inteligente.

2.2 Realidad Virtual y Entornos Virtuales

A pesar de que el concepto de Realidad Virtual (RV) parece de creación bastante moderna, lo cierto es que sus orígenes, al menos en el campo de la informática, datan de hace aproximadamente medio siglo.

Existe un consenso relativamente generalizado en otorgar a “Sensorama” (Fig. 2. 2), creado en 1956 por Morton Heilig, el título de primera aplicación de Realidad Virtual, que combinaba una mezcla de videoproyección, audio, vibraciones, viento y olores, y estaba diseñado para hacer sentir al usuario que formaba parte de una película. Patentado en 1961, el sistema permitía al usuario experimentar diversas sensaciones mientras recorría uno de los cinco escenario posibles: un paseo en moto por Nueva York, un paseo en bicicleta, un paseo en un todoterreno por una duna, un vuelo en helicóptero sobre Century City o una danza del vientre. Puesto que los gráficos por computadora estaban por crearse, toda la experiencia se basaba en imágenes cinematográficas pregrabadas.

Figura 2.2: Imagen del Sensorama Patentado por Heilig

Heilig también patentó el que muchos consideran como el primer casco de Realidad Virtual en 1962 (Fig. 2.3), que daba la posibilidad de ver imágenes en tres dimensiones (3D), escuchar sonido en estéreo y percibir olores. Además, propuso la idea de un teatro inmersivo que permitiría la proyección de imágenes en 3D, sonido estéreo y olores propagados por el sistema de ventilación, aunque esta idea no llegó a materializarse.

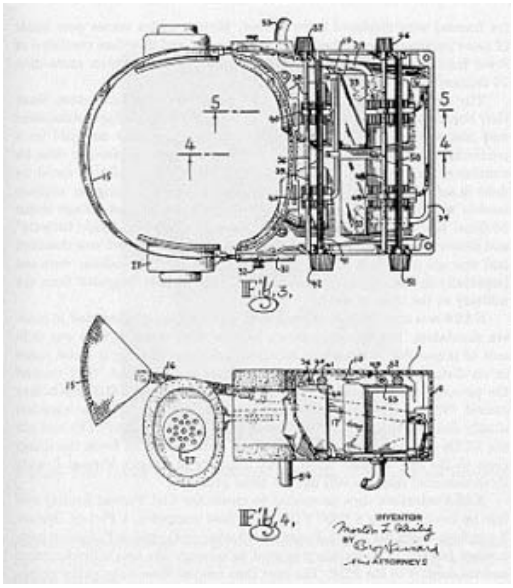


Figura 2.3: Casco de Realidad Virtual de Heilig

En 1961, Corneau y Bryan, empleados de Philco Corporation, construyeron el que parece ser el primer casco de RV de verdad. Este dispositivo permitía ver imágenes en movimiento y estaba dotado de un sensor magnético que determinaba la orientación de la cabeza del usuario. Estaba conectado a un sistema de vídeo controlado a distancia para visualizar situaciones peligrosas.

La utilización de gráficos computarizados tuvo que esperar aún algún tiempo, y se debe al trabajo realizado en el MIT por Roberts y Sutherland. Roberts escribió el primer algoritmo para eliminar superficies oscuras y ocultas de una imagen, abriendo así el camino a la utilización de gráficos 3D. Por su parte, el trabajo de Sutherland consistió en el desarrollo de algoritmos que pudiesen realizar esta tarea de manera eficiente. Uno de los frutos de estos esfuerzos se encuentra en el desarrollo por Henri Gouraud, en el año 1971, de un algoritmo de iluminación que aún es muy utilizado hoy en día. Este algoritmo hace posible que una superficie formada por polígonos cobre el aspecto de una superficie suave y continua.

Mientras tanto, Sutherland también había desarrollado, en el año 1965, el conocido como Ultimate Display (Sutherland, 1965), un casco de RV (Fig. 2.4) dotado de visión estereoscópica y de un sistema mecánico de seguimiento. Este sistema ha sido calificado por la National Academy of Sciences como el elemento fundamental en este nuevo campo de investigación. Durante esos años, la labor de Sutherland fue bastante prolífica, y la red de contactos que creó en el MIT, ARPA y la Universidad de Harvard le permitieron desarrollar numerosos proyectos financiados por entes tan dispares como la CIA, el Departamento de Defensa y los Laboratorios Bell, además de los anteriores.

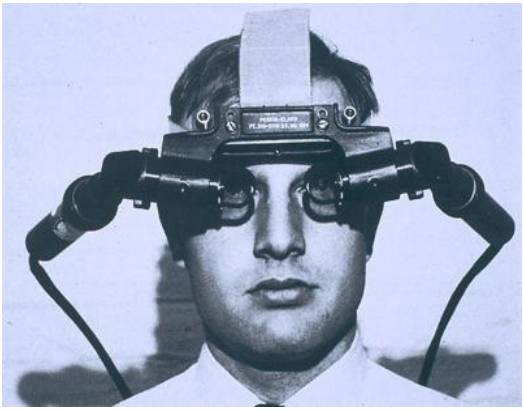


Figura 2.4: The Ultimate Display de Sutherland

De estos años data también la creación de la Realidad Aumentada (RA), una combinación de RV y escenarios reales, también de la mano de Sutherland. El propósito fue un intento de realizar una aplicación práctica aplicada a la medicina, relacionada como la representación del flujo sanguíneo en modelos de las válvulas cardíacas (Sutherland, *y otros,* 1971). En esa época, Thomas Furness comenzó a trabajar en el primer simulador de una cabina de avión para entrenar a pilotos (Furness, 1986). El problema inicial consistía en la creciente complejidad de las cabinas de estos aparatos, por lo que Furness comenzó a buscar la forma de facilitar la interacción con los pilotos. La solución fue el desarrollo de una cabina que proporcionaba información 3D a los pilotos, quienes podían controlar el aparato a través de una representación virtual del terreno con campo de visión de 120º en horizontal. Encendieron este aparato por primera vez en septiembre

de 1981, y ha constituido la base para el desarrollo de los sistemas de entrenamiento militar creados a partir de ese momento.

Los primeros sistemas de RV que funcionaban en estaciones de trabajo no aparecieron hasta los 80,s, en 1981, Michael McGreevy y Jim Humphries crean VIVED (Virtual Visual Environment Display system), para los astronautas de la NASA (Fig. 2.5), una de las primeras estaciones de bajo costo dotadas de un campo de visión amplio, estéreo, con sensores de posición en el casco de RV, enfatizando la transferencia de las imágenes en 3D, que constituyen aún una referencia para los sistemas de hoy en día.

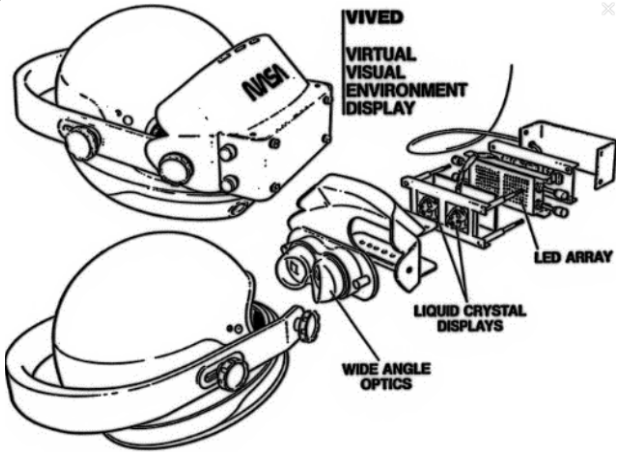


Figura 2.5: Estructura de VIVED

Actualmente, se entiende que un sistema de Realidad Virtual es una aplicación que proporciona una experiencia interactiva e inmersiva en un entorno tridimensional simulado (Gigante, 1993). Como término más general se utiliza Entorno Virtual para definir un entorno tridimensional simulado con el que se puede interactuar por medio de un computadora de escritorio, y es gracias a ellos por lo que estos sistemas han experimentado un crecimiento tan rápido desde mediados de los años 90.

A pesar del desarrollo sufrido, el alto costo de los dispositivos de Realidad Virtual hace que aún hoy no se encuentren al alcance de muchos grupos de investigación, y menos aún del público en general. Además, muchos de ellos dependen todavía de cables conectados a la computadora o de equipos que hay que llevar encima, por lo que su uso prolongado resulta bastante incómodo.

Sin embargo, los que sí han experimentado una expansión notable han sido los Entornos Virtuales, con aplicaciones que se pueden encontrar en áreas como: visualización de información científica (R. van Liere y otros, 2002; Gherbi y Hérisson, 2001; Férey y otros, 2004; Cugini y otros, 2000), CAD (Rodríguez-Toro y otros, 2006), entretenimiento (Hirose, 2006; Zhu y otros,

2008; Ha y Woo, 2006; Sunar y otros., 2006), Medicina (Heng y otros, 2006; Park y otros, 2005) o Planificación urbanística (Drettakis y otros, 2007). Estos constituyen sólo algunos ejemplos de entre la amplia gama de aplicaciones que se han construido en los últimos años. También se han desarrollado un gran número de aplicaciones en el área de la educación, que por su relevancia para este trabajo se explican con mayor detalle en las próximas secciones y se anexan al final de la tesis.

2.3 Agentes

Según la metodología constructivista, el papel del profesor es ayudar al estudiante durante el proceso de construir su propia perspectiva interna del mundo exterior. Para eso, la interacción entre ambos, profesor y alumno, es de vital importancia. No sólo es necesario que el profesor exponga la teoría, si no que es necesaria una cierta comunicación entre ambos para que el alumno pueda recibir una realimentación sobre las acciones que realiza. Así, partiendo del tema que nos ocupa, esta situación debe ser nuestra pretensión del ideal para el sistema propuesto, un entorno virtual dónde el estudiante tenga plena comunicación con el tutor virtual recibiendo respuestas cuando lo necesite.

El término agente es una acepción, que de forma inadvertida se ha ido adoptando en distintas áreas para denominar elementos con propiedades diferentes y que, por el momento, no ha logrado poner de acuerdo a los investigadores en cuanto a su significado. De hecho, probablemente el punto donde más coinciden unos y otros es en destacar que no existe una definición comúnmente aceptada del significado de este término.

Un agente pedagógico virtual inteligente (APVI) es una personificación para la figura del tutor (que instruye, guía y define estrategias pedagógicas a aplicar) en un entorno virtual de aprendizaje. Las bases en que se apoya su desarrollo están en los Sistemas de Tutoría Inteligente (STI), los Entornos Virtuales (EV), la pedagogía y la psicología. Estos entornos son utilizados por alumnos para formarse en una materia particular, y el objetivo de los agentes pedagógicos es potenciar ese aprendizaje. Para ello, adaptarán su comportamiento según las necesidades del estudiante y el estado actual del entorno, proporcionando una realimentación continua a sus acciones. Sistemas más sofisticados pueden admitir varios estudiantes simultáneamente en el mismo entorno de aprendizaje, e incluso un número variable de agentes pedagógicos, creando así un escenario de aprendizaje colaborativo.

Seguramente la diferencia más significativa entre los STI y los entornos de aprendizaje que disponen de un agente pedagógico es la práctica habitual de *encerrar* a estos últimos en una

representación visual para construir lo que a menudo se conoce como agente pedagógico animado. Esto hace que el alumno “vea” al agente que le está enseñando a través de una figura en movimiento que crea la ilusión de tener vida (*life-like*)[2], lo que a menudo tiene repercusiones positivas en la motivación.

El concepto de “agente» caracteriza a una entidad software con una arquitectura robusta y adaptable que puede funcionar en distintos entornos o plataformas computacionales y es capaz de realizar de forma “inteligente» y autónoma, distintos objetivos, intercambiando información con otros agentes humanos o computacionales. Las características destacables del comportamiento de un agente son:

1. Funcionamiento continuo y autónomo.
2. Comunicación con el entorno y con otros agentes (posiblemente humanos), por medio de un lenguaje o formalismo de comunicación.
3. Robustez.
4. Adaptabilidad como capacidad de realizar objetivos y tareas en distintos dominios de forma incremental y flexible.

Otros atributos importantes, aunque no todos los agentes los tienen, son los siguientes:

1. Razonamiento y aprendizaje. Ambos aspectos son necesarios para que el agente sea capaz de comportarse inteligentemente.
2. Movilidad. Los agentes móviles son capaces de desplazarse entre los nodos de una red y ejecutarse en distintas plataformas. (Wooldridge y Jennings, 1995).

Teniendo en cuenta que la definición de agente es muy compleja porque los expertos no se han puesto del todo de acuerto, esto los convierten en unos elementos polivalentes dotados de capacidades bastante superiores a todo lo conocido hasta el momento y que son capaces de hacer funcionar una aplicación sin más que dejarlos que se ejecuten en una máquina (Wooldridge y Jennings, 1998). Si bien es cierto que los agentes nacieron en el área de la inteligencia artificial distribuida, también es cierto que siguen siendo los desarrolladores quienes les dotan de inteligencia y de la capacidad de trabajar en paralelo de forma coordinada.

Situándonos en un plano un poco menos extremo, al software desarrollado con agentes se le confiere, la capacidad de mejorar propiedades como flexibilidad, modularidad o eficiencia (Odell, 2000; Ramdane-Cherif y otros., 2005). Aunque los agentes, vistos como abstracciones que se utilizan para desarrollar software, pueden facilitar que las aplicaciones construidas con ellos

2 Este término ha sido utilizado para definir la personificación o representación de un agente como algo muy cercano o altamente cercano con la figura del profesor real. Generalmente se asocia a agentes conversacionales cuyos gestos y movimientos son tomados y emulados de la realidad.

posean estas propiedades, también es cierto, como en cualquier otro paradigma de desarrollo de software, que es necesario hacer un esfuerzo de diseño para que estas propiedades estén presentes.

Debido a esto, existe una notable confusión respecto a lo que aportan los agentes al desarrollo de software, por lo que, para efectos de este trabajo, se intentarán tener en cuenta las afirmaciones de otros autores que ya se han enfrentado a estos problemas:

Los agentes inteligentes son un noventa y nueve por ciento informática y un uno por ciento Inteligencia Artificial (Etzioni, 1996). Al desarrollar un sistema basado en agentes, el porcentaje del diseño que es específico de la orientación a agentes es comparativamente pequeño (Wooldridge y Jennings, 1998). Consideraremos que un agente es una herramienta de abstracción para desarrollar software que nos permite situarnos un escalón por encima de objetos y componentes (Zambonelli, *y otros.*, 2003) y que, utilizados de forma adecuada, facilitan la construcción de sistemas complejos.

Los agentes pedagógicos heredan todas las dificultades de implementación tanto de los agentes como del software educativo. Si además se utiliza una representación animada para mostrarlo, aparecen problemas nuevos. En primer lugar, los agentes deben mostrar un comportamiento coherente, coordinando su comportamiento con el de otros agentes, y respondiendo de forma lógica a los estímulos de su entorno, incluyendo dentro de éstos a las acciones del usuario. Además, necesitan poseer el conocimiento sobre el dominio que el estudiante está aprendiendo. En general los agentes comunes tienen cierto grado de inteligencia que les permite desenvolverse en su entorno para conseguir sus objetivos. En el caso de los agentes pedagógicos esa inteligencia no consiste en poder resolver los ejercicios que deben solucionar los estudiantes, sino ser capaz de *explicar* cómo se resuelven, dando consejos y ayuda contextualizada. Esto requiere una profunda comprensión de las relaciones entre cada una de las acciones necesarias para solucionar el problema.

Si el agente pedagógico es animado, hay que conseguir una armonía entre sus explicaciones y su representación para hacerlo “creíble”. Con esto no sólo nos referimos a conseguir una sincronización entre el movimiento de la boca y las palabras dichas cuando el agente habla, sino a producir un comportamiento natural y apropiado para el papel que juega dentro del entorno virtual.

Una ventaja adicional de los agentes pedagógicos animados es que pueden hacer uso de comunicación no verbal. Mediante gestos de alegría, duda, o impaciencia, pueden rápidamente y sin necesidad de hablar, indicar al alumno que está haciendo las cosas bien, mal, o que debería

plantearse realizar alguna acción urgentemente. Naturalmente, hay que tener cuidado durante el diseño con el uso de estos movimientos para que el alumno no se distraiga. Además en ocasiones esos movimientos pueden llegar a causar que el agente pierda la ilusión de vida que haya podido crear en el usuario. Si siempre que el alumno hace algo bien el agente ejecuta el mismo gesto (por ejemplo asentir), el usuario podría terminar considerándolo una respuesta automatizada y carente de todo sentimiento. Para evitarlo, los diseñadores suelen meter diferentes animaciones para indicar una misma cosa, eligiendo una u otra de forma más o menos aleatoria. Esa variedad hace que la ilusión de vida se conserve, al menos, una mayor cantidad de tiempo.

Los gestos que el agente realiza tienen también un importante papel para la motivación del estudiante, pues muestran respuestas emotivas del agente. El agente puede felicitar al usuario cuando logra resolver un ejercicio especialmente difícil, y acompañarla con un gesto de alegría. Si el usuario se confunde reiteradamente, su respuesta será la contraria, mostrando una expresión de pena. Todo esto entra en realidad en el campo del *razonamiento afectivo*. Conseguir agentes que muestren emociones a los alumnos tienen varios beneficios para el aprendizaje (ERL99):

- Dan la impresión de que el agente se preocupa por el progreso del estudiante. Esto hace que el alumno sienta que el agente “está con él” en su tarea de aprender, lo que le anima a preocuparse por su propio progreso y la opinión que el agente tiene de él.
- Si el agente es “consciente” de las emociones del estudiante puede animarle y ayudarle cuando detecta su frustración o pérdida de interés.
- Si se consigue que el agente muestre entusiasmo por la materia que está enseñando, el estudiante podría adquirir dicho entusiasmo y encontrar más atrayente el aprendizaje.
- Los agentes con rica e interesante personalidad pueden hacer que el aprendizaje sea más divertido, pues el estudiante puede disfrutar su interacción con él, lo que creará una percepción más positiva del entorno de aprendizaje. De este modo, el alumno tenderá a utilizar más el software educativo, con los claros beneficios que esto supone.

La fuerte presencia visual mostrada por los agentes animados puede, por lo tanto, generar un impacto afectivo en los usuarios que potencie el aprendizaje gracias a lo que algunos autores denominan “el efecto persona”.

Comparando la reacción de un grupo de alumnos ante el uso del mismo software educativo en el que lo único que cambia son las características del agente pedagógico, llegan a la conclusión de que la inclusión de tales figuras mejora la relación entre el alumno y el sistema. Aunque reconocen que su estudio no es muy completo y deberían llevarse a cabo investigaciones más detalladas, aconsejan a los creadores de entornos interactivos de aprendizaje que incluyan agentes pedagógicos animados, incluso aunque éstos no proporcionen consejos.

En su análisis, han descubierto que lo realmente importante para que el alumno recuerde lo que se le está enseñando es que el sistema lea las explicaciones, en lugar de tenerlas que leer él mismo. De ahí deducen que los creadores de software educativo deberían prestar especial atención a la generación del lenguaje hablado, en lugar de a la representación de personajes animados. Su estudio, sin embargo, se centra en el aprendizaje, es decir en la cantidad de información que los alumnos pueden recordar tras utilizar el sistema, pero no en la impresión que les ha causado su uso. En ese sentido parece claro que los agentes pedagógicos hacen considerablemente más amigable la interacción entre el usuario y el sistema, y los estudiantes suelen considerar interesantes los programas de enseñanza que poseen uno de tales agentes.

Lo que hasta el momento nadie ha considerado es cabe la posibilidad de que los resultados de los experimentos que aseguran esto último se vean en realidad falseados por culpa de la “novedad”. Actualmente, los agentes pedagógicos se consideran muy interesantes, y los alumnos los suelen puntuar positivamente. Se concluye por lo tanto que la educación asistida por computadora es buena, y motivante. Lo que no contestan es si lo es en general, o solamente hoy en día. Hoy una figura animada dentro de una computadora que muestra sentimientos es algo nuevo que atrae a los usuarios. Tal vez en unos años se vuelva algo habitual y carente de interés.

Por otro lado los experimentos que tratan de probar los beneficios de los programas educativos a menudo se realizan con grupos de control en colegios. Los alumnos son sacados de la clase con su profesor para hacer uso de las computadoras. Y salir de la rutina diaria es algo que todo el mundo considera interesante.. Si dentro de muchos años se hace común la educación por computadora, quizá lo realmente motivante sean las clases con profesor, por ser lo que se sale de la rutina.

En cualquier caso, lo que en ralidad interesa es desarrollar sistemas lo más completos posibles, que permitan al usuario obtener lo que desea o que faciliten la comprensión o aprendizaje de un dominio, aún considerando que la moda o novedad pueden hacer que se pierda el interés rápidamente por ellos.

Cuando se presenta a los estudiantes un tema por primera vez, suele ser necesario demostrarles cómo resolver problemas relacionados, y el modo de ejecutar ciertas tareas (John, 1998). Eso ha originado que a algunos agentes pedagógicos se les haya incluido la capacidad de poder resolver los problemas que el usuario debe superar. De ese modo, los alumnos pueden también aprender a través de ejemplos. Naturalmente, para que esto valga la pena el estudiante deberá prestar atención a los pasos que el agente pedagógico da, aunque éste no es un problema nuevo y lo han sufrido los profesores durante siglos.

Las demostraciones computarizadas para resolver ejercicios se han realizado habitualmente mediante presentaciones multimedia. Aunque en los sistemas con agentes pedagógicos se podría continuar utilizando esta técnica, en general se prefiere que sea el propio agente el que resuelva el ejercicio porque la demostración puede adaptarse a diferentes estados del entorno, o a distintos estados iniciales. Además, si el entorno de aprendizaje es un entorno tridimensional, el usuario podrá ver la presentación desde cualquier punto, y el agente adaptará sus gestos en función de la posición del usuario. Por último, la capacidad de ejecutar tareas de los agentes pedagógicos puede utilizarse como caso extremo cuando el alumno no sabe continuar a mitad de un ejercicio. El agente podría continuar el trabajo, mostrando cómo realizar sólo partes del escenario completo en lugar de todo el ejercicio como ocurría con las demostraciones multimedia.

Aunque la posibilidad de que el agente resuelva el problema es interesante como último recurso, en general es preferible que antes de llegar a ese extremo pueda proporcionar ayuda con diferentes niveles de detalle. El principal objetivo de los agentes pedagógicos animados es guiar a los estudiantes a través de asignaturas complicadas y, sobre todo, ofrecerles consejos adaptados al contexto del episodio de aprendizaje actual. Esos consejos no deben ser detallados desde el principio, pues hacen que el alumno no necesite razonar cuando pregunta al agente. Es más interesante que en primera instancia el estudiante reciba una respuesta vaga, pero correcta, que le inste a tratar de encontrar la solución por sí mismo. Si aún así, el usuario sigue perdido, puede volver a pedir ayuda, a lo que el agente irá dando información cada vez más concreta. De ese modo, el alumno puede parar de preguntar cuando considere que ya ha entendido su problema, y, si es incapaz de solucionarlo por sí mismo, en última instancia el agente le puede dar todos los detalles o, incluso, solucionarlo él mismo.

Por otro lado los diversos experimentos han demostrados que los alumnos encuentran muy molesto que el agente pedagógico repita la misma explicación, o dé los mismos consejos. Para evitarlo, se recomienda tener al menos dos versiones para la misma explicación, una detallada y otra del estilo de “Recuerda que ... ”, lo que incrementa la ilusión de *inteligencia* del agente. Naturalmente el sistema tendrá que controlar qué explicaciones se han dado y cuando, para no repetirlas a no ser que haya pasado el tiempo suficiente.

Una ventaja adicional de tener varios niveles de ayuda o las explicaciones abreviadas es que pueden ser utilizadas por el agente sin que el usuario lo solicite. Si el usuario muestra dificultades con el ejercicio actual, ya sea porque se equivoca o porque se siente incapaz de realizar ninguna acción y se queda parado, el agente puede dar consejos, no solicitados, para tratar de ayudar al alumno a superar su desagradable situación sin necesidad de preguntar. Naturalmente los diseñadores deben vigilar estas intervenciones automáticas pues en ocasiones pueden resultar innecesarias y molestar al usuario. Eso muestra que hay una delgada línea entre un agente pedagógico útil

y uno que es más una distracción que una ayuda. De hecho, los agentes pedagógicos resultan interesantes en el software cuando éste es educativo y el sistema es utilizado para enseñar. La inclusión de agentes pedagógicos en sistemas que son utilizados por expertos suele ser más una molestia que una ayuda, pues el experto no necesitará prácticamente nunca los consejos de un agente que posiblemente sepa menos que él.

En general, los agentes pedagógicos tienen las siguientes ventajas e inconvenientes, algunas de las cuales son una herencia de sus antecesores, los STI:

- Pueden proporcionar una enseñanza muy parecida a las clases particulares, sin tener los problemas de personal disponible. En ocasiones es necesario enseñar a un gran número de personas en un cierto área. En esos casos, es posible que no haya profesores suficientes para asumir tanta carga docente. Si la enseñanza se realiza a través de computadoras, los problemas de falta de personal calificado se convierten en problemas de recursos tecnológicos, a menudo mucho más sencillos de solucionar. Además los que antes eran tutores pueden dedicarse a realizar su trabajo normal, limitándose a controlar el trabajo de los alumnos y ayudar cuando sea necesario, un trabajo menos exigente que el de dar clase.

- La enseñanza es individualizada, adaptada a las necesidades del usuario.

- Elimina el temor a pedir ayuda, o a preguntar lo mismo varias veces.

- Pueden hacer el aprendizaje más divertido.

- Incrementa el sentimiento de eficacia de los usuarios.

- Pueden hacer uso de comunicación no verbal y mostrar sentimientos que hacen más familiar la comunicación entre el sistema y el estudiante.

- Son buenos enseñando información objetiva con una clara diferenciación entre respuestas válidas e inválidas, o ayudando a solucionar problemas en entornos de aprendizaje constructivista, pero su utilidad enseñando material basado en teoría y opiniones está aún por demostrar. En particular estos sistemas no son aptos para *educar*, aunque puedan enseñar.

- La generación y reconocimiento del lenguaje natural está aún en su infancia, y no está preparada para un uso general. El mismo problema sufre la generación y reconocimiento del habla.

- Los agentes pedagógicos pueden ser considerados una molestia en ocasiones, si no están bien diseñados o la interacción no está bien conseguida.

- Son muy difíciles de implementar. El diseño de agentes pedagógicos animados es un campo activo de investigación muy prometedor.

2.3.1. El Estándar FIPA

Dos de los problemas que deben resolverse cuando aparece una nueva tecnología son: por un lado, la facilidad para interconectar un sistema con otros sistemas basados en ella y, luego, la posibilidad de extender la mencionada tecnología, ampliando la funcionalidad del sistema existente. Para conseguirlo es necesario disponer de estándares, cuya utilización posibilite la consecución de estos objetivos.

En el área de los sistemas multiagente, quien más ha trabajado en esta dirección ha sido la FIPA (Foundation for Intelligent Physical Agents), organización integrada dentro de IEEE (Institute of Electrical and Electronics Engineers)[3] que se ha encargado de elaborar una serie de especificaciones que deben cumplir las plataformas de gestión de sistemas multiagente. Existen algunas plataformas que se han desarrollado a partir de estas especificaciones, como JADE y FIPA-OS, y algunas otras tienden a adoptarlas para conseguir que los usuarios sigan desarrollando sus aplicaciones con ellas.

Son dos los principios que han guiado la elaboración de las especificaciones FIPA. Por un lado, conseguir que los sistemas sean completamente abiertos, de manera que sistemas heterogéneos puedan interactuar como sociedades de agentes. Por otro lado, sólo se han definido las interfaces de los distintos elementos, para que cada equipo de desarrollo pueda tomar las decisiones de diseño que crea convenientes, siempre y cuando se siga cumpliendo con la especificación.

El núcleo de la especificación lo constituye la plataforma de agentes, la cual proporciona toda la infraestructura y servicios necesarios para poder crear un sistema multiagente dentro de ella. El estándar FIPA para la arquitectura de plataformas multiagente (IEEEFIPA, 2002a) define los servicios que deben proporcionar las plataformas de agentes: un sistema de gestión de agentes (Agent Management System), un sistema encargado del transporte de mensajes entre agentes (Internal Platform Message Transport), un servicio de directorio o páginas amarillas (Directory Facilitator) y un canal de comunicación para los agentes (Agent Communication Channel).

---

3 Institute of Electrical and Electronics Engineers, en español Instituto de Ingenieros Electricistas y Electrónicos, una asociación técnico-profesional mundial dedicada a la estandarización, entre otras cosas. Es la mayor asociación internacional sin ánimo de lucro formada por profesionales de las nuevas tecnologías. El propósito principal de la IEEE es fomentar la innovación tecnológica y excelencia en beneficio de la humanidad.

Además, para la comunicación que tiene lugar entre los agentes se ha definido el lenguaje estándar FIPA ACL (Agent Communication Language) (IEEE-FIPA, 2002b), derivado del lenguaje KQML (Knowledge Query and Manipulation Languaje) propuesto por Finin y otros., (1993) Labrou y Finin (1997), que define la estructura que deben tener los mensajes que se envíen entre agentes. Un mensaje ACL contiene una serie de elementos de mensaje, como la identidad del emisor y del receptor, algunos de los cuales son de uso opcional. Además, se permite el uso de elementos no definidos en el estándar. El único elemento que es de inclusión obligatoria es la performativa o acto comunicativo, que describe la intención del mensaje (p.e. realizar una pregunta, solicitar un servicio o proporcionar información), y en función del acto comunicativo variarán el resto de elementos del mensaje.

2.4 Arquitecturas Software

El campo de las arquitecturas software, como área de investigación, es relativamente nuevo. En Shaw y Clements(2006a y 2006b) se hace un repaso de la historia de esta disciplina, y aunque la primera mención de arquitectura software data del año 1969, cuando Ian P. Sharp utilizó el término en una conferencia de la OTAN (Randell y Buxton, 1970), no es hasta mediados de los años 90 cuando de verdad comenzó a despegar, considerando los expertos en el área, que es aproximadamente a partir del año 2000 cuando alcanzó su mayor apogeo (Shaw y Clements, 2006a; Kruchten y otros., 2006).

Se define arquitectura software como la estructura del sistema, que comprende elementos software, las propiedades de esos elementos visibles externamente y las relaciones entre ellos. La arquitectura software conforma el esqueleto de cualquier sistema software, y es la principal responsable de los atributos de calidad del sistema.

Una arquitectura adecuada, correctamente diseñada, documentada y evaluada, constituye la base para que un proyecto finalice con éxito (Bass y otros., 2003). Esta afirmación implica que la arquitectura de un sistema define componentes y la interacción existente entre ellos, pero no detalles internos a esos componentes, que se puede considerar que no pertenecen a la arquitectura de la aplicación. Además, la arquitectura del sistema se puede ver desde muchas perspectivas, todas ellas válidas siempre que cumplan algún fin: análisis, comunicación o comprensión.

Se define al estilo arquitectónico de sistemas como la abstracción de distintas arquitecturas software (Klein y otros., 1999). Así, las conocidas como arquitectura cliente-servidor o arquitectura en tres capas serían, en realidad, estilos arquitectónicos.

Uno de los grupos de investigación más fuertes en este área se encuentra en el Software Engineering Institute (SEI) de la Universidad Carnegie Mellon, que genera gran parte de la producción científica relacionada con las arquitecturas software.

Dentro de este grupo se han desarrollado diversos métodos para diseñar, documentar y evaluar arquitecturas software, como pueden ser el Attribute-Driven Design (ADD), Attribute-Based Architectural Styles (ABAS), Quality Attribute Workshop (QAW), Active Reviews for Intermediate Designs (ARID), Architecture Tradeofi Analysis Method (ATAM), Cost-Benefit Analysis Method (CBAM) y Views and Beyond (V&B). Resultan especialmente interesantes porque no son técnicas independientes, sino que se proporcionan formas de combinar unas con otras (Nord y otros., 2004).

El Attribute-Driven Design (ADD) es un método de diseño arquitectónico dirigido por los atributos de calidad que se quiere que posea el sistema, más que por la funcionalidad de la aplicación, que queda en un segundo nivel como afirman Bass y otros (2001). Está basado en la utilización de dos elementos:

> **Escenarios Generales:** son descripciones generales de situaciones que afectan a cada atributo de calidad por separado. Para cada atributo de calidad se han identificado una serie de escenarios que son válidos para todos los sistemas, aunque eso no quiere decir que sean aplicables en todos los casos. De esta forma, es necesario filtrar los escenarios aplicables y los no aplicables. Posteriormente, es necesario particularizar el escenario para cada sistema en cuestión.
>
> **Primitivas de Atributo:** son un conjunto de componentes y conectores que conforman un elemento mínimo que sirve para mejorar un atributo de calidad. Por ejemplo, una caché sirve para mejorar el rendimiento de un sistema. Aunque cada primitiva está orientada a mejorar un atributo, puede tener efectos secundarios sobre otros atributos. Por tanto, para cada primitiva de atributo es necesario identificar qué escenarios satisface y cuáles son sus posibles efectos secundarios.

Attribute-Based Architectural Styles (ABAS) como proponen Klein y Kazman (1999), por su parte, es a la arquitectura lo que los patrones de diseño a los objetos, es decir, son soluciones basadas en estilos arquitectónicos que han surgido de la experiencia de resolver problemas que se presentan de manera frecuente. ABAS utiliza los atributos de calidad para definir un marco de aplicación de un estilo arquitectónico concreto, proporcionando un razonamiento, cuantitativo o cualitativo, que fundamente la utilización de un estilo arquitectónico en un diseño determinado.

Architecture Tradeofi Analysis Method (ATAM) de Kazman y otros (2000) es la evolución de un método anterior llamado SAAM (Software Architecture Analysis Method) también propuesto por los mismos autores en 1994, para el análisis de arquitecturas software basado en la utilización de escenarios, en el que la evaluación de una arquitectura no se realiza para identificar de manera precisa el comportamiento de un atributo de calidad, lo cual no es posible en fases tempranas del diseño por falta de información, sino para descubrir qué decisiones de diseño afectan de una u otra forma a los atributos de calidad. Esto se hace a través de la identificación de:

- Riesgos: decisiones aplazadas o decisiones cuyo efecto no se alcanza a valorar.
- Puntos sensibles: partes de la arquitectura que pueden tener mucha influencia en algún atributo de calidad.
- Puntos de compromiso: partes de la arquitectura cuya modificación significa mejorar algún atributo de calidad a costa de empeorar otro. Es lo que sucede, en algunos casos, con la modificabilidad y el rendimiento.

El objetivo es poder tomar decisiones razonadas acerca del diseño para, en sucesivos análisis, dedicar más esfuerzo a completar esas partes de la arquitectura.

Active Reviews for Intermediate Designs (ARID) de Clements (2000) es un método de análisis de arquitecturas resultado de combinar Active Design Reviews (ADR) propuesto por Parnas y Weiss (1985) y ATAM. Resulta más ligero que ATAM y es útil para ser aplicado en etapas más tempranas del diseño arquitectónico e, incluso, sobre partes del sistema en lugar de sobre el sistema completo. La evaluación se basa en detallar el funcionamiento del sistema en una serie de escenarios predefinidos, a través de los cuales se pueden identificar posibles puntos problemáticos de la arquitectura.

Su utilidad no se encuentra en la sustitución de ATAM, sino que más bien prepara el camino en sistemas en los que, por su complejidad, puedan ser necesarias revisiones de la arquitectura en etapas intermedias del diseño. QAW propuesto por Barbacci y otros (2003) es un método creado para complementar ATAM. Su utilidad reside en la identificación de los atributos de calidad que dirigen el proceso de diseño de la arquitectura, por lo que su aplicabilidad es previa al diseño arquitectónico. Más concretamente, lo que pretende definir este método es el significado de los atributos de calidad en el contexto del sistema en desarrollo, la manera de descubrir, caracterizar y priorizar los atributos de calidad, y la forma de utilizar esta información. El resultado de este proceso es una lista de factores que van a dirigir el diseño de la arquitectura y una lista de escenarios que servirán para evaluar los atributos de calidad.

Cost-Benefit Analysis Method (CBAM) desarrollado por Kazman y otros en 2002, es un método para evaluar los beneficios, costos y riesgos de las diferentes decisiones que se toman para diseñar la arquitectura software del sistema. Al igual que QAW, también está pensado para su integración con ATAM, ya que los resultados producidos por la evaluación de la arquitectura, especialmente las estrategias arquitectónicas a seguir, se utilizan como entrada en CBAM para tomar decisiones basadas en criterios económicos.

Views and Beyond (V&B) diseñado por Clements y otros (2002a) es la propuesta realizada en el SEI para documentar la arquitectura software de un sistema. De acuerdo con la definición de arquitectura como la estructura o estructuras del sistema, V&B propone la definición de una serie de vistas relevantes de la arquitectura software del sistema, documentando cada una de ellas, así como las características que afecten a más de una o a todas en general. El número y el tipo de las vistas de un sistema no está determinado *a priori*, aunque en general se pueden agrupar en vistas de módulos, vistas de componentes y conectores, vistas de localización y combinaciones de ellas. Para documentar las vistas de una manera sistemática y homogénea han creado unas plantillas que contienen la estructura de la información que debe aportarse sobre cada vista.

En la misma línea se mueve el estándar IEEE 1471-2000 (IEEE, 2000), que define un marco conceptual para describir arquitecturas y la información que debe incluir una documentación que cumpla el estándar. También utilizan el concepto de vista como una representación del sistema desde el punto de vista de un conjunto de intereses, se realiza una comparación entre las dos formas de documentar una arquitectura, y se demuestra que utilizando V&B es posible satisfacer los requisitos impuestos por el estándar IEEE 1471-2000.

Lo más sobresaliente y recientemente aplicado, relativo a estos aspectos se refiere a trabajos dirigidos a la medida cuantitativa de los atributos de calidad de una arquitectura software (Chastek y Ferguson, 2006). Aunque todavía están en un estado muy preliminar, ya se pone de manifiesto la dificultad de tomar estas medidas, debido a las interacciones entre los atributos de calidad, a las interacciones de la arquitectura con otros elementos del proyecto, y a la utilización de vistas para documentar la arquitectura.

2.5 Aplicación de la Realidad Virtual a la Enseñanza

Las aplicaciones basadas en Realidad Virtual (RV), tal como hoy las conocemos, se vienen utilizando en la enseñanza desde que empezaron a desarrollarse los primeros sistemas de RV en los años 60. Sin embargo, su número se ha incrementado notablemente desde principios de los años 90, cuando las computadoras de uso convencional comenzaron a soportar con mayor

facilidad las aplicaciones con interfaces 3D.

Muchos educadores e investigadores piensan que el uso de la Realidad Virtual en la enseñanza puede aportar grandes beneficios a los estudiantes, y prueba de ello es la gran cantidad de aplicaciones de este tipo que se han desarrollado hasta la fecha.

Una cuestión que es importante tener en cuenta es cuándo se debe y cuándo no se debe hacer uso de la Realidad Virtual en un software educativo. Pantelidis (1996) propone un conjunto de criterios para decidir acerca de la aplicación de la Realidad Virtual en la enseñanza y el entrenamiento. El autor y sus compañeros de trabajo proponen que la RV se puede utilizar cuando:

- Se pueda emplear una simulación.
- La enseñanza o el entrenamiento en el mundo real pueda ser peligrosa, imposible o inconveniente.
- Pudieran producirse errores significativos por parte del alumno en el mundo real, errores que pudieran ser devastadores o desmoralizadores para el alumno, perjudiciales para el ambiente, causantes de averías en equipos o costosos.
- El modelo del entorno enseñe o entrene tan bien como la situación real.
- La interacción con el modelo sea igual o más motivadora que la interacción con la situación real.
- Se desee lograr experiencias compartidas por un grupo.
- Sea necesario hacer perceptible lo imperceptible.
- La realización de una clase atractiva que requiera viajes, dinero o logística.
- Si se desea enseñar tareas que involucren destrezas manuales o movimientos físicos.

Por el contrario, la Realidad Virtual no debe usarse cuando:

- La interacción con los humanos reales es necesaria.
- El entorno virtual pudiera ser física o emocionalmente dañino.
- El entorno virtual pudiera causar que el usuario confunda el modelo con la realidad.

Tomando como punto de partida lo anterior y como resultante de la aplicación de la Realidad Virtual, hoy nos encontramos ante un despliegue significativo de entornos virtuales que la aplican como premisa y herramienta básica. Dentro de desarrollos, existe una gran diversidad de aplicaciones de los Entornos Virtuales a la enseñanza y lo que se ha denominado como Entornos Virtuales de Enseñanza-Aprendizaje (EVEA) y que incluye aplicaciones con características muy particulares.

En primer lugar, se puede realizar una distinción primaria en cuanto a su propósito, lo que da lugar a los Entornos Virtuales para Educación y los Entornos Virtuales de Entrenamiento.

Los primeros suelen ser sistemas desarrollados para apoyar el aprendizaje, por lo que suelen estar enfocados hacia la adquisición de conocimientos. Además, suelen estar planteados para que puedan soportar varios usuarios, de manera que el aprendizaje se realice de forma colaborativa, y en muchos casos no incluyen un tutor inteligente, aunque sí contemplan la presencia de tutores humanos como afirma Johnson y otros (1998a y 1999b).

Los segundos suelen estar más enfocados al desarrollo de habilidades de tipo procedimental, y los principales usuarios son adultos que deben adquirir habilidades en su entorno profesional, por lo que es habitual que estos sistemas se encuentren ambientados en entornos industriales (Lozano y otros., 2003), médicos (Lucas y otros., 2008), educativos y militares (Johnson y Valente, 2008).

También se puede realizar una clasificación de estos sistemas en función del tipo de tutoría que incorporan, y es aquí donde surgen las mayores diferencias. Se pueden encontrar sistemas que no incorporan ningún tipo de tutoría, en los que los EVs se usan simplemente como herramienta de prácticas. Existen sistemas de este estilo en entrenamiento médico (Viciana- Abad y Reyes-Lecuona, 2005), militar (Loftin y *otros*., 2004) e industrial (Hosseini y otros., 2002), pero también en Entornos Virtuales educativos para niños (Popovici y otros., 2005) y adultos (Asay-Davis y *otros*., 2000).

En el caso de los sistemas médicos y militares, suele suceder que, aunque no existan tutores inteligentes, sí puede hacer acto de presencia algún tipo de agente más o menos inteligente que hace las funciones de compañero de aprendizaje (Swartout y otros., 2006; Chou y otros., 2003; Muller-Wittig y otros., 2001).

Por el contrario, en los EVs educativos suele ser más habitual que se incorpore algún mecanismo para que un profesor esté presente mientras los estudiantes se encuentran en él. Si se da el caso de que incorporen algún tipo de tutoría, se pueden encontrar sistemas con tutores más o menos cableados, cuya tutoría es poco flexible, y sistemas que incorporan tutores inteligentes que se adaptan a las características y acciones de los estudiantes.

En este último caso se pueden realizar dos distinciones, en función de la presencia o no de un tutor personalizado dentro del EV y del tipo de tutoría que realiza. Si incluyen un tutor con presencia física en el entorno, entonces se habla de agentes pedagógicos, sobre los cuales se dan más detalles en la próxima sección.

Respecto al tipo de tutoría, en general los sistemas se pueden clasificar en dos tipos: los que basan en el análisis de las acciones que realiza el usuario (Buche y otros., 2004), que requi

de unas capacidades de planificación y adaptación complejas y los que se basan en el diálogo con el tutor (Graesser y otros., 2008), que conllevan un esfuerzo notable de modelado de las posibles situaciones que se dan en un diálogo y que todavía implican ciertas limitaciones en cuanto al establecimiento de turnos para hablar. Como menciona Rickel y sus colaboradores (2002a), existen sistemas que intentan aplicar ambos tipos de tutoría, pero aún son escasos y están poco desarrollados.

Con base en las distintas características que se han descrito, los sistemas que resultan de mayor interés para el desarrollo de esta tesis son los Entornos Virtuales que incorporan tutores inteligentes, no necesariamente presentes en el EV, y que se basen en el análisis de las acciones que realizan los usuarios dentro del EV.

A pesar de ello, en la sección de trabajo relacionado se analizan sistemas de todos los tipos aquí mencionados, ya que todos ellos presentan características que puede ser interesantes incluir en la arquitectura modelo que se pretende plantear con la elaboración de la presente tesis doctoral.

2.6 Agentes Pedagógicos

Los agentes pedagógicos son agentes inteligentes, y como su desempeño es en un entorno virtual, suelen denominarse también agentes virtuales, por lo que al hablar de ellos muchas veces se habla también de agentes pedagógicos virtuales inteligentes. Aunque un agente pedagógico no necesariamente debe poseer forma humana, diversos estudios han demostrado que sí debe poseer características antropomórficas, es decir, se le deben poder atribuir características habituales del comportamiento humano. Un agente pedagógico virtual habita un entorno virtual que es el medioambiente en el que se desenvuelve; al ser inteligente posee las características asociadas a un agente inteligente: reactividad y autonomía en su actuar, capacidad de tomar decisiones en pro del logro de sus objetivos, entre otras.

Como primer antecedente histórico, los libros cuentan historias de cabezas parlantes pertenecientes a Roger Bacon, a Carlo Magno, al maestro de Tomás de Aquino, al obispo Grosseteste y al Papa Silvestre II, entre otros. Por otro lado, en el siglo XVI, el gran médico y alquimista Paracelso daba la receta para la creación de un *homúnculo*, un diminuto ser humano viviente.

Remontándonos a la historia, se podría considerar que los primeros hombres virtuales fueron los autómatas construidos en los siglos pasados. Desde la Grecia antigua existe la noción de autómata: en efecto el término «automatos» significaba «que se mueve por sí mismo» y era sinónimo de misterio y fascinación. En la historia de los autómatas, durante el siglo XVIII, se

destacan la creación de piezas excepcionales: los androides. Estos autómatas con formas humanas son el resultado del genio y la habilidad de creadores magistrales como Vaucanson en Francia y Jaquet-Droz en Suiza.

Los primeros hombres mecánicos fueron las figuras móviles de los campanarios de fines de la Edad Media. Encima de la *Piazza San Marco* hay dos grandes figuras que golpean una campana para dar las horas. La utilización de las máquinas era en cierta forma una magia disfrazada, una manera de hacer aceptable lo que todavía era imposible. Se decía que Regiomontano construyó un águila artificial que voló para saludar al emperador Maximiliano cuando éste se acercó a Nuremberg en 1470; se supone que Leonardo da Vinci construyó un león artificial en 1500 para el rey de Francia, Luis XII. Se dice que hacia 1640, Descartes creó un autómata que guardaba en una caja y transportaba siempre consigo, hasta que en un viaje un capitán le obligó a lanzarlo a la mar.

Al tratar de relacionar estos intentos con la creación de humanos virtuales en nuestras modernas aplicaciones debemos remontarnos a la creación de las primeras computadoras. Aunque las computadoras digitales están sustentadas en conceptos desarrollados en siglos pasados, es entre los años 30 y los 40 cuando quedan realmente disponibles para los investigadores. Se dice que la primera persona que concibió el uso de la computadora más allá de sus aplicaciones militares, como una herramienta para transformar el pensamiento y la actividad creativa del hombre, fue Vannevar Bush en 1945. En su trabajo “As we may think”, describe las dificultades que el hombre tiene para conseguir, revisar, manejar y transmitir los resultados de su investigación. Para resolver este problema el inventa MEMEX, un dispositivo con aspecto de escritorio, con teclado y conjunto de botones y manillas, donde se podía guardar libros, registros y comunicaciones para consultarlos de manera rápida y flexible a través del uso de microfilms.

MEMEX fue la primera versión del computador personal. Bush no sólo fue un visionario de la aplicación de la computadora para almacenamiento y recuperación de información, y el valor del indexamiento asociativo en esa actividad, sino también anticipó la naturaleza multimedia del uso del computador en el futuro.

Hoy sabemos que los agentes pedagógicos son el resultado de la combinación de dos áreas de investigación como son los agentes de interfaz animados según refieren Cassell y otros. (2000), Hayes-Roth y Doyle, (1998); Ball y otros., (1997); André, (1997) y los Sistemas Tutoría Inteligente tal como lo expresan Wenger(1987), Sleeman y Brown (1982). La primera de las dos áreas proporciona una metáfora de interacción hombre-computadora, a través del diálogo con un agente animado, mientras que la segunda se ocupa de la creación de tutores inteligentes.

Aunque su utilización no se encuadra únicamente dentro de las aplicaciones de la RV a la enseñanza, ya que existen ejemplos de su uso en aplicaciones web (Shaw y otros., 1999b), muchos de los sistemas que utilizan estos agentes lo hacen por medio de la utilización de Entornos Virtuales. La primera referencia sobre su uso data del año 1990 del trabajo de Chan y Baskin, aunque su aplicación se ha extendido desde 1997, año en el que el número de publicaciones en este área comenzó a incrementarse notablemente (Lester y otros., 1997a; Lester y otros., 1997b; Rickel y Johnson, 1997; Alem y otros., 1997).

Los agentes pedagógicos pueden desempeñar distintos papeles dentro de una aplicación de enseñanza, y no únicamente el de tutores. Así, se han utilizado como compañeros de equipo según lo refieren Chan (1996), Rickel y Johnson (2002) Chou y otros, (2003), como generadores de estrategias pedagógicas (Person y otros., 2001) o como ayudantes de instructores y compañeros (Marsella y Johnson, 1997; Rickel y Johnson, 1999).

Uno de los aspectos destacables de estos agentes es el llamado «efecto persona», identificado por Lester (1997), donde se muestra cómo los estudiantes mejoran su aprendizaje cuando cuentan con la ayuda de un agente pedagógico de aspecto humano con presencia física dentro del Entorno Virtual. Posteriormente, otros investigadores como Baylor (2001), Moreno y otros (2001) han identificado este efecto en estudiantes que utilizan aplicaciones con agentes pedagógicos, si bien otros investigadores atribuyen los beneficios a otros factores, como la estrategia pedagógica, más que al uso de agentes pedagógicos (Choi y Clark, 2006). Uno de los aspectos que han centrado más la atención en éste área ha sido la posibilidad de dotar a los agentes pedagógicos de distintas capacidades que se asemejen a las humanas como lo refieren Swartout y otros., (2006):

> ***Emociones:*** algunos autores argumentan que la capacidad de mostrar emociones facilita la relación con el estudiante y le motiva para continuar con el aprendizaje (Elliott y otros., 1999). Los agentes pueden mostrar emociones a través de expresiones faciales, movimientos, gestos y tonos de voz.
>
> ***Comportamiento no verbal****:* esta característica le da al agente la posibilidad de ofrecer respuestas al estudiante sin necesidad de interrumpirle, como en el caso de asentir con la cabeza para indicar que se comprende lo que el estudiante dice. También se puede utilizar para indicarle al alumno la opinión del agente sobre una acción que ha realizado, como una negación con la cabeza cuando se ha realizado una acción incorrecta (Rickel y Johnson, 1999).
>
> ***Conversación no verbal:*** cuando dos personas mantienen una conversación, resulta

habitual que realicen gestos que remarquen lo que se está diciendo con palabras, y el tono de voz también cambia el significado de una frase. Esta capacidad proporciona a los agentes la posibilidad de darle distinto significado o de enfatizar lo que le dicen al estudiante como afirma Cassell (2000).

***Conversación con lenguaje natural**:* aparte de las interacciones no verbales, también es deseable que los agentes pedagógicos puedan comunicarse con los estudiantes con una comunicación oral flexible que se adapte a la situación, en forma de explicaciones, preguntas o respuestas como lo expresan Graesser *y otros* (2005b).

Percepción: para que los agentes puedan presentar un comportamiento similar al humano, una de las características necesarias es que perciban el entorno de forma similar a como lo haría un humano (Kim y otros., 2005; Herrero *y otros*., 2005). De esta forma, deben poder ver las cosas que están en su campo de visión, pero no las que están ocultas. Igualmente, pueden captar sonidos distinguiendo su lugar de procedencia, lo que les permite, por ejemplo, girarse hacia un estudiante cuando oyen que éste les está hablando.

Además de estas características, los agentes pedagógicos aportan una serie de beneficios que los hacen útiles dentro de los sistemas de enseñanza. Por una parte, ofrecen la posibilidad de utilizarlos para demostrar la forma de llevar a cabo una determinada tarea dentro del EV (Rickel y Johnson, 1999). Por otra, se pueden utilizar movimientos, miradas y gestos para dirigir la atención del estudiante a un determinado punto o para guiarle dentro del EV (Noma y Badler, 1997; Lester *y otros*., 1997b; Cassell *y otros.,*2007).

Todas estas características, como afirman Johnson y otros (2000) hacen que los agentes pedagógicos presenten dos ventajas sobre otro tipo de sistemas: incrementan las formas de comunicarse con un estudiante y aumentan la capacidad de la aplicación para motivar a los estudiantes. Sin embargo, la cantidad y complejidad de las características requeridas hacen que existan numerosas cuestiones abiertas a la espera de ser resueltas de manera satisfactoria, como la percepción del entorno y la interacción con él, la generación de comportamientos complejos y creíbles, el mantenimiento de un diálogo elaborado con el estudiante o el manejo eficiente de todos estos aspectos en su conjunto (Gratch y otros., 2002).

2.6.1. Agentes y Enseñanza

En los últimos años hemos sido testigos de muchos proyectos que han invertido muchos esfuerzos en el diseño de agentes para la enseñanza cada vez más complejos y elaborados. A continuación

se presentan los que a juicio de la autora han resultado ser los más eficientes y representativos, ya sea por la temática que abordaron, las características que comportan o bien el modelado completo del sistema y sobretodo los atributos del agente utilizado en la interfaz. En este punto se expone un resumen esquemático, a manera de tabla, mencionando los principales atributos y funciones de cada uno. Esta tabla se complementa con información detalla en el ANEXO 3.

Tabla 2.1: Principales Agentes de Enseñanza

NOMBRE DEL AGENTE	Tecnología y Arquitectura	**Funcionalidades y Atributos**
Adele (Agent for Distance Education - Light Edition)	Desarrollado para que funcione en un navegador web. La arquitectura de es, en primera instancia, de tipo cliente/servidor, aunque gran parte de la lógica de la aplicación permanece en el cliente. El sistema se compone de cuatro módulos, como son el agente pedagógico, la simulación, el módulo de comunicación cliente/servidor y el almacén central del servidor.	Es un agente pedagógico utilizado para enseñar a realizar diagnósticos y tratamientos médicos, el estudiante puede realizar diversas acciones sobre el paciente, como realizar preguntas sobre el historial médico, realizar exámenes físicos, ordenar la realización de pruebas y, finalmente realizar el diagnóstico.
Auto Tutor	Constituye posiblemente el sistema que cuenta con un mayor y más serio soporte de teorías cognitivas y del aprendizaje. El sistema tiene una arquitectura cliente/servidor con comunicación asíncrona, similar a un sistema web de tipo cliente ligero. Cuando un cliente se comunica con el servidor, le envía un objeto estado que contiene toda la información relativa al estado de la sesión: historia, acción actual, etc., de manera que el servidor es independiente del número de clientes que estén conectados a él. Es un sistema de enseñanza basado en el diálogo, el elemento principal de su arquitectura lo constituye el gestor de diálogos (Dialogue Manager).	Es un sistema de enseñanza basado en el diálogo que simula a un tutor humano y que ha sido desarrollado en el Instituto de Sistemas Inteligentes de la Universidad de Memphis, es complejo porque incorpora un modelo de conocimiento similar al humano y un proceso cognitivo, y utiliza estos mecanismos psicológicos para promover el aprendizaje. Existe también una versión que interactúa con entornos 3D, pero la capacidad de interacción por parte del alumno no es muy elevada, y básicamente sirve para que el sistema muestre el comportamiento de un problema determinado cuando el estudiante ajusta distintos parámetros. Por tanto, se puede considerar más como una interfaz que como un entorno de aprendizaje.

PPP Persona (Personalized Plan-Based Presenter Persona)	Basado en una arquitectura cliente-servidor.	Desarrollado en el Centro de Investigación en Inteligencia Artificial de Alemania (DFKI). No es un agente pedagógico propiamente dicho, su propósito principal consiste en desarrollar un agente capaz de realizar presentaciones en un entorno web, lo cual le confiere la capacidad de proporcionar explicaciones a un estudiante.
Vincent	Cada uno de los capítulos que integran el dominio contiene páginas de teoría, de demostración y ejercicios, y es durante la realización de los ejercicios cuando Vincent monitoriza al estudiante. Es un agente pedagógico utilizado en un entorno llamado TEMAI, un sistema de entrenamiento basado en web, que se utiliza para dar formación a trabajadores de una fábrica de calzado para que aprendan a controlar las líneas de producción. TEMAI consta de cuatro módulos: una base de datos de material de aprendizaje, utilizada para proporcionar explicaciones y ejercicios, un conjunto de entornos de aprendizaje, un modelo probabilístico del estudiante y Vincent, el agente pedagógico.	El encargado de mantener el modelo del estudiante actualizado es Vincent. El comportamiento de Vincent está dividido en cognitivo y físico. El cognitivo está controlado por un módulo que se encarga de las tareas pedagógicas, tareas de gestión del conocimiento experto (especialmente del modelo del estudiante) y tareas de diagnóstico. El comportamiento físico está controlado por otro módulo, que cambia la postura y las frases que dice Vincent en función de la salida del módulo cognitivo.

Doris	Los estudiantes interactúan con el STI a través de una interfaz web, y Doris monitorea esta interacción y envía la información recogida a un sistema de razonamiento basado en casos. Doris tiene la capacidad de percibir el entorno en el que habita y modificarlo, posee autonomía, capacidad de interactuar con el estudiante, capacidad de adaptarse a cambios en el entorno (i.e. estrategia de tutoría), movilidad (entre la máquina del estudiante y el servidor), y conocimiento acerca de los estudiantes con los que trabaja. La arquitectura consta de tres módulos más una base de conocimiento.	Doris es un agente pedagógico que actúa en un entorno web y cuyo objetivo es investigar los mecanismos, herramientas y métodos que se pueden utilizar para proporcionar una educación que saque partido a las características de la web. Doris presenta dos tipos de comportamiento: cognitivo y reactivo. El cognitivo es el encargado de enviar mensajes al estudiante, y de almacenar información acerca de él en la base de conocimiento. El reactivo, a su vez, es el encargado de las acciones que manipulan la apariencia de Doris, seleccionando una apariencia en función de la situación actual.
FILIP	La arquitectura de FILIP se compone de siete agentes: cinco de ellos se corresponden con los habituales de un STI, mientras que los otros dos, no parecen ser más que dos módulos de comunicación.	Es una plataforma multiagente para el desarrollo de sistemas de aprendizaje inteligentes basados en simulaciones, orientado a la adquisición y desarrollo de habilidades por parte de distintos tipos de operadores. Se ha utilizado para desarrollar un sistema llamado ATEEG (Automation of Trainee Evaluation and Exercise Generation) (Alem y Keeling, 1996), que se emplea para la instrucción de controladores aéreos. ATEEG utiliza un pizarra como medio para compartir información entre los distintos agentes, utilizando para ello un servidor de objetos que almacena los objetos compartidos por varios agentes.

DIT (Distributed Intelligent Tutor)	Está formado por un grupo de cinco agentes que se distribuyen la responsabilidad de la tutoría: el agente pedagógico, el generador de ejercicios, el resolutor de ejercicios, el explicador y el corrector.	Es un tutor inteligente capaz de planificar una lección, generar ejercicios y plantear y explicar las soluciones a los ejercicios (Khoualdi y Benghezal, 2004). El agente pedagógico realiza labores de coordinación y supervisión del resto de agentes. Además, también está encargado de la comunicación con el usuario. Para llevar a cabo el proceso de enseñanza cuenta con una librería de planes: unos están orientados a planificar el orden en que se enseña una materia, mientras que los otros están orientados a la manera en que se enseña un determinado concepto.
Baghera	La arquitectura del sistema está diseñada en dos niveles, el primero de los cuales conforma la estructura de la aplicación, mientras que el segundo comprende agentes que se dedican a evaluar a los estudiantes. La aplicación está concebida como un sistema abierto, de manera que el número de agentes existente no es fijo	Baghera es un proyecto desarrollado para estudiar el comportamiento emergente de un sistema basado en agentes, aplicado a la enseñanza de geometría por medio de una interfaz web (Webber y Pesty, 2002; Webber y otros., 2002; Webber y otros., 2001). Cada estudiante cuenta con la asistencia de 3 agentes: el agente personal de interfaz que tiene como principal objetivo monitorear las acciones del alumno y comunicárselas a otros agentes; el agente tutor, encargado de plantearle al alumno distintos problemas en función de los objetivos actuales, basándose en los conocimientos que posée. Y el agente mediador, responsable de elegir el resolutor de problemas más adecuado para enviarle la solución del alumno al problema planteado.

I-Help	I-Help es una actualización de un sistema anterior llamado PHelpS (Peer Help System) (Collins y otros., 1997), cuya arquitectura no estaba basada en agentes y era de tipo cliente/servidor, en lugar de distribuida. Este sistema se utilizaba para entrenamiento en lugares de trabajo. La arquitectura se basa en dos tipos de agentes: agentes personales y agentes de aplicación, que comparten una ontología y un lenguaje de comunicación comunes.	Es un sistema multiagente distribuido utilizado para que los alumnos de un curso puedan proporcionarse ayuda mutua (Vassileva y otros., 1999; Vassileva y otros., 2003). A diferencia de otros sistemas, no proporciona tutoría inteligente, sino que ayuda a que un estudiante encuentre la mejor ayuda posible de entre sus compañeros y otros recursos disponibles.
INES (Intelligent Nursing Education Software)	El sistema está construido, en dos estilos arquitectónicos. El primero de ellos, abstracción de datos y organización orientada a objetos, se utiliza para representar los objetos del entorno, como jeringas y agujas, y los agentes que conforman el sistema. El segundo estilo, basado en eventos con invocación implícita, se utiliza para la implementación de los agentes. La arquitectura de INES consta de 4 capas: La primera de ellas controla los dispositivos de entrada. La siguiente capa, la interfaz de usuario concreta, se ocupa de la visualización de los ejercicios. La tercera capa es la interfaz de usuario abstracta, una representación de alto nivel de lo que se muestra en la interfaz concreta. La cuarta capa constituye el núcleo central de INES, y es donde se encuentran los agentes y donde se realiza el proceso de tutoría que además, está constituido por tres modelos: uno de dominio, otro del estudiante y otro de instrucción.	sistema basado en agentes desarrollado para ayudar en la formación de estudiantes de enfermería (Hospers y otros., 2004). se ha desarrollado con el objetivo de que sea fácil de usar y que además pueda proporcionar demostraciones, explicaciones y realimentación a los estudiantes. Actualmente, la interfaz de la aplicación está basada en web, aunque parece que existía el propósito de conectar la aplicación con entornos 3D.

GIA (Generic Instructional Architecture)	Su objetivo es desarrollar una arquitectura y una serie de componentes software que proporcionen unas bases para el desarrollo evolutivo de STIs (Cheikes, 1995). La arquitectura de GIA es lo que autores como Genesereth (1993) llaman una arquitectura federada en la que existen unos agentes facilitadores que proporcionan servicios a otros agentes, facilitadores o no, formando una comunidad. Un servicio fundamental es el reenvío de mensajes en función de su contenido. Los agentes que componen GIA proporcionan los servicios básicos de un SIT, lo que incluye un registro del estudiante (administrador de entrenamiento), planificación de lecciones (gestor de curriculum), administración de ejercicios (gestor de actividades), toma de decisiones de entrenamiento (planificador de entrenamiento) y modelado del conocimiento del estudiante (modelo de estudiante).	La idea principal de GIA se basa en que un STI se puede diseñar como una colección de agentes independientes que colaboran e intercambian información a través de un lenguaje. Además, los autores destacan que la clave de todo se encuentra en el lenguaje de comunicación. La interfaz de usuario está modelada como una serie de agentes de interfaz, cada uno de los cuales es responsable de gestionar los distintos recursos que utiliza el estudiante.
Jonás	Enxuto se compone de tres módulos: uno de modelado, otro de simulación y un agente pedagógico. Jonás está compuesto por un sistema experto, que consta de una base de conocimiento y una máquina de inferencia, una conexión con la interfaz de usuario, la base de datos de modelos y los resultados de la simulación, y un sistema de administración.	Es un agente pedagógico que forma parte de un sistema llamado Enxuto desarrollado para entrenar a trabajadores de talleres en la creación de nuevos procesos de fabricación de productos (Baranauskas y otros., 1997; Baranauskas y Borges, 1997; Borges y Baranauskas, 1997).

ABITS (Agent Based Intelligent Tutoring System)	ABITS es el módulo que permite transformar un sistema personalizado en uno adaptativo. El conocimiento está representado a través de Objetos de Aprendizaje (Hodgins, 2002) los cuales son utilizados también para realizar el modelado del estudiante. A través de los atributos de los objetos de aprendizaje utilizados se realiza un perfil de las preferencias del estudiante. Externamente, ABITS se comunica con el sistema de gestión del curso, al que le proporciona los objetos de aprendizaje que hay que presentarle al estudiante, y con una serie de bases de datos de donde obtiene información acerca de los usuarios y de los posibles objetos de aprendizaje que se pueden utilizar. Internamente, la arquitectura de ABITS se basa en la utilización de grupos de tres tipos de agentes: agentes de evaluación, agentes afectivos y agentes pedagógicos.	Es un Sistema Inteligente de Tutoría que funciona en un entorno web y que es capaz de realizar modelado del estudiante y generación automática de temarios (Capuano y otros., 2000).

2.6.2. Entornos Virtuales de Enseñanza-Aprendizaje

De la misma manera que han sido elegidos y presentados los agentes pedagógicos que a juicio de la autora resultan los más representativos, también a continuación se presentan una tabla que resume las cualidades y atributos de Entornos Virtuales de Enseñanza-Aprendizaje, que por sus particularidades se considera importante referir. Este apartado igualmente, se complementa en el ANEXO 4, dónde se incluye información puntual de cada uno de ellos, así como imágenes representativas de su interfaz gráfica.

Tabla 2.2 Entornos Virtuales de Enseñanza-Aprendizaje

NOMBRE DEL EV	Tecnología y tipo de EV	Funcionalidades y Atributos
GCW (Global Change World)	Tiene como objetivo el estudio de la colaboración y el aprendizaje usando como herramienta un EV (Jackson y Fagan, 2000). Para ello, se ha diseñado una aplicación que permita aprender conceptos sobre calentamiento global en un EV que representa a la ciudad de Seattle. Funciona sobre dos máquinas Hewlett-Packard 9000 que ejecutan el software de RV DVISE. La interfaz de usuario está formada por un casco de RV, un control de navegación y un sensor de posición, todos ellos de Division Corporation, además de auriculares y micrófonos para permitir la comunicación entre los habitantes del EV.	La reproducción de la ciudad de Seattle se ha hecho de una manera selectiva, de forma que sólo se ha hecho un modelo del centro de la ciudad con los edificios más representativos. Para causar cambios en el medio ambiente, los estudiantes pueden ajustar tres grandes ruedas que aparecen en el EV y que representan: la cantidad de plantas que existen en el mundo, la cantidad de industrias que hay en el mundo y el impacto causado por la población mundial.
NICE (Narrative, Immersive, Constructionist/ Collaborative Environment)	La aplicación consta de un jardín central al que puede conectarse un número ilimitado de clientes, los cuales pueden hacer uso de distintos tipos de interfaces. Cada usuario es representado por un avatar con cabeza, cuerpo y manos, y si los usuarios están utilizando sensores de movimiento, el sistema puede reproducir sus movimientos como gestos del avatar correspondiente. Para transmitir la información del jardín que controla el crecimiento de las plantas se ha utilizado el protocolo TCP/IP, mientras que para transmitir la posición de los avatares se utiliza o bien UDP o bien multicast. El segundo se utiliza dentro de subredes, mientras que el primero se prefiere para conectar clientes fuera de una subred.	Es una iniciativa que pretende crear un EV que explote la colaboración como forma de motivación para el aprendizaje. Es una aplicación dirigida a niños cuyo objetivo es que creen un ecosistema y colaboren a través de la red para facilitar esta tarea. Además, se pretende que creen historias a partir de la colaboración que ha tenido lugar en el EV (Johnson y otros., 1998a; Johnson y otros., 1999a).

REP (Round Earth Project)	Es un EV que promoueve la colaboración. Para que los niños pudiesen cambiar su concepción de que la Tierra es plana y entendiesen que es redonda, se creó un EV en el que dos niños pudiesen colaborar, de manera que uno pudiesen concentrarse en observar mientras el otro hacía algo que le obligase a desplazarse por la Tierra y acabar apareciendo boca abajo para el otro niño. Además, se obligó a que la tarea de cada niño fuese necesaria para la del otro, forzando la comunicación y cooperación entre los dos.	Persigue el estudio de cómo se puede usar la RV para enseñar conceptos que son contrarios a las concepciones que ya tiene el alumno (Johnson y otros., 1999b). En concreto, se ha utilizado para enseñar a niños que la Tierra es redonda, cuando en la vida diaria todo parece apuntar a que es plana.
CVE-VM (Collaborative Virtual Environment-Virtual Museum)	De esta forma, CVE-VM es un EV que funciona en Internet y que permite el aprendizaje colaborativo. Está desarrollado con Java[1] y VRML[2], y la comunicación entre los dos se realiza a través de EAI (External Authoring Interface). Con VRML se proporciona toda la representación 3D, mientras que Java se encarga de manejar todos los eventos y mensajes que aparecen en el EV. La aplicación utiliza un servidor cuyo principal objetivo es mantener la conexión con los usuarios y controlar la introducción de objetos en el EV.	Es una de las partes que integra el proyecto Virtual Museum (Wazlawick y otros., 1999), cuyo objetivo es la creación de un EV colaborativo que ayude a la enseñanza en los colegios brasileños. No se persigue que el sistema haga las veces de tutor, sino que constituya una herramienta que los profesores puedan utilizar para proponer actividades que deben realizar los alumnos y así ir adquiriendo conocimiento (Kirner y otros., 2001). La comunicación entre los usuarios tiene lugar a través de un chat, con un aspecto similar al de los chats convencionales: un área para escribir los mensajes, otra para mostrar los mensajes de todos los usuarios y una lista con todos los usuarios conectados que ofrece la posibilidad de mantener conversaciones privadas con otros usuarios.

Herman the Bug	Este entorno ha servido para identificar el llamado efecto persona (Lester *y otros*., 1997c), por el que se muestra una mejora del aprendizaje por el hecho de supervisarlo mediante un tutor animado. Se ha dotado al agente de: **Espacio de comportamientos:** es una librería de gestos y movimientos que el agente puede realizar y de frases y sonidos que puede emitir. **Modelo del contexto:** es una base de conocimiento que contiene información sobre el problema a resolver, un historial de explicaciones y consejos que se le han dado al alumno y un modelo del artefacto que está siendo construido por el alumno. **Motor de secuenciación de comportamientos:** construye las acciones del agente utilizando el espacio de comportamientos en función del conocimiento contenido en el modelo del contexto.	Es un agente pedagógico desarrollado por el grupo dirigido por el profesor James Lester en Intellimedia (The Intellimedia Center for Intelligent Systems, North Carolina State University) (Lester *y otros.,* 1997c; Lester *y otros.,* 1999c). Herman es un insecto que hace las veces de tutor en Design-a-Plant, un EV diseñado para enseñar conceptos sobre anatomía y fisiología botánica. El objetivo principal de este entorno es investigar la interacción entre agentes y estudiantes para resolver problemas (mixed-initiative problemsolving) en entornos de aprendizaje constructivista (Piaget, 1954).

Cosmo	Se ha creado un planificador de comportamiento que, tras la petición de una explicación o una pista, selecciona la explicación y construye una secuencia de acciones para proporcionársela al alumno. Este trabajo se basa en trabajos anteriores sobre animación y credibilidad de agentes pedagógicos (Stone y Lester, 1996; Lester y Stone, 1997). Este agente pedagógico se ha probado en un EV llamado Internet Advisor, en el que Cosmo enseña conceptos relacionados con redes IP: cómo se construyen las direcciones, cómo se enrutan los paquetes, qué hace un router, etc., y la misión del estudiante es hacer que un paquete llegue desde una computadora hasta su destino. La representación de Cosmo en el Internet Advisor tiene una serie de partes móviles, como la cabeza, una antena, los ojos, los brazos y las manos, lo que le permite adoptar distintas expresiones y realizar distintos gestos, como coger objetos o señalarlos con el dedo.	Es un agente pedagógico animado desarrollado por el mismo grupo que Herman the Bug (Lester y otros., 1997b; Lester y otros., 1999a). El objetivo principal es el aumento de la credibilidad de este tipo de agentes. Para que la sensación de credibilidad sea mayor, es necesario que los agentes pedagógicos aprovechen su conocimiento acerca de la posición de los objetos en el EV, su posición relativa a otros objetos o sus explicaciones previas para generar gestos, movimientos y explicaciones que resulten más naturales (Lester y otros., 1999a). Cosmo monitorea las acciones de los estudiantes e interviene en dos situaciones: cuando el estudiante está mucho tiempo sin hacer nada o cuando se equivoca.

2.6.3 Entornos virtuales de entrenamiento

En la siguiente tabla se revisan los principales EV creados para el entrenamiento, especialmente aquellos diseñados con el fin de colaborar en los procesos para mejorar las habilidades procedimentales y en el ANEXO 4 se incluye información adicional sobre ellos.

Tabla 2.3 Entornos Virtuales de Entrenamiento

Nombre del EV	Tecnología	Atributos y Funcionalidades
STEVE (Soar Training Expert for Virtual Environments)	STEVE y el estudiante cohabitan en un escenario tridimensional que simula el entorno de trabajo donde el estudiante tendrá que realizar su labor y STEVE puede demostrarle cómo realizar las tareas o puede supervisar la realización de las mismas por parte del estudiante, así como proporcionarle ayuda cuando éste lo requiera. La arquitectura del EV, consta de una serie de componentes independientes que pueden ejecutarse como procesos separados o incluso en máquinas distintas, y se comunican entre ellos con paso de mensajes (Johnson y otros., 1998b). STEVE se compone de tres módulos (Rickel y Johnson, 1999): **Módulo de Percepción:** monitoriza el estado del EV, mantiene una representación coherente del mismo y proporciona esta información a los otros dos módulos. **Módulo Cognitivo:** interpreta la entrada del Módulo de Percepción, elige las metas apropiadas, construye y ejecuta planes para conseguir estas metas y envía órdenes al Módulo de Acción. **Módulo de Acción:** controla la voz, movimientos, mirada y gestos de STEVE, y le permite manejar los objetos del EV.	Es un agente animado diseñado específicamente para ayudar a uno o varios estudiantes a llevar a cabo labores físicas que sigan un procedimiento determinado (Rickel y Johnson, 1999). Este sistema incorpora características tales como la síntesis de voz, para dar las explicaciones pertinentes a los alumnos, el reconocimiento de voz, de manera que el alumno puede comunicarse con STEVE de manera oral, y el uso de cascos de RV, lo que posibilita, por una parte, que el alumno se pueda mover de una forma más realista por el EV, y por otra, que STEVE pueda saber fácilmente hacia dónde está mirando el alumno para detectar si está atendiendo a las explicaciones o si se encuentra distraído o incluso falto de motivación.

MAKATSINÁ	El sistema Makatsiná centra su enseñanza en el aprendizaje de la habilidad para resolver estructuras triangulares por el método de los nodos, de acuerdo a la filosofía reactiva Beer (1990). Este Sistema de Entrenamiento Inteligente (SEI) utiliza al experto compilado, en el que se combina este último con reglas, básicamente por el tipo de proceso tutorial que se realiza: entrenador-deportivo, aplicado a una tarea cognitiva que conlleva integrar habilidades.	Es un Sistema de Enseñanza Inteligente (SEI) creado en México por Laureano y de Arriaga (2000), donde se utiliza un modelo cognitivo cuyo análisis y diseño se desarrolló de acuerdo a las investigaciones realizadas por Castañeda (1993), Redding (1992) y Ryder & Redding (1993).

2.7. Proceso de Interacción hombre-máquina

La palabra interacción, sencillamente, refiere a un sistema previo que puede organizarse de manera formal o informal y por ello, se menciona que la interacción, en dicho sistema, realiza procesos de intercambio en sentido amplio.

En el tema que nos ocupa, la Interacción, es un término que se refiere a una relación dada entre el ser humano o la persona y la máquina a través de una interfaz. Esta definición está configurada en la comprensión que lleva al ser humano a realizar una extensión de sus capacidades. Por la extensión de nuestras capacidades por medio de las máquinas, se entiende las ventajas que dan al ser humano para realizar otras tareas concomitantes, dejando las rutinarias o de tipo autómata a las máquinas. Además por extensión se comprende la posibilidad de realizar tareas que comprendas a las máquinas como interfaz para la comunicación directa o indirecta con otros seres humanos.

En esta relación de hombres o personas y máquinas, se comprende que las interacciones en sí, se relacionan con los procesos internos automáticos del ser humano. Estos procesos internos son rutinas de procesamientos de la información, así las máquinas llevan en sí algoritmos que procuran mejorar el desempeño de la persona y aumentar su inteligencia, como asimismo sus niveles de conciencia, dado que las personas utilizan las máquinas para su uso personal.

Todavía no hay una definición concreta para el conjunto de conceptos que describen la **interacción hombre-máquina**. En términos generales, podríamos decir que es la disciplina que estudia el intercambio de información entre las personas y los computadores. Ésta se encarga del diseño, evaluación e implementación de los aparatos tecnológicos interactivos, estudiando el mayor número de casos que les pueda llegar a afectar. El objetivo es que el intercambio sea más eficiente: minimizar errores, incrementar la satisfacción, disminuir la frustración y, en definitiva, hacer más productivas las tareas que rodean a las personas y las computadoras.

La Interacción Humano-Computadora o Human Computer Interaction, mejor conocida por sus siglas en inglés HCI, es el estudio de la interacción entre el ser humano, las computadoras y las tareas que se desarrollan; principalmente se enfoca a conocer cómo la gente y las computadoras pueden interactuar para llevar a cabo tareas por medio de sistemas y software.

HCI es una materia que se basa en algunos aspectos relevantes de la teoría y métodos de muchas disciplinas, dentro de las cuales se incluye ciencias físicas y sociales, ingeniería y arte. Existen importantes contribuciones a la HCI que se han tomado de las ciencias de la computación, psicología, matemáticas, artes gráficas, sociología, inteligencia artificial, lingüística, filosofía, antropología y ergonomía.

HCI incluye partes fundamentales de la ergonomía debido a que se preocupa por entender cómo las computadoras y el ser humano pueden interactuar para desarrollar tareas existentes y nuevas. La ergonomía en HCI se enfoca a buscar los aspectos de diseño en los sistemas computacionales para que tengan un uso más efectivo y eficiente, así como el desarrollo de principios, guías, métodos y herramientas para mejorar el diseño y desarrollo de los sistemas interactivos computacionales.

2.7.1 Mecanismos para la interacción

Dado que la interacción persona-computador estudia la comunicación entre el ser humano y las máquinas, esto provoca que se tengan que tener unos conocimientos por parte de ambos, humano y máquina. Desde la perspectiva de las máquinas, hace falta que cuenten con un adecuado sistema operativo, técnicas gráficas, lenguajes de programación y entornos de desarrollo. Por la otra parte, es importante tener unos conocimientos previos, como teoría de la comunicación, disciplinas de diseño de gráficos e industriales, lingüísticos, ciencias sociales, psicología cognitiva y función del ser humano.

Con el fin de tener un concepto más aproximado sobre el campo de la interacción humano-computador contemplamos en que está especializado:

- Unión de las tareas de los humanos con las máquinas.
- Capacidades humanas para utilizar las máquinas (incluyendo la capacidad de entender las interfaces)
- Algoritmos y programas de la interfaz en sí.
- Conceptos de ingeniería que se plantean a la hora de diseñar y construir interfaces.
- El proceso de especificación, diseño, e implementación de la interfaz.
- Sacrificios del diseño.

En conclusión, cuenta con aspectos científicos, de ingeniería y de diseño.

Obviamente el análisis se extiende en el sentido de que no sólo hay una manera de interactuar con estas máquinas, sino que existen diferentes estilos. Preece (1994) dice que: " Estilos de Interacción" es un término genérico que se utiliza para agrupar las diferentes maneras en que los usuarios se comunican o interaccionan con el ordenador o la computadora.

Los estilos de interacción más importantes son:

I.- La interfaz o interfaces por línea de órdenes: Es una manera de dar instrucciones directamente a la computadora. Pueden tener la forma de teclas de función (F1 al F12), un caracter, abreviaciones cortas, palabras enteras o una combinación de las dos primeras.

2.- Menús y formularios: Un menú es un conjunto de opciones visualizadas en la pantalla, que se pueden seleccionar y la selección de una de ellas o más supone la ejecución de una orden subyacente y normalmente un cambio en el estado de la interfaz. A diferencia de la interfaz por línea de órdenes, los usuarios tienen la ventaja de no tener que recordar ni palabras, ni sintaxis, siempre y cuando los textos que acompañan a los menús sean significativos, lo que no siempre se produce. Uno de los problemas que tienen los menús es que ocupan mucho espacio en la interfaz, dicho problema se trata de resolver con los menús desplegables o menús pop–up.

3.- Manipulación directa: El término manipulación directa describe sistemas que tienen las siguientes características:

a.-Representación continua de los objetos y de las acciones de interés.

b.-Cambio de una sintaxis de comandos compleja por la manipulación de objetos y acciones.

c.-Acciones rápidas, incrementales y reversibles que provocan un efecto visible inmediatamente en el objeto seleccionado. La interfazceWIMP: WIMP quiere decir Ventanas, Íconos, Menús y Apuntadores (Windows, Icons, Menus and Pointers)

4.- Interacción asistida: agentes – asistentes: Utiliza la metáfora del asistente personal o agente que colabora con el usuario en el mismo ambiente de trabajo. Así el usuario en vez de dirigir la interacción, trabaja en un entorno cooperativo. Por todo ello, el usuario y los agentes o asistentes se comunican, controlan eventos y realizan tareas.

2.7.2 Agentes de la interfaz (AI)

Según Henry Lieberman (1997) es un programa que puede ser considerado por el usuario como un asistente o programa que le ayuda y no se le considere una herramienta desde el punto de vista de una interfaz de manipulación directa.

El Agente de la interfaz lee la entrada que el usuario presenta y puede hacer cambios en los objetos que el usuario ve en la pantalla, aunque no necesariamente una–a–una con las acciones del usuario. El agente puede observar muchas interacciones del usuario antes de hacer una acción o con una sola interacción puede lanzar una serie de acciones y actuar en determinados períodos de tiempo que le hemos fijado.

2.7.3 Principales componentes

Se considera que hay cuatro componentes principales en un sistema hombre– computadora:

- El usuario
- El sistema de computadora
- La tarea
- El ambiente

La interacción de estos componentes es una de las más importantes partes del sistema: el interfaz hombre-máquina. En un sistema hombre-computadora, con la ayuda de las aplicaciones y software apropiado, el usuario introduce sus órdenes a la computadora, y esta responde al usuario de acuerdo con las funciones para las que fueron diseñadas las órdenes introducidas. En general, a esta interacción se le conoce como interacción hombre-computadora, interacción que tiene gran influencia sobre el humano y el desempeño del sistema en una forma compleja, y ha sido en los últimos tiempos una parte sustancial de la investigación en ergonomía y factores humanos.

La interacción hombre-máquina se refiere a la conexión existente entre el ser humano y las nuevas tecnologías. Esta interacción ayuda a mejorar las posibilidades de las nuevas tecnologías en la enseñanza en dos importantes aspectos:1) Guía a un análisis cuidadoso y sistemático sobre qué información, herramientas y capacidades necesita las personas para conseguir sus objetivos; 2) Proporciona herramientas y técnicas para conseguir dicha interacción.La interacción de hardware, software y usuario es una de las más importantes partes del sistema de comunicación: la interfaz hombre-máquina es un canal comunicativo entre el usuario y la computadora.

El usuario y la computadora comparten responsabilidades para desarrollar las diferentes actividades que contribuyen a alcanzar la meta global, ya que ambos deben interactuar y comunicarse en un diálogo que no sólo involucra a los comandos, sino un conjunto estructurado de requerimientos, preguntas y respuestas. Como en cualquier clase de diálogo, el que se da entre el usuario y la computadora es más exitoso si ambas partes tienen un conocimiento y lenguaje común, lo que implica que el usuario debe tener el conocimiento adecuado y suficiente acerca de la computadora, y el diseñador del sistema de la computadora debe tener un conocimiento adecuado del usuario.

2.7.4 La interfaz como principal mecanismo para la interacción

Según el diccionario de la Real Academia de la Lengua Española, *interfaz* se refiere a una "Conexión física y funcional entre dos aparatos o sistemas independientes". Moisés Mañas (2013) define a la interfaz como "El mecanismo mediador entre el humano y la computadora", "El punto, el área, o la superficie a lo largo de la cual dos cosas de naturaleza distinta convergen". Además aclara que "Cuando existen dos sistemas cualesquiera que se deben comunicar entre ellos la interfaz será el mecanismo, el entorno o la herramienta que hará posible dicha comunicación". Esto implica, además, que existe un sistema de traducción entre el hombre y la maquina, porque ambos manejan códigos diferentes: verbo-icónico en el caso del hombre y binario en el caso del procesador electrónico.

Existe una idea fundamental del concepto de interfaz, que es el de mediación, entre hombre y máquina. Es decir, la interfaz es lo que "media", lo que facilita la comunicación, la interacción entre ambas partes. Hoy en día, para comprender lo que nos rodea, son necesarias las metáforas. Las mismas son instrumentos del conocimiento aditivo que siempre dicen algo más. La interfaz entre el hombre y la máquina carece de conceptos específicos, es por esto que se se ha recurrido para su interpretación, al uso de metáforas explicativas.

La interfaz entre el usuario y la computadora no sólo es lo que el usuario puede ver, oír y tocar; también incluye los conceptos que el usuario necesita conocer acerca del sistema y como puede

ser utilizado para desarrollar las diferentes tareas. De acuerdo con Johnson (1992), el usuario de un sistema de computadora necesita por lo menos:

- Reconocer que el sistema puede ser utilizado para alcanzar una meta particular.
- Identificar los procedimientos necesarios que deben llevarse a cabo con la computadora para alcanzar la meta deseada.
- Conocer los comandos necesarios para que el sistema ejecute las funciones requeridas como parte de su tarea.
- Identificar y entender los diferentes estados del programa.
- Poseer las habilidades necesarias para comunicarse con el sistema (escribir, apuntar o hablar).

El campo de interacción hombre-máquina se concibe con el diseño del interfaz y es altamente interdisciplinar por naturaleza. Esto supone investigaciones desde la Psicología, la Informática, la Ingeniería, la Educación y las Comunicaciones. La interfaz hombre-máquina es un canal comunicativo entre el usuario y la computadora.

Un asunto central de la investigación interacción hombre-máquina es determinar los efectos humanos, tanto psicológicos como cognitivos, y las características afectivas de las interacciones entre los usuarios y las computadoras en tareas específicas. De esta manera, los investigadores de la interacción hombre-máquina desarrollan modelos de actividades humanas y uso de estos modelos en el diseño de nuevos interfaces.

La producción, distribución y administración de la información se han convertido en las actividades principales de los conocimientos modernos en los que se basa la sociedad. De esta manera, la interfaz entre humanos y las nuevas tecnologías será continuamente mejorado en función de realizar el ideal de comunicación humana mundial. Los humanos quieren que expresiones como el diálogo, gestos o escribir a máquina sean entendidos inmediatamente por los ordenadores y otros sistemas de información. El “paradigma de la persona entera” y de “la red hombre-máquina” son los puntos culminantes en el mundo futuro de la comunicación.

2.8 Interfaz Gráfica de Usuario (IGU)

2.8.1 Definición etimológica y aproximación conceptual

Una IGU es una interfaz de usuario en la que una persona interactúa con la información digital a través de un entorno gráfico de simulación. Este sistema de interacción con los datos se denomina WYSIWYG (What you see is what you get, 'lo que ves es lo que obtienes'), y en él, los objetos, íconos (representación visual) de la interfaz gráfica, se comportan como metáforas de la acción y las tareas que el usuario debe realizar (tirar documento = papelera). Estas relaciones también se denominan interfaces objetos-acción (object-action-interface, OAI).

El concepto de interfaz es un concepto amplio que ha sido definido, según el ámbito de conocimientos, desde varios puntos de vista: desde la biología (interfase), ha sido definida como la "capa" de un organismo que separa su interior del exterior, desde la electrónica y las telecomunicaciones, se ha definido como "puerto a través del que se envían o reciben señales desde un sistema o subsistemas hacia otros". En química interfaz es la superficie entre dos fases distintas en una mezcla heterogénea".

Si vamos a la etimología de la palabra "interfaz" encontramos una palabra compuesta, por dos vocablos:

> **Inter** proviene del latín inter, y significa, "entre" o "en medio", y **Faz** proviene del latín facĭes, y significa "superficie, vista o lado de una cosa". Por lo tanto una traducción literal del concepto de interfaz atendiendo a su etimología, podría ser "***superficie, vista, o lado mediador***".

En el contexto de la interacción persona-computadora u hombre-máquina, hablamos de interfaz de usuario, para referirnos de forma genérica al espacio que media la relación de un sujeto y la computadora o sistema interactivo. La interfaz de usuario, es esa "ventana" de un sistema informático, que posibilita a una persona interactuar con ella.

Cuando hablamos de interfaz gráfica de usuario, el concepto es aún más específico en cuanto que *interfaz gráfico* de usuario al contrario que el concepto de "interfaz" tiene una localización determinada y definida: Si la interfaz etimológicamente supone la cara o superficie mediadora, la IGU, supone un tipo específico de interfaz que usa metáforas visuales y signos gráficos como paradigma interactivo entre la persona y la computadora o dispositivo.

El concepto de interfaz gráfico, nos da pistas sobre el modelo de interacción y la tipología de signos que contiene esta superficie mediadora.

Como definición incial, podemos observar cómo interfaz gráfica de usuario, nos remite a conceptos relacionados tales como capa, puerto, superficie o método de interacción. Este tipo de elementos por sí mismos no dejan claro a qué tipo de objeto nos enfrentamos, dónde estaría ubicado, o cual sería su naturaleza.

Por otro lado, la inclusión del concepto “gráfica” dentro de la propia definición de IGU, supone un dato que nos acerca un poco más a su propia naturaleza visual y efectivamente nos hace constatar éste, como un objeto de análisis óptimo de investigación desde la perspectiva de la teoría de la imagen y la gramática visual.

Desde un punto de vista semiótico, en el contexto de un *positivismo contemporáneo*, habrían dos “enfoques” posibles respecto a un objeto de análisis. Estos dos enfoques posibles son el enfoque sintáctico y el enfoque pragmático. Cada uno de estos enfoques centra su atención sobre una de las partes que normalmente interviene en un proceso de semiosis.

El enfoque semio-sintáctico, abstrae, en el análisis, al sujeto con un mensaje (conjunto de signos), se aproxima al objeto de forma “objetiva”, pero ficticia. Este tipo de análisis son interesantes para revelar ciertos aspectos del objeto en una situación idílica, desde la perspectiva de un sujeto-modelo, realizada en el laboratorio y por lo tanto en un contexto artificial.

El enfoque semio-pragmático, en cambio, toma la relación de objeto y sujeto, teniendo en cuenta las variables cognitivas del sujeto en un ambiente natural, teniendo en cuenta el problema desde la realidad mental del mismo. Este enfoque es el que más se puede aproximar a la realidad práctica, aunque no está exento de problemas, ya que según Chamorro (2006), “aún no ha sido formulada una teoría del sujeto” y por lo tanto tampoco la posibilidad de realizar un análisis “científico” desde esta perspectiva.

A la hora de realizar un análisis semio-cognitivo de la interfaz gráfica de usuario, deberíamos abordar el problema desde ámbos enfoques, para poder sacar el mayor número de datos y referencias posibles.

La interfaz gráfica de usuario, desde el lado del objeto (abstrayendo al sujeto que contempla), no es más que el dispositivo de un sistema informático, un área funcional tan importante como pueda ser la carrocería si se tratase de un coche. Un sistema necesita normalmente varios mecanismos para accionar, funcionar, e interrelacionarse con el entorno, desde un punto de vista objetual (sintáctico) la interfaz gráfica de usuario, no es más que una parte del sistema, desde la cual es posible realizar cambios sobre éste. Por lo tanto el análisis sintáctico de la interfaz, nos aleja de la definición “conceptual” y nos acerca a la realidad objetual de la interfaz, como parte física del sistema informático. Desde esta perspectiva, la interfaz gráfica, tiene peso, medidas, localización

física, limitaciones tecnológicas y propiedades, que habría que analizar y describir.

Desde este mismo punto de vista semio-sintáctico, la interfaz es un dispositivo físico, que como tal, exige por parte del usuario, una serie de condicionantes fisiológicas, y supone, el uso de dispositivos que permitan poner en contacto al sujeto con el sistema tecnológico. Estos dispositivos, que serán reseñados más adelante, son los llamados *dispositivos de interfaz humano*, como el ratón o el teclado, dispositivos que permiten a través de las posibilidades físiológicas del sujeto, producir parte de la interacción con la interfaz gráfica de usuario y por lo tanto, parte fundamental de la misma.

Si nos acercamos al problema desde el lado del sujeto (enfoque pragmático), entonces sí podríamos entender de alguna manera la afirmación de que una interfaz gráfica pueda ser un método de interacción con un sistema.

Cuando hablamos del interfaz, hablamos del proceso mediante el cual, un sujeto, se acerca a un sistema tecnológico con el que interacciona a través de los signos inscritos en dicha superficie. El proceso interactivo, requiere de una serie de "requisitos" cognitivos básicos por parte del sujeto, como percibir, decodificar, memorizar, decidir y navegar a través de la interfaz gráfica (Delgado, y otros, 2004). Desde esta perspectiva, la IGU sólo cobraría sentido, en cuanto el sujeto es capaz de "comprender" el significado y el proceso de interacción, y sus facultades cognitivas son capaces de interpretar adecuadamente los signos que se producen sobre el interfaz y usarlas adecuadamente.

Por lo tanto podríamos concluir diciendo que según el punto de vista sobre el objeto de análisis, obtendremos una información u otra. En este caso, los dos posibles desde el punto de vista semiótico, nos dan dos resultados diferentes pero complementarios: por un lado tenemos un área física que pertenece a un sistema informático o interactivo, y por otro lado, tenemos un sujeto limitado por sus capacidades lingüísticas y cognitivas que debe dar respuetas de interpretación y acción sobre el sistema interactivo.

2.8.2 Dimensiones: física y simbólica

Desde el punto de vista semiótico, una imagen proyectada en un soporte, supondría un área simbólica (lenguaje) inscrita dentro de un área física (soporte o medio), siempre y cuando sea observado por un sujeto capaz de interpretar y reconocer los signos que intervienen en dicho espacio. Los signos necesitan cuanto menos, dos condiciones básicas para poder funcionar como tales, por un lado un soporte donde poder manifestarse (un medio o canal a través del cual los signos pueden circular y manifestarse), y por otro lado una persona capaz de interpretar y

dotar de sentido dichos signos. Para ello es necesario la existencia de una superficie física, un medio físico, pero ese área debe ser un área abierta al lenguaje, a la semiosis.

Según la explicación de Uxia Rivas (2012), Entendemos por “semiosis” el proceso mediante el cual un conjunto de signos producen significación en la mente de un sujeto. Sin semiosis no tiene sentido el área física para una persona ya que no produciría ningún tipo de significación. Sin un área física donde representar los signos, resultaría igualmente imposible llevar a cabo la semiosis visual. Por lo tanto podríamos afirmar, que la interfaz gráfica de usuario como área interactiva, la cual pone en contacto un usuario con un sistema informático, también constituye un espacio semiótico que necesita de un espacio físico para poder cumplir con el objeto de la interacción.

Cuando hablamos de “área física” y “área simbólica”, nos referimos de algún modo, de dos dimensiones reconocibles que tiene cualquier artefacto, esto es , su dimensión física de soporte (en un cuadro es el lienzo, en una película es el negativo, etc) y su dimensión simbólica, que es aquella que hace referencia al significado concreto que es interpretado por un sujeto, capaz de percibir, decodificar y entender los signos inscritos en el medio físico. Por lo tanto, hablar de área física y área simbólica, es hablar de dos dimensiones “reales” del mismo objeto, es reconocer las dos dimensiones relacionadas del artefacto, donde necesariamente cobra sentido ante la mirada de un sujeto.

Podríamos afirmar que la IGU es un tipo “ especial” de *artefacto tecnológico* sujeto a los procesos de semiosis, y por lo tanto sujeto a la misma naturaleza que puedan tener un cuadro, una película o una fotografía.

2.8.3 Comunicación e interacción

Aún aceptando que la IGU, al igual que una fotografía, es un artefacto, que dispone como tal, de dimensión física y simbólica, abierta a los procesos semióticos y comunicativos, valdría la pena preguntarse, ¿Es la interfaz gráfica de usuario un espacio de comunicación o de interacción ?

Partiendo del análisis que hace Carlos Marrero (2006), entendemos por ***comunicación***, en el contexto de la comunicación humana, cuando dos o más individuos, son capaces de establecer a través de algún medio, una transmisión de información significativa entre los implicados. De una forma u otra, la comunicación implica compartir unos códigos lingüísiticos, un mismo canal de comunicación, e implica nacesariamente por parte del receptor de la información, la capacidad de interpretar los signos expuestos en el mensaje informativo de modo que resulten significativos.

En lo que respecta a la ***comunicación visual***, el proceso comunicativo, quedaría acotado, allí donde se produce la transmisión de información entre un medio audiovisual (cine, televisión, libro, cartel, móvil) , y un individuo, el cual debe ser capaz de interpretar adecuadamente un conjunto de signos visuales dentro de un contexto, y dotar de sentido a aquello que ve.

Por ***interacción*** entendemos la acción que se ejerce recíprocamente entre dos o más sistemas, en nuestro caso, entre el sistema persona y el sistema informático. Un proceso interactivo supone la capacidad de poder producir cambios y modificaciones sobre ciertas variables de alguno de los sistemas implicados.

La comunicación y la interacción están íntimamente relacionadas, ya que, en el proceso de comunicación siempre existe una cierta interacción entre el usuario y el artefacto: para poder ver la tele (comunicación), hace falta encenderla y elegir un canal (interacción)(Manzini & Enzio, 1996).

Igualmente, para que sea posible la interacción, es necesaria la existencia de algún tipo de comunicación o transmisión de información de un sujeto a otro, o desde un artefacto a un sujeto o viceveresa. Para realizar una acción concreta dentro de un contexto interactivo, por ejemplo, pulsar el botón de encendido, antes debo percibir, interpretar, y por lo tanto conocer (dentro del contexto de la comunicación) el significado del dispositivo que contiene la acción del encendido, para poder accionar adecuadamente sobre el mismo.

La comunicación hace referencia a un aspecto concreto dentro del proceso interactivo, aquél que tiene que ver con la transmisión de la información necesaria para que la interacción se pueda realizar adecuadamente. En algunos casos concretos, algunos medios de comunicación, tienden a minimizar la capacidad interactiva, la interacción es mínima en cuanto no se puede ejercer ningún tipo de acción sobre el medio informativo, modificando aspectos semánticos (narración) o formales del objeto informativo.

Para Manzini y Encio (1996), la concepción de un artefacto interactivo, supone una ampliación concreta del proceso comunicativo: la capacidad de interacción supone para el objeto, la necesidad de incorporar un programa de acciones abierta a la transformación por parte del sujeto.

Aunque el concepto de interacción y el concepto de comunicación, como hemos visto, estén íntimamente unidos y relacionados, quizás deberíamos aclarar, que cuando hablamos de procesos interactivos, ya suponemos que el proceso, incluye necesariamente, procesos de comunicación. En cambio, aunque existe una cierta interacción en los procesos comunicativos, no solemos asignar esta propiedad a los mismos. Cuando hablamos de comunicación no suponemos que deba haber una interacción más allá de una serie de condiciones básicas por la cual entramos en

contacto con el sistema de comunicación.

En el caso de la IGU, es evidente que una de sus particularades como artefacto, es esa dimensión interactiva que introduce como objeto simbólico. Esta cuestión condiciona muchas cosas, quizás más de las que aparentemente percibimos de forma lógica e instintiva, tras aprender a interaccionar de un modo básico con el ordenador. Por lo tanto, desde un punto de vista semio-cognitivo, deberíamos estar atentos, a la relación entre comunicación e interacción que se produce entre objeto y sujeto, y de este modo determinar de qué forma, puede condicionar el contexto de la comunicación interactiva a la propia percepción del sujeto cuando se enfrenta a los signos que sirven de vehículo a los procesos que venimos describiendo.

Una interfaz del usuario no sólo se compone de la representación de los datos de entrada, resultados y de los estados del sistema. En una interfaz de usuario se encuentra también la comunicación, la conversación entre el usuario y el sistema, cuestiones de comportamiento que deben ser expresados y representados de alguna manera.

Una interfaz provee interacción visual cuando el usuario para comunicarse con el sistema y llevar a cabo sus intenciones, utiliza diálogo asincrónico, basado en manipulación directa y en eventos. El usuario puede expresarse seleccionando, señalando, arrastrando, moviendo objetos presentes en la pantalla. Un ejemplo sería un aplicativo de geometría donde el usuario puede mover, arrastrar, dimensionar figuras geométricas.

Por otra parte, la interfaz también debe utilizar mecanismos visuales para expresarse ante el usuario, esto es en el caso de dar indicaciones, aclaraciones, mensajes de error u otro tipo de diálogo que vaya dirigido desde la máquina al usuario. Entonces, una interfaz visual debe utilizar gráficos, colores, movimientos, animaciones, sonido para transmitirle información al usuario del sistema.

El dominio de aplicación donde se encuentra interacción visual está constituido por las interfaces basadas en manipulación directa, interfaces gestuales, interfaces icónicas, video juegos, entre otros.

El diseño icónico de una interfaz del usuario se distingue del diseño visual por la calidad y la semántica expresada a través de los recursos visuales que se emplean en él.

La filosofía icónica parte de proveer una imagen del sistema que concuerde fielmente con la representación mental que el usuario tenga sobre el problema. Apunta fundamentalmente a que el usuario perciba que su mundo real con el representado son compatibles.

Esto significa que se debe trabajar profundamente sobre modelos de usuarios para poder detectar cómo es su mundo, cómo él ve las entidades que maneja, cómo interactúa, de qué manera él trabaja con ellas.

De esta manera, el diseñador podrá ser capaz de construirle así, una simulación o una proyección "casi real" de ese mundo, sobre el espacio de la pantalla.

Como su nombre lo indica, en un diseño icónico, el instrumento visual más importante que se utiliza, tanto para visualizar la información como para expresar el diálogo, es el ícono.

Pero, ¿qué se entiende por el concepto de ícono?. En este contexto, un ícono es mucho más que una pequeña imagen dentro de un botón de una barra de herramientas.

La definiciones más adecuadas de ícono fueron especificadas por Grittins (1986) y Rogers [89], que lo describen como una imagen, una figura o un símbolo que representa un concepto subyacente. También, fue definido por Shi Kuo Chang (1990) como un par (parte física, parte lógica), en donde la parte física sería la imagen del mismo y la parte lógica, su significado. Y finalmente Pierce la expresa como una relación de semejanza, en tanto se parecen al objeto que representan. La relación con aquello a lo que se refieren es directa, por ejemplo: pinturas, retratos, dibujos figurativos, mapas,etc.

A través de esta última definición, se puede inferir que lo que distingue al ícono de una imagen tradicional, es que tiene una semántica o parte lógica asociada que debe estar coherentemente solidificada a su parte física, a lo largo de toda la existencia del mismo. Ahora bien, otra duda que uno se puede cuestionar es ¿dónde se aplica realmente el diseño icónico? ¿solamente en la representación?. La respuesta correcta a estos interrogantes, es no.

Análogamente al diseño visual, el diseño icónico puede estar presente en cualquier parte de la interfaz. Puede ser aplicado en:

La Visualización Icónica:

En este caso, se utiliza al ícono como un medio de representación, para modelar metafóricamente tanto el dominio de la aplicación como los conceptos que se manejan a nivel de interfaz. Se puede visualizar icónicamente todas las componentes y entidades del sistema, sus funcionalidades, datos de entrada, de salida, sus estados posibles.

O también, es el caso del diseño icónico de un sistema groupware sobre conferencias, donde se visualiza claramente el estado de las butacas, si están libres u ocupadas, y el de los participantes, si solicitan o no permiso para hablar.

La Interacción Icónica:

El diseño icónico aplicado en la interacción entre el hombre y la máquina, da lugar a un diálogo asincrónico. Este diálogo puede estar acompañado por la selección de íconos, por una manipulación directa que debe ser significativa, por un feedback semántico, por gestos metafóricos, por una animación representativa, por menús, icónicos, dependencias, mediante el cuál los usuarios llevan a cabo sus tareas en una forma natural y simple.

A través de la visualización e interacción icónica, el usuario interactúa con un ambiente casi real, donde se muestra reflejado y proyectado todo su mundo sobre el espacio de la pantalla. Cuando en una interfaz se utiliza como medio de expresión y representación al ícono, la misma se convierte en un "sistema icónico". Este sistema inicialmente contará con un conjunto de íconos estructurados denominados "íconos primitivos". Estos íconos presentes en la interfaz, pueden ser manipulados o combinados de alguna manera, que traiga como consecuencia la generación de nuevos íconos que intervendrán en el sistema en forma transitoria o permanente.

Por lo tanto, el cardinal del conjunto de íconos presentes en una interfaz puede ser incrementado mediante la creación de nuevos íconos, que surgen de alguna composición o combinación de los íconos miembros de dicho conjunto. Estos íconos generados de otros ya existentes se denominan "íconos complejos" pues expresan un concepto visual más elaborado.

Las combinaciones y relaciones entre íconos con el propósito de generar otros, están regidas mediante determinadas reglas. En dichas reglas intervienen ciertos operadores que, por ser aplicados o por operar sobre los íconos, son llamados "operadores icónicos".

En definitiva, a un sistema icónico formalmente, se lo puede definir como un sistema compuesto de un conjunto de íconos primitivos o conjunto generador, más un conjunto de operadores icónicos.

2.8.4 Signo Visual e interactivo

Llegados hasta este punto, y partiendo de la idea de que la IGU es un artefacto dispuesto en sus dos dimensiones física y simbólica, que participa de los procesos de comunicación, pero inscritos en el proceso de interacción, cabría preguntarse, si los signos que son usados en el medio digital, mantienen diferencias respecto a los mismos signos en un medio conocido como tradicional, por ejemplo la portada impresa de una revista.

Si aceptamos la posibilidad de identificar unidades gráfico-semánticas en la misma interfaz (botones, íconos, menús, barras, signos verbales), podríamos llegar a pensar que los signos que

usa la IGU no son en absoluto diferentes a los que nos podamos encontrar en una autopista. El lenguaje o gramáticas visuales y verbales, son inherentes al sujeto, no al objeto de diseño. El objeto diseñado es adaptado en los procesos de diseño a las condiciones lingüísticas del sujeto y a sus capacidades cognitivas.

Existen algunas diferencias que hacen pensar que los signos inscritos en un medio digital y los signos inscritos en medios tradicionales, pueden y deben ser interpretados desde una perpectiva semiótica y gramatical de la imagen:

> **1.- El contexto del signo:** como hemos venido observando, es determinante en la interpretación por parte de un sujeto. Por poner un ejemplo, un sujeto cualquiera no obtiene la misma significación si observa un signo en el interior de un coche, a cierta velocidad, y atendiendo a ciertos intereses personales concretos como salir de una autopista, que el mismo signo proyectado en la pantalla de la computadora, aún siendo ámbos signos idénticos en su gramática visual. El contexto en el que el signo se encuentra ubicado y es percibido por un sujeto, es determinante en la significación que es capaz de producir en el mismo.
>
> 2- La segunda cuestión tiene que ver con las relaciones funcionales asociadas a cada uno de los signos en el contexto interactivo. Los signos en la interfaz, al contrario que los signos que aparecen en una señal de tráfico, no indican solamente una información que debe ser percibida, decodificada, recordada y cumplida y como concluye Rasmussen, (1983), proceso que podría ser asociado a una elemento en el contexto de la señalización vial. Al contrario, un ícono (pictograma en la interfaz), usa la representación simbólica para indicar en qué lugar se puede realizar un tipo de acción concreta sobre el sistema. Esta acción está dentro del contexto de la interacción, entre el sujeto y el sistema. Por lo tanto la naturaleza del signo, en la mente del sujeto es otra. El signo, una vez interpretado por el sujeto, debe ser asociado a una acción sobre el sistema, lo cual añade y es la hipótesis que sostengo en este punto, una nueva dimensión al signo que no existía hasta la llegada de la interacción gráfica con las computadoras.

Por lo tanto podríamos reconocer un nuevo tipo de signo, el ***signo interactivo***, que lleva de algún modo asociado, en el contexto digital, la dimensión interactiva, la cual supone una relación del signo con la ejecución de una taréa o acción concreta en el sistema (todo esto en la mente del sujeto que debe aprender y usar los signos).

Esta nueva dimensión del signo, no sólo condiciona la propia naturaleza de este tipo de signos en su forma, diseño y contexto, sino además supone nuevos retos para diseñadores, y para el ámbito de la interacción hombre-máquina, ya que algunos signos representados actualmente en

las interfaces gráficas, han tenido usos diferentes hasta ahora, y en el nuevo contexto digital, deben ser percibidos, interpretados y usados con nuevas funcionalidades.

2.8.5 La IGU desde una perspectiva semiótico-cognitiva

Podríamos definir la interfaz gráfica de usuario, en el contexto de la interacción persona-computadora, como un artefacto interactivo, que por su diseño y a través de ciertas interfaces humanas, posibilita la interacción de una persona con el sistema informático, haciendo uso de las gramáticas visuales y verbales (signos gráficos como íconos, botones, menús y verbales, como tipografía).

Como todo artefacto, exige por parte de la persona que interacciona, la capacidades físiológico-cognitivas mínimas, para poder interpretar adecuadamente los signos, y poder realizar acciones efectivas sobre la propia interfaz.

Desde el punto de vista semiótico-sintáctico, la dimensión física del artefacto, implica por parte del sujeto que interacciona, el uso de interfaces humanas, que comuniquen la parte física de la interfaz con la parte simbólica de la misma.

Desde el punto de vista semiótico-pragmático, la dimensión simbólica del artefacto, implica por parte del sujeto que interacciona, el uso y conocimiento de las gramáticas visuales, uso de capacidades para poder realizar codificaciones sígnicas, propias de otros artefactos, más como se ha desarrollado anteriormente, en un nuevo contexto interactivo.

Este nuevo contexto interactivo, supone una dimensión nueva para los signos, que deben ser aprendidos y asociados a funcionalidades concretas, y ser distinguidos de signos análogos que carecen de dicha tipología de funcionalidades en el mismo contexo. Ello exige de algún modo proponer una ***gramática interactiva del lenguaje visual***.

2.8.6 Evolución de las Interfaces Gráficas

La historia de la interfaz gráfica ha estado marcada en su evolución por dos factores decisivos: la investigación y el negocio.

El origen de su nacimiento está en la búsqueda de un método de interacción amigable con las computadoras que superara la interfaz de línea de comandos. La repercusión que ha tenido su descubrimiento en la computación e informática se ha traducido en muchos beneficios para

aquellos individuos y empresas que han se ha aprovechando de esto y han explotando los hallazgos propios y ajenos.

La guerra de los interfaces desde una perspectiva semio-cognitiva, es tan importante como la guerra de sistemas operativos de los sistemas informáticos actuales sobretodo en los últimos años. Poseer la interfaz, es de algún modo, tener una herramienta poderosa de control sobre las personas que la utilizan. Es poder para definir los modelos de interacción, definir los signos que intervendrán y por lo tanto tendrán que ser aprendidos por el "usuario". Esa razón es por la que empresas como Apple o Microsoft, Be o Xerox, han mantenido verdaderas luchas de poder por dominar el territorio de la interfaz demandándose unas a otras.

La interfaz gráfica como artefacto tecnológico, nace en el año 1973 en el centro de investigación Xerox Alto, donde inició con el objetivo básico de encontrar un modelo óptimo de interacción persona-computadora(1er. periodo), pasa por un proceso de eclosión y de madurez donde se definen sus elementos básicos (2do. período), para acabar convirtiéndose en un producto de consumo estético dentro de los sistemas interactivos, donde la interfaz más allá de un medio de interacción óptimo, se transforma en un objeto inteligente abierto a los procesos de "customización"[4] o personalización por parte del usuario (3er. período) y finalmente alcanza su madurez y se reacondiciona con el advenimiento no sólo de múltiples dispositivos, sino de nuevas generaciones de usuarios que "nacen" con ella.

Describiremos a través de un análisis breve el periodo que va desde el año 1970, año de nacimiento del centro de investigación Palo Alto, hasta los primeros años del siglo XXI, que podríamos considerar ya de madurez para la interfaz, porque se introducen en el mercado una serie de interfases de características que podríamos considerar de última generación como son las de Mac OS X, Windows, GENOME, ADROID y KDE.

La historia que se sintentiza a continuación, no recoge todas las interfaces producidas, en cambio intenta trazar una síntesis cronológica, localizando los momentos decisivos y los elementos más importantes en el contexto de las computadoras personales y la informática de consumo.

4 Customizar es un verbo que no forma parte del diccionario de la Real Academia Española (RAE) pero que, sin embargo, tiene un uso bastante frecuente en nuestra lengua. Se trata de una adaptación del término inglés *customize*, que refiere a modificar algo de acuerdo a las preferencias personales. Puede decirse, por lo tanto, que customizar un objeto es lo mismo que **personalizarlo**.

2.8.6.1 Primer periodo: Nacimiento de la IGU (De 1970 a 1981)

El primer periodo de la historia del interfaz está marcado por la investigación y la búsqueda de un paradigma de interacción definitivo y óptimo, que sustituyese la práctica, pero compleja, *interfaz de línea de comandos*[5] .

Ya desde los años cuarenta del siglo pasado, investigadores como Bush (1945) habían trazado de forma teórica, modelos de máquinas y computadoras personales que debían servir para almacenar, editar y compartir información de forma sencilla. No fue sino hasta la llegada de los años setenta, cuando se empezó a trabajar en el desarrollo de dicho modelo interactivo y se formalizarían los primeros modelos de computadoras personales como elementos multimedia, capaces de representar información textual y visual, dando la posibilidad de interactuar con ella de forma amigable.

Uno de estos protagonistas sería el centro de investigación PARC (Palo Alto Reseach Center), una división que la empresa Xerox Corporation constituiría en el año 1970. Allí se desarrollarían las investigaciones más importantes de esta época relacionadas con la computación, culminando en el año 1981 con el desarrollo del la computadora Xerox Star, la cual resumía once intensos años de investigación realizados en el Xerox Parc.

Principales Precursores de la Interfaz gráfica de usuario

Durante esta primera etapa hubo muchos intentos por mejorar las IGUs, los protagonistas más importantes, y sus contribuciones más destacadas para la historia de la interfaz gráfica de usuario son los siguientes:

- **Vannevar Bush:** Desarrolló en el año 1945 su tesis “As We may think” en el que hablaba sobre los medios de almacenamiento del conocimiento e información futuras y proponía la forma en que todo ello debía estar centralizado en un sistema que él llamó MEMEX (Memory Extensión), en el que desarrolla a nivel teórico, el concepto de computadora personal incluyendo además el concepto de hipertexto, como propuesta para un modelo de información interconectada.
- **Theodor Holm Nelson:** introduciría el concepto de hipermedia, virtualidad, hipertexto (1963), y escribiría varias obras de ficción que inspirarían mucho a todos los investigadores del ramo, sobretodo con “Computer Lib and Dream Machines“ publicada en el año 1981.
- **Alan Kay**, matemático y biólogo molecular, fue otro de los investigadores que hizo aportaciones al contexto de la informática y especialmente a la interfaz gráfica de usuario. Sus aportaciones están relacionadas con el lenguaje orientado a objetos, Smaltalk, desarrollado en

5 Interfaz de línea de comandos es un paradigma de interacción basado en texto, en el que se introducen las instrucciones en forma de comandos textuales a través del teclado.

el centro de investigación Xerox Parc, uno de los fundamentos tecnológicos que posibilitaría la posterior implementación de una interfaz gráfica de usuario basada en la representación de íconos. Parte de estas investigaciones darían lugar a la primera interfaz gráfica de usuario. También desarrolló su "*The Dinabook*" una tipo de computadora portátil especialmente diseñada como medio pedagógico para niños.

- **Douglas Engelbart** en sus investigaciones en el Augmentation Research Center, centro que creó y dirigió dentro del Xerox Alto, desarrolló el primer prototipo de ratón y concluiría su tesis titulada *Augmenting Human Intellect: A Conceptual Framework*. Douglas recuperaba aquella idea de Vannevar Bush en la que se desarrollaba el concepto de incremento de las capacidades cognitivas del intelecto humano a través de sistemas almacenamiento de datos en su tesis.
- **Ivan Sutherland,** un ingeniero informático precursor de las comunicaciones y la rama de la informática especializada en la imagen digital. Fue el creador del *Sketchpad*(1963), un software de diseño gráfico pionero tipo vectorial (CAD), que influenciaría el modo alternativo de interacción con las computadoras y por lo tanto, referente ineludible en el nacimiento y desarrollo de las interfases gráficas.

2.8.6.2 Segundo Periodo: Desarrollo de las IGU (De 1981 a 1995)

El segundo periodo de la evolución histórica de las interfases gráficas está relacionado con la revolución de las computadoras personales surgida en el año 1981. Para la interfaz gráfica, este período significa su implementación definitiva, en los hogares, centro de investigación, educación y de negocios.

A partir del año 1981 se produce el despliegue industrial definitivo de la venta de computadoras personales y su implementación definitiva en todos los ámbitos de la infraestructura social y económica de los países desarrollados. En esta década se introduce formalmente la computadora personal o también llamada de escritorio, en el mercado (PC), y su éxito estará en gran parte condicionado por la capacidad de la interfaz gráfica de facilitar la interacción. En este sentido podemos considerar a la interfaz gráfica de usuario un elemento totalmente democratizador puesto que gracias a ella se puede decir que se pudo llevar la informática al grueso de la sociedades industrializadas. Durante este período se terminan de definir y ampliar los modelos hasta hoy vigentes, relativos a la la interacción persona-computadora de la Interfaz gráfica.

A nivel comercial, Apple por una lado, acabaría definiendo el modelo incluido en su MAC OS, y por otra parte, el modelo de Windows quedaría definitivamente desarrollado al final de este

periodo por la empresa Microsoft, ambos inspirados y herederos del modelo de interacción WIMP desarrollado en el Xerox Parc.

El trayecto que va de la entrada en el mercado de la primera Apple con interfaz gráfica de usuario, **Apple Lisa**, a la definición conceptual y tecnológica de las interfases de computadoras con millones de colores en pantalla, está sucedido por un proceso de eclosión de dispositivos informáticos, de computadoras e interfaces gráficas pertenecientes a un periodo complicado donde la lucha más importante consistia en tratar de ocupar un nicho de mercado e imponer una interfaz u otra en cada computadora.

Los principales protagonistas de este periodo serán IBM, Microsoft, Apple y el proyecto de software libre GNU/LINUX. Por el camino se quedan ordenadores, interfases y sistemas operativos de compañias como Commodore, Amiga, Next Computer o Be, las cuales realizarán pequeñas aportaciones fundamentales para entender la evolución de la interfaz gráfica de usuario. Cada una de estas empresas tendrá un papel en la evolución de la interfaz y cada una en su momento hizo aportaciones personales significativas definiendo los paradigmas de interacción actuales disponibles en cualquier dispositivo interactivo.

La gran mayoría de la interfaces propuestas en este período usaron íconos para identificar aplicaciones en el escritorio, y trabajaban con el sistema de ventanas para representar aplicaciones y documentos en el escritorio. Prácticamente todas las computadoras fueron diseñadas teniendo el concepto de WYSIWYG como prioridad en el diseño.

2.8.6.3 Tercer periodo: Automatización y personalización. (De 1995 a 2001)

Cuando los elementos necesarios para interaccionar con la información han sido debidamente desarrollados en sus aspectos funcionales básicos y se ha llegado a un modelo óptimo de interacción, la interfaz gráfica entra en un proceso de desarrollo centrado en la estética, propensa a añadir pequeñas funcionalidades del “detalle”, normalmente provocadas por necesidades mercantiles del producto frente a la competencia y no en relación a las necesidades reales del producto respecto al usuario, tal como lo refiere Stephenson (2003).

Existen varios elementos que identifican bien este periodo de la interfaz gráfica de usuario:

El primero de los cambios significativos que hacen pensar que la interfaz ha entrado en un nuevo período, tiene que ver con su transformación en ***superficie inteligente***. Los procesos de inteligencia añadidos a la interfaz, han convertido a ésta en un *autómata inteligente*, capaz de tomar decisiones propias sobre su propia forma, en lo que respecta al modo de estructurar

y organizar elementos en la misma interfaz. Por lo tanto, ya no podemos considerar la interfaz gráfica como un mero artefacto interactivo. La interfaz ha sido dotada de inteligencia artificial, muy rudimentaria y por lo tanto ha sido transformada en superficie inteligente capaz de ayudarnos a tomar decisiones.

El segundo cambio significativo del interfaz es su transformación de objeto de uso, a objeto de consumo estético a través de los procesos de ***customización o personalización***.

Se trata de un proceso mediante el cual, el usuario varía ciertos aspectos de la apariencia de un objeto, de modo que acerca el nivel simbólico del objeto a sus propios gustos estéticos. Es a través de este proceso donde la interfaz se convierte en un objeto de consumo estético y da un salto cualitativo, desde el estado de necesidad básica representada por la interacción amigable con el usuario, a un estado de producto fetiche abierto a los procesos de personalización y por lo tanto abierto al consumo del producto como objeto en sí mismo. Las acciones más habituales consisten en poner imágenes de fondo en el escritorio, cambiar el conjunto de íconos, cambiar el orden de los elementos en un menú, etc.

La customización, desde un punto de vista afectivo, supone un elemento importante en la interacción persona-computadora. Customizar visto desde un punto de vista semio-cognito, supone adaptar y sustituir parte de los signos incluidos por defecto en el interfaz, por signos más próximos y por lo tanto con cualidades para producir mejor significación por parte del usuario, y esta afirmación hecha por Norman (2005), evidentemente tiene una repercusión en el nivel visceral del usuario.

Aunque los procesos de customización no respondan a necesidades funcionales importantes para el sistema, sí responde como hemos visto a elementos que pueden ayudar a hacer más “amigable” o confortable el objeto interactivo, y por lo tanto, hacer más agradable la interacción con el mismo.

2.8.6.4 Cuarto Periodo: Interfaz de enfoque del usuario (A partir de 2001)

Los tipos de IGUs que se encuentran en juegos de computadora, y las IGUs avanzadas basadas en realidad virtual, se usan con frecuencia en tareas de investigación. Muchos grupos de investigación en Norteamérica y Europa están trabajando actualmente en la interfaz de enfoque del usuario o ZUI (*Zooming User Interface*), que es un adelanto lógico de las IGU, mezclando 3D con 2D. Podría expresarse como “2 dimensiones y media en objetos vectoriales de una

dimensión”.

En informática, una Interfaz de enfoque del usuario o una interfaz de usuario con acercamiento/alejamiento (ZUI en inglés, que significa Zooming user interface) es un entorno gráfico, donde los usuarios pueden ajustar el tamaño del área de visión para ver con más o menos detalle. Un ZUI es un tipo de interfaz gráfica de usuario. Los elementos de información aparecen directamente en un escritorio virtual infinito (generalmente creado con gráficos vectoriales), en lugar de ventanas. Los usuarios pueden desplazar la visión por la superficie virtual en dos dimensiones y acercar / alejar la visión sobre los objetos de interés. Por ejemplo, cuando uno se acerca a un objeto de texto puede ser representado como un pequeño punto, a continuación, una miniatura de una página de texto, a continuación, una página a tamaño completo y, finalmente, una vista ampliada de la página.

Reflexiones en torno a la evolución de la interfaz gráfica.

Revisando la historia hemos podido comprobar que la interfaz como artefacto, está inscrita en los ciclos de vida de cualquier producto industrial. Elementos fundamentales que la configuran, como pueda ser la barra de tareas, o el menú, tiene una historia propia que contar y corresponde a muchas investigaciones y pruebas de diseño, el haber llegado a su actual definición conceptual, formal y funcional.

Adicionalmente poodemos afirmar que las interfaces gráficas se convirtieron, de un artefacto tecnológico con propiedades interactivas que posibilita la interacción con la computadora, a constituirse como artefacto inteligente capaz de orientar al usuario y provocarse cambios a sí misma, en relación a los datos tomados de éste. Esto abre un nuevo ámbito de investigación dentro del contexto de las interfaces que tienen que ver con la adición de inteligencia y que es parte de las orientaciones del presente trabajo de tesis.

Las IGU están actualmente abiertas a los procesos de customización, permitiendo que el usuario modifique aspectos visuales de ellas de modo que la puedan adaptar a sus gustos. Este proceso la convierte de algún modo en un objeto con identidad propia, maleable y dispuesto para el consumo estético. Estas propiedades acercan el interfaz a objetos interactivos de ocio, como juguetes con los que podemos interaccionar y a través de los que poder accede a ciertas informaciones.

También con todo esto se inicia una reflexión en torno a la doble naturaleza que envuelve a la interfaz, su dimensión mediadora de los procesos de interacción y la dimensión mercantil, sometida al mercado como objeto de consumo, lo que nos hace pensar que los procesos de customización y automatización inteligente, inminentemente son los ingredientes fundamentales que de aquí en adelante tendrán las interfaces como principales motivadores para la adhesión de

los usuarios.

2.8.7 Elementos interactivos de la IGU

2.8.7.1 Dispositivos de Interfaz Humana

Los dispositivos de interfaz humana son los diseñados para conectar alguna parte del cuerpo del ser humano con la interfaz gráfica de modo que puedan ser introducidos datos en el sistema. Normalmente son dispositivos que permiten introducir directamente, y en tiempo real, información de “orientación” y “acción” a la computadora sincronizando simultáneamente con una interfaz gráfica.

Los dispositivos de interfaz humana como el ratón, pueden representar en la interfaz gráfica gestos físicos y movimientos, como “apuntar”, “pulsar”, “arrastrar”, “trasladar”, “mover” de forma metafórica que de otro modo sería muy complejo simular.

La interfaz humana forma actualmente, una parte indisoluble respecto a la interfaz gráfica de usuario. Son partes interconectadas de un mismo paradigma de interacción, donde se necesitan uno al otro indispensablemente para que la interacción con el sistema se realice adecuadamente.

Existen diferentes tipos de interfaces humanas los cuales han sido desarrollados paralelamente a lo largo de la historia de la interfaz gráfica. Los más importantes han sido, el teclado, el ratón de ordenador, el trackball (bola), el cursor táctil (touch pad), la tableta gráfica y el Joystick.

Cada uno de estos dispositivos sirven para introducir un tipo de información específica en el sistema a través de la interfaz gráfica.

2.8.7.2 Ventanas

Las ventanas son recursos interactivos usadas para la visualización, jerarquización y navegación de la información en una IGU. A través de las ventanas, pueden ser visualizados un conjunto de documentos, aplicaciones e íconos, sobre los cuales es posible realizar diversas acciones.

Las ventanas permiten una forma relativamente fácil de interacción con la información. Su comportamiento es como el de un objeto, y pueden ser abiertas, cerradas, movidas, escaladas, ampliadas (zoom) y navegadas (scrolling).

Las ventanas fueron uno de los primeros recurso interactivos desarrollado en el contexto de la interfaz WIMP en el PARC. Constituye un marco, a través del cual es posible visualizar y manipular información del sistema.

Han sido definidas dos tipos generales de ventanas: *las ventanas de aplicación y las ventanas de ficheros*. Ambas atienden a una diferenciación semántica, esto es relacionado con el contenido de la ventana y no de la ventana en sí misma.

1. Las ventanas de aplicación son aquellas que surgen para representar las variables de una aplicación concreta en el sistema. En el paradigma WIMP cada una de las aplicaciones se representan en un espacio delimitado por una ventana. Es una forma de establecer niveles jerárquicos dentro de la interfaz y de posibilitar la representación y manipulación independiente de aplicaciones.

2. Las ventanas de ficheros son normalmente usadas por los gestores de archivos en el sistema y sirven para visualizar un conjunto de documentos, aplicaciones, íconos posibilitando diversas acciones sobre estos elementos.

Anatomía general de una ventana

Las ventanas se han ido definiendo a lo largo del tiempo, de modo que al día de hoy, podríamos hablar de una antomía de las ventanas, formada principalmente por cinco elementos básicos: Marco, cabecera de ventana, area de contenido, barra de scroll, y pie de ventana.

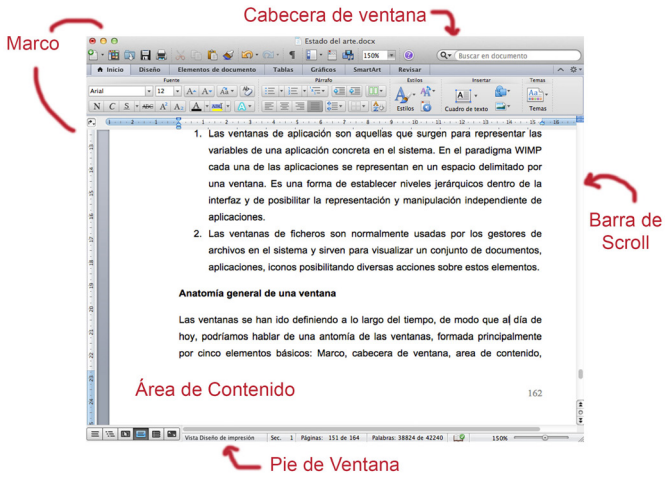


Figura 2.6: Anatomía general de una ventana

El **marco** lo forman el conjunto de recursos gráficos que ayudan a marcar el límite visual entre la ventana y el resto de la interfaz. Normalmente sobre ciertas partes del marco se posibilitan acciones de redimensionamiento de la ventana por parte del usuario. Ha variado estilísticamente y está sujeto a las modificaciones de customización por parte del usuario.

La **cabecera de ventana** es un área dispuesta de forma horizontal que sirve para posicionar los íconos que representan y ejecutan acciones generales sobre el comportamiento de la ventana. Actualmente han sido estandarizadas tres acciones básicas: maximizar, minimizar y cerrar. Hay una cuarta aún no estandarizada pero que es usada por algunos softwares, y consiste en ocultar la ventana, sin cerrar la aplicación, pero sin ser minimizada en la barra de tareas.

El **espacio de contenido** está sujeto al tamaño de la ventana normalmente. Hay contenidos que se adaptan al contenido, y si el contenido de la información representada supera el tamaño de la ventana, entonces la ventana muestra la barra de scroll, que servirá para movernos por el contenido.

La **barra de scroll** de las ventanas ha tenido varios posicionamientos a lo largo de la historia, pero actualmente los sistemas operativos más importantes la localizan en la vertical derecha, para mover el contenido en dirección vertical, y en lado horizontal inferior para posicionar el menú de scroll horizontal. Normalmente está formado por un conjunto de cuatro elementos. Dos flechas-botones situadas a los dos lados de la barra, y un elemento deslizador el cual puede ser arrastrado y actualmente muy usado en navegadores web.

El **pie de ventana** es usado para visualizar información básica de la aplicación o del contenido de esa ventana.

Tipos de ventanas

Podemos hacer una catalogación de ciertos tipos de ventanas, desde un punto de vista semántico, esto es, atendiendo al tipo de significados asociados a la ventana:

a) Ventana Modal

Las ventanas modales son aquellas ventanas específicas que han sido diseñadas como medio de prevención de alguna acción del usuario sobre el sistema. Surgen allí donde el sistema prevé algún error por parte del usuario. Suelen contener un mensaje de advertencia y un botón de confirmación de la acción a realizar.

b)Ventana de Confirmación

Las ventanas de confirmación son aquellas que sugen de modo preventivo igualmente, pero esta vez dan al usuario una serie de posibilidades de acción. La ventana de confirmación más habitual es aquella que surge cuando intentamos guardar un documento que ya tenemos creado en el sistema. Suelen contener el mensaje de advertencia, un ícono indicando la gravedad del asunto, y una serie de botones donde se le pide al usuario una decisión al respecto.

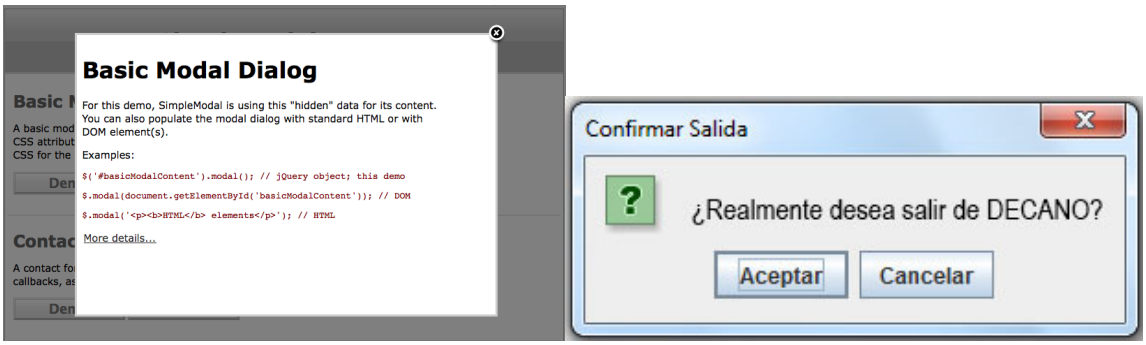


Figura 2.7: A la izquierda ejemplo de Ventana Modal y a la derecha ejemplo de ventana de confirmación

2.8.7.3 Menús

Los menús son listas de comandos, atributos, o cualquier tipo de elementos, agrupados de forma estructurada normalmente inscritos dentro de una barra de menús o de un área específica en la interfaz, los cuales pueden ser activados y posibilitan la ejecución de los elementos, comandos o items que contienen, obteniendo una respuesta inmediata al respecto.

Los items del menú normalmente constituyen descripciones textuales, aunque también incluye en ocasiones signos adicionales que dan información sobre la posibilidad de ser ejecutado (apagado-encendido), el estado del item (activado - desactivado) o el tipo o clase a la que pertenece siendo acompañada de un ícono.

Normalmente los menús sintentizan una estructura de elementos de forma jerárquica por niveles, representados de modo que se muestra una lista, tanto de forma horizonal como vertical de los elementos de un menú, y a continuación, se accede a cada uno de los sub-elementos de cada elemento del menú.

Estados de un Menú

Los ítems de un menú suelen tener estados, es decir, posibles comportamientos que dispone el menú en relación a la interacción por parte del usuario. Los estados habituales de un item de un menú suelen ser:

Activo: Es el estado normal de un ítem sin que el usuario interactúe con él, aunque debe mostrar claramente la posibilidad de poder hacerlo.

Inactivo: El ítem muestra una apariencia difusa indicando que no puede ser seleccionado por el usuario en ese momento, pero que puede hacerlo bajo otras condiciones de

selección de parámetros en el sistema.

Seleccionado (rollover): Cuando el usuario ha posicionado o elegido esa opción, sin haberlo activado, el ítem suele cambiar de estado, normalmente indicando que está seleccionado en ese momento.

Activado: El estado corresponde al momento en que el ítem seleccionado es activado, normalmente ocurre cuando el usuario indica que ha seleccionado cliqueando sobre él o pulsando una tecla que haga esta indicación al sistema (intro).

Pulsado: Cuando el ítem ha sido activado con posterioridad. En una página web suele mostrar un color diferente para indicar que el ítem ha sido pulsado con anterioridad.

Tipos de Menús

Existen varias tipologías de menús, según el contexto en el que se ubiquen, conteniendo una gramática particular y un objeto concreto dentro de la interacción con el usuario. Se comentan a continuación los tipos de menús más habituales en la interacción con las computadoras:

a) Menús contextuales

Menú contextual, es aquel que muestra una lista de ítems posible de ejecutar sobre un objeto concreto en el contexto definido. Un objeto o ícono en el interfaz, puede tener varios estados y diferentes contextos. Los menús contextuales normalmente están ocultos, y son activados por el usuario sobre un objeto en concreto. Éste muestra las opciones que se pueden aplicar sobre el objeto en ese preciso momento. En sistemas Windows y Macintosh son activados con el botón derecho del ratón o con atajos vía teclado o combinaciones de teclas, y son representados de forma flotante al lado del objeto interrogado.

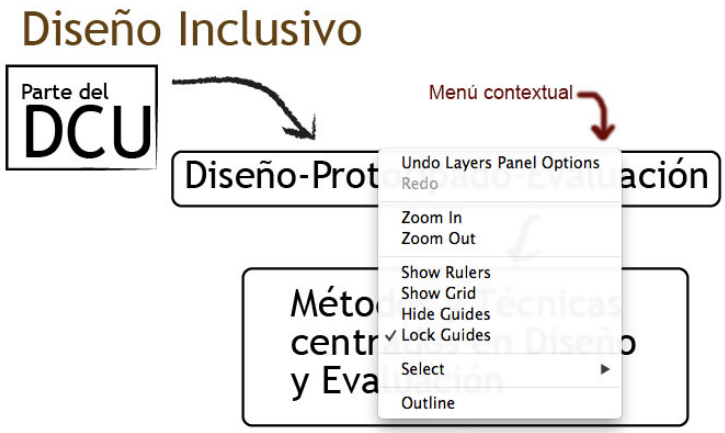


Figura 2.8: Ejemplo de Menú contextual que se activa sobre un objeto.

b) Menús de navegación *(scroll)*

Un menú de s*croll*, es un menú que combina en su interior la posibilidad de realizar movimientos de navegación sobre sus ítems. Cuando las opciones de un menú son demasiadas para mostrar de una sóla vez en la interfaz, se usan diversos recursos de navegación con los ítems, uno de ellos es posicionar un barra en el interior del menú de modo que se pueda navegar usando la barra adecuada. No son muy habitales estos menús, Apple los usó en su sistema operativo y Windows lo usa en su menú de inicio cuando las aplicaciones instaladas son demasiadas para mostrar.

c) Menús jarárquicos

Un menú Jerárquico es un menú representado en forma de árbol, cuyos ítems de un mismo nivel, abren un nuevo menú con nuevas opciones correspondientes a un siguiente nivel. Son usados con frecuencia para sintetizar un árbol amplio de ítems, sin perder la jerarquía de su organización. Este menú es el usado por Windows en su menú de inicio.

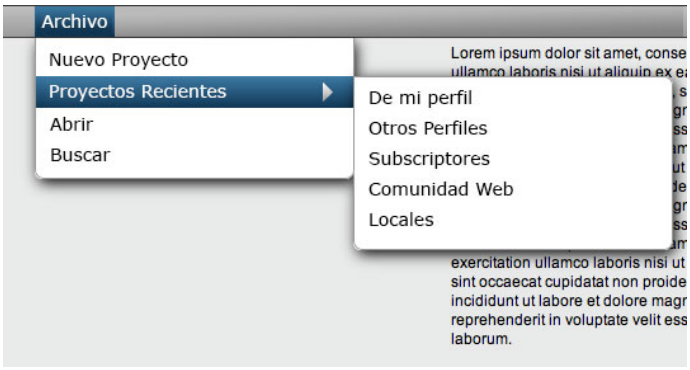


Figura 2.9: Ejemplo de Menú jerárquico.

d) Menú de inicio

Un menú de inicio es un tipo de menú jerárquico desarrollado inicialmente por Microsoft para Windows y actualmente implementado en las interfases de los sistemas operativos GNU/ LINUX. Es un menú jerárquico que intenta recoger un acceso global a todas las variables y elementos y aplicaciones del sistema.

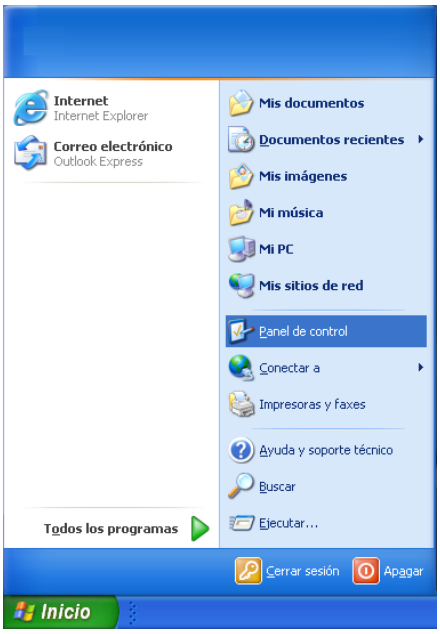


Figura 2.10: Menú Inicio implementado por Windows

Usabilidad en menús

Actualmente, y debido a la repercusión tan fuerte que ha tenido el diseño de interfases para páginas web, el elemento menú textual se ha convertido en uno de los elementos más usado para interaccionar con la documentación en línea. La usabilidad de los menús ha sido estudiada por psicólogos e ingenieros y se ha podido comprobar que los menús entrañan algunos problemas que vale la pena mencionar:

1. Por un lado los menús tienen problemas en relación a la visualización de la información ya que normalmente sólo es posible ver el primer nivel de la jerarquía del menú, pero no el resto de los ítems.

2. Una vez se ha accedido a sus opciones es necesario poder memorizarlas, ya que no están siempre visibles, por lo tanto ofrece problemas a gente que dispone de capacidades psicomotrices afectadas, como las personas llegadas a una cierta edad.

Estas son algunas cuestiones básicas que invitan a diseñar menús de forma visible atendiendo a la mejor usabilidad por parte del usuario como lo dice García (2005).

2.8.7.4 Íconos

Los íconos en el contexto de las interfaces gráficas son signos esquemáticos que representan algún tipo de fichero, carpeta, aplicación, o dispositivos de un sistema informático. Los íconos, como ya se ha mencionado, son signos interactivos y por lo tanto inscritos en una gramática especial que debe ser aprendida por el usuario y se constituyen como parte esencial de los mecanismos de interacción de cualquier interfaz que debe ser diseñados cuidadosamente.

Los íconos usados en la interfaz, provienen principalmente de la representación metafórica planteada en el PARC y cuya misón era que quedaran inscritos dentro de la metáfora del escritorio. A su vez éstos se inspiran en los signos desarrollados en la comunicación gráfica de las señales viales y demás signos codificados por la cultura occidental hasta hoy.

Los íconos son fundamentales para la interpretación y memorización y uno de los elementos fundamentales en el desarrollo de las interfaces gráficas por varias razones:

• Las personas reconocen íconos e imágenes más rápido de lo que tardarían en comprender el mismo concepto a través de la representación verbal. A ciertas distancias pueden ser mejor reconocidos que signos textuales.

• Los íconos cruzan la barrera de la cultura de mejor modo que el lenguaje verbal. Existen algunos signos que tienen reconocimiento internacional.

• Los íconos son capaces de trasmitir conceptos en menos espacio que en lo que lo describiría una palabra a través del lenguaje verbal.

• El ícono como imagen, tiene la capacidad de trasmitir información espacial, relacional, multivariable y representar objetos del mundo real.

Figura 2.11: Ejemplos de íconos

2.8.7.5 Tipografía Digital

Un elemento no menos importante en la interacción con las computadoras, a través de las interfases gráficas son los signos textuales. La tipografía digital podría constituir un trabajo de investigación en sí y constituye un campo relativamente reciente aún no excesivamente investigado.

Las empresas que precisamente se lanzaron a investigar cuestiones relacionadas con la legibilidad en la pantalla, fueron Microsoft y Apple. Esta última desarrolló las primeras fuentes exclusivas para la pantalla, de modo que fueran legibles en el entorno digital de su sistema operativo. Microsoft se ha preocupado por desarrollar su propio sistema de fuentes y ha encargado el diseño de ciertas fuentes específicas para la pantalla, también con la intención de mejorar la visualización de texto en sus sistemas operativos. Encargó al diseñador Matthew Carter una fuente tipográfica especialmente adaptada para pantalla, la fuente Verdana (1994). Uno de los tipos más usados actualmente en la world wide web.

Uno de los principales problemas para la tipografía digital es la legibilidad en pantalla, ya que en este medio tiene una serie de limitaciones y particularidades que la afectan:

• Por un lado la pantalla tiene límites de representación tecnológicos que afectan a la apreciación de los signos textuales. Esto hace que en la pantalla, los signos de palo seco ofrezcan normalmente más limpieza visual que los tipos con remate, por lo que la preferencia se decanta por tipos de palo seco.

• Por otro lado existen problemas relacionados con la naturaleza del signo: el tipo en la pantalla está compuesto como signo luz, frente al tipo en papel constituido como signo materia. La legibilidad del tipo luz ofrece mayor dificultad sobre el ojo, ralentiza la lectura y dificulta la compresión.

Todos estos elementos deberían ser tomados en cuenta por una persona que diseñe para el medio digital.

Uno de los recursos textuales más trascendentes en la actualidad en sistemas digitales y especialmente para los sistemas de información en línea, es el hipertexto.

El ***hipertexto*** es un tipo especial de texto, que contiene propiedades interactivas en el contexto de los sistemas digitales, con un funcionamiento muy similar al de un botón. Dispone también de su propia gramática. Su gramática por defecto corresponde a la adición de texto azul y subrayado sobre el signo textual para indicar que puede ser accionado sobre el mismo. Dispone de estados como cualquier otro recurso interactivo.

La *hipertextualidad* supone el recurso interactivo de enorme trascendencia que está afectando e influyendo sobre las gramáticas desarrolladas en las interfaces gráficas de los sistemas operativos.

2.8.7.6 Controles

Botones

Un botón es un objeto de control sobre la interfaz que posibilita introducir un dato de confirmación al sistema. Actúa como metáfora visual y funcional de los botones incluidos en los dispositivos tecnológicos. Su gramática visual tiene ya un recorrido histórico con posibilidad de ser estudiada. Los botones se utilizan para generar acciones que te llevarán a un resultado inmediato. Un botón es, para entendernos, la evolución del texto enlazado, y su aspecto debe ayudar a que de la sensación de que es un objeto “presionable”, ya sea mediante volumen, sombras o elementos

atractivos que hagan que el usuario identifique su función.

Han sido catalogados varios tipos de botones en relación a sus formas:

a) Botón en Relieve

Es el más común y el más usado en los sistemas operativos. Imita la gramática visual de un botón de un dispositivo físico, por lo que se suele usar un tratamiento cuidado de los bordes, de modo que simule volumen. Suele incluir una descripción breve en el interior, y suele sorportar diversos estados al igual que el comportamiento de las ventanas.

Figura 2.12: Ejemplo de botones en relieve

a) Botones en forma de radio

Son botones redondos que posibilitan ser señalados a través de la acción del usuario. Normalmente son usados en formularios o menús, para dar elección a elegir un ítem de una lista. El interfaz de Mac lo usó con frecuencia en su sistema operativo.

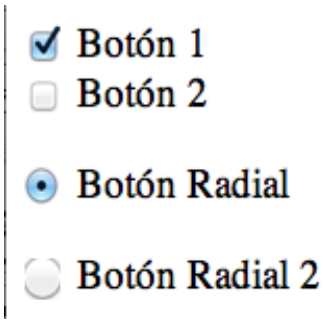


Figura 2.13: Ejemplo de botones radiales y de confirmación.

b) Botones de confirmación *(checkbox)*

Son botones similares a los botones de radio, pero con forma cuadrada. Se representan de forma hueca, y suelen ser usados para seleccionar ítems en una lista.

Elementos de entrada de texto

Los elementos de entrada de texto, nos indican en qué lugar del interfaz puede ser usado el teclado. Cuando todo el interfaz se convierte en escritorio, surgen las aplicaciones específicas que permiten introducir texto. Pero existen partes de ciertas aplicaciones que requieren un área que posibilite la introducción de información textual por parte del usuario. En este contexto es en el que los campos de texto cobran sentido.

Campo de texto

El campo de texto ha desarrollado también su propia gramática visual. Normalmente delimita un área en blanco, e indica a través del borde la posibilidad de introducir texto en la misma.

2.8.7.7 Elementos de Información de Salida

Los elementos de salida, tienen que ver con elementos que se han ido configurando para dar información de estado del sistema al usuario en un momento dado. Normalmente las aplicaciones reservan un área de la ventana, donde posicionan estos datos. Existen varios elementos de información de salida, que vale la pena mencionar:

a) Barra de progreso

La barra de progreso es un elemento que indica al usuario el progreso de la acción que realiza el sistema. Todas las acciones del sistema, no son realizadas de forma instantánea. Cuando el sistema requiere tiempo para realizar una acción, es fundamental dar feedback al usuario a través de la representación del proceso y por lo tanto del progreso de la acción.

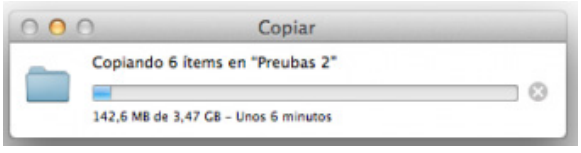


Figura 2.14: Barra de progreso

b) Cuadro de consejo [tip box]

Es un recurso gráfico inspirado en los bocadillos de los cómics, que surge en ciertos elementos de la interfaz para indicar información adicional sobre algún elemento u acción del usuario sobre el sistema.

Figura 2.15: Ejemplo de Cuadro de Consejo o Tip box

c) Barra de estado [Status Bar]

La barra de estado ofrece información variada al usuario sobre diferentes variables de la aplicación o del sistema. Normalmente es posicionada en la parte inferior de la ventana de aplicación. Suele estar divida en varias áreas de modo que en una misma horizontal se muestran varios campos con diferentes informaciones. Suelen ofrecer información técnica específica, muy útil cuando el usuario la necesita.

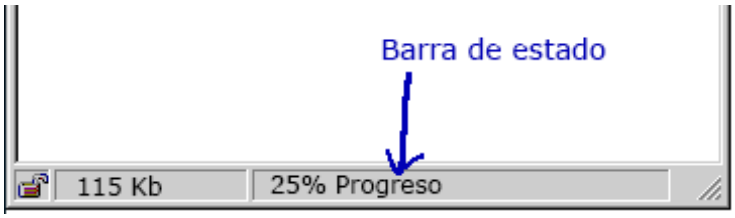


Figura 2.16: Barra de estado

2.8.7.8 Elementos compuestos

a) Barra de tareas

La barra de tareas es un elemento bien definido en sistemas operativos Windows, que posteriormente han sido implementados en sistemas Unix a través de sus respectivos entornos gráficos. Consisten en una barra dispuesta de forma horizontal, en la que se posicionan diversos elementos interactivos, normalmente íconos, que activan aplicaciones y sirve además para ir

añadiendo y alojando aplicaciones útiles para el usuario. Suelen estar dividido cuanto menos en cuatro partes:

• **Botón de Inicio**: Sirve para activar el menú de inicio y poder acceder a sus funciones.

• **Área de aplicaciones más usadas**: Muestra de forma sintética íconos de las aplicaciones más usadas en el sistema como puedan ser el escritorio y el navegador de internet o navegador de archivos.

• **Área de descanso**: En un principio desocupada, es la parte de la barra de tareas destinada a disponer los elementos minimizados cuando el usuario ejecuta más de una tarea en el sistema.

• **Área de aplicaciones del sistema:** Muestra de forma sintética, a través de íconos, diferentes aplicaciones relacionadas con cuestiones técnicas del sistema que operan en el momento de su ejecución.

Figura 2.17: Ejemplo de Barra de tareas.

b) Combo de texto (combo box)

El combo de texto, es un elemento formado en un estado inicial por un campo de texto y una pestaña. El usuario puede introducir texto sobre el campo, pero si pulsa la pestaña despliega una ventana completa con elementos de navegación incluida. Es un elemento combinado que dispone de varias posibilidades de interacción y de acceso a la información introducida.

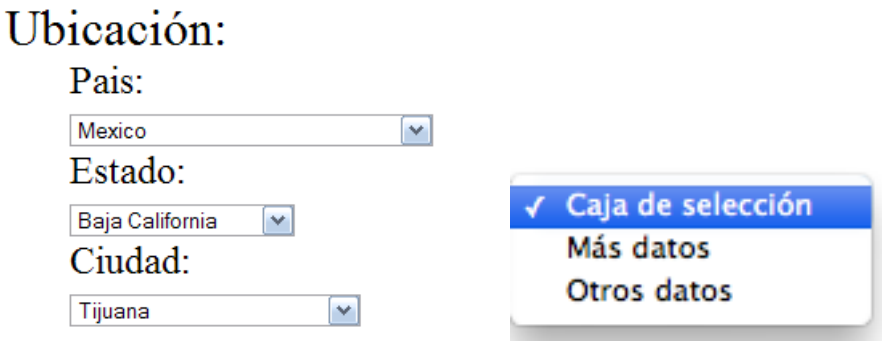


Figura 2.18: Ejemplos de combo de texto o text box

2.8.8 Principios, Directrices, Estándares y Normas

2.8.8.1 Principios y Directrices

A pesar de que existen modelos para la construcción de las interfases ya probados, mucho de ellos finalmente son una síntesis de principios y normas para mejorar la presentación de la interfaz en un contexto de interacción hombre-computadora, es decir se basan en disciplinas como la Ergonomía y las Ciencias congnitivas y promueven el diseño centrado en los usuarios.

Hablando de los principios de construcción de la interfaz, podemos decir que se trata de objetivos generales que pueden ser útiles para organizar el diseño. Sin embargo, no se especifican métodos para obtener esos objetivos, y está limitado al uso práctico para minimizar el trabajo del usuario

a. **Principios Preece (1994).** Están basados en principios de alto nivel y de una aplicación muy general

- Estudiar la población de usuarios
- Reducir la carga cognitiva
- Aplicar técnicas de ingeniería para resolver la problemática del error humano
- Mantener consistencia y claridad

b. Principios Simpson (1985), quien desarrolla una serie de principios básicos basados en la necesidad de crear interfases usables.

- Definir a los usuarios
- Dejar el control a los usuarios
- Minimizar el trabajo de los usuarios
- Realizar un programa sencillo
- Es preciso ser consistente
- Son necesarias las realimentaciones
- Procuarar no cargar la memoria de trabajo
- Tratar de no hacer un uso abusivo de la memoria a largo plazo

c. Principios Schneiderman (1992)

- Consistencia
- Permitir a los usuarios experimentados el uso de atajos
- Dar información de realimentación
- Hacer la gestión de errores sencilla
- Permitir que se puedan deshacer acciones
- Reducir la carga cognitiva de la memoria a corto termino

d. Principios de Usabilidad de Jakob Nielen (1995)

- **Principio 1: Visibilidad del estado del sistema** (Visibility of system status). El sistema siempre debe mantener a los usuarios informados sobre lo que está pasando, a través de información adecuada en un plazo razonable. Ejemplos prácticos:
    - Las barras de proceso o cargando que nos indican cómo avanza un proceso deben ser descriptivos y evidenciar el estado, actividad y tiempo. Por ejemplo: la carga de un slider, la carga de una imagen de perfil, el envío de un formulario, etc.

    - Los mensajes de confirmación de los procesos que le indica al usuario la finalización o validación de este, por ejemplo el mensaje de confirmación de envío de un formulario, una encuesta, etc.

Principio 2: Consistencia entre el sistema y el mundo real (Match between system and the real world). El sistema debe hablar el idioma de los usuarios, con palabras, frases y conceptos familiares para el usuario, en lugar de términos orientados sistema. Siga las convenciones del mundo real, haciendo que la información aparezca en un orden natural y lógico. Ejemplos prácticos:

- En el mundo real el color rojo detiene, el amarillo advierte, el verde valida. No se puede pretender usar un botón de aceptar rojo y esperar que el usuario no lo rechace, consciente o inconscientemente.
- Los usuarios deben interactuar bajo códigos de comunicación y de conducta iguales a la realidad. Los errores de validación por ejemplo, también deben ser rojos.

Principio 3: El usuario es libre y tiene el control (User control and freedom). Los usuarios a menudo eligen funciones del sistema por error y necesitarán un marcado claramente como "salida de emergencia" para salir del estado no deseado sin tener que pasar por un diálogo extendido. El usuario podrá deshacer y rehacer. Ejemplos prácticos:

- Los sliders y carruseles deben poder ser controladas por los usuarios. Incluir botones de "start" y "stop" además de los tradicionales sistemas de navegación como "bullets o viñetas" y "flechas". Los elementos multimedia como galerías, videos y audios no deben reproducirse automáticamente. El usuario debe tener el control total de la página.
- El usuario debe saber siempre donde se encuentra y también cómo regresar. Por lo cual se debe proyectar siempre un sistema de navegación interno, ya sea "miga de pan", botones de anterior o siguiente, y/o paginación, elemntos de énfasis, etc. Así el usuario podrá siempre sabrá donde está ubicado.

Principio 4: Consistencia y estándares (Consistency and standards). Los usuarios no deberían tener que preguntarse si diferentes palabras, situaciones o acciones significan lo mismo. Siga las convenciones de la plataforma. Ejemplos prácticos:

- Los enlaces de ampliar información de las noticias o destacados siempre deberán decir lo mismo, sin abreviaciones. No podemos tener en una sección un botón que dice "leer más" y en otra sección un botón de "ampliar información".

- Los diseños de todas las aplicaciones como formularios y encuestas deben ser iguales en todas las páginas.

- Los diseños y copys de los módulos deberán ser igual en todas las páginas. Si un botón dice “adelante”, no podrá existir otro botón que diga “siguiente”.

Principio 5: Prevención de errores (Error prevention). Incluso mejor que buenos mensajes de error es un diseño cuidadoso que evite que un error se produzca. Se debe evitar las condiciones que posibiliten los errores. Ofrezca una opción de confirmación antes de que se ejecute una acción. Ejemplos prácticos:

- Toda función debe llevar información contextual que le expliquen al usuario qué debe hacer y cómo hacerlo. Puede ser mediante tooltips o un párrafo inicial.

- Las acciones deben pedir confirmación al usuario. Por ejemplo: “¿Está seguro que desea borrar este documento?”.

- Las validaciones deben ser en tiempo real.

Principio 6: Reconocer en lugar de recordar (Recognition rather than recall). Como ya hemos referido, se debe minimizar la carga de memoria del usuario mediante decisiones, acciones y opciones visibles. El usuario no debería tener que recordar información. Las instrucciones de uso del sistema deben ser visibles y la ruta de navegación fácilmente recuperable cuando sea necesario. Ejemplos prácticos:

- El usuario deberá reconocer en todo momento como encontrar lo que busca. Los botones, menús de opciones, páneles, enlaces, los destacados, etc., deben ser lo suficientemente visibles y notorios para que el usuario no deba memorizar las funcionalidades.

- Un ejemplo claro son los carros de compras, donde el usuario debe cumplir varios procesos para llegar al estado final, cada estado debe ser obvio y el usuario no tendrá que memorizar como llenarlo.

Principio 7: La flexibilidad y la eficiencia de uso (Flexibility and efficiency of use). Se debe acelerar la interacción para el usuario de tal manera que el sistema pueda servir tanto a los usuarios sin experiencia y con experiencia. Se debe permitir a los usuarios adaptar las acciones frecuentes. Ejemplos prácticos:

- Se deben crear aceleradores que permitan al usuario visitar las secciones frecuentes. Por ejemplo se deben implementar módulos de “noticias destacadas”, “enlaces populares”, “lo más visitado”, “últimos eventos”, etc.

- El ejemplo más común es el atajo o ruta simple al inicio, generalmente dispuesto en el logotipo o con un ícono en la parte superior.

Principio 8: Diseño estético y minimalista (Aesthetic and minimalist design). Los diálogos no deben contener información que es irrelevante o raramente necesaria. Cada unidad adicional de información en un diálogo compite con las unidades relevantes de información y disminuye su visibilidad relativa. Ejemplos prácticos:

- Se deben manejar prioridades en la información, mucha información ya sea visual o textual entorpece la lectura del usuario. No toda información es relevante, los usuarios solo leen lo que les interesa. Usar resúmenes resaltados y highlights en los textos es suficiente para destacar un contenido, interpretar datos numéricos o estadísticos a través de gráficas e infografías permite resumir y priorizar la información. El diseño debe marcar el contenido prioritario y no distraer la atención de este.

Principio 9: Ayude a los usuarios a reconocer, diagnosticar y recuperarse de los errores (Help users recognize, diagnose, and recover from errors). Los mensajes de error deben ser expresados en un lenguaje sencillo (sin códigos), indicar con precisión el problema y sugerir una solución constructiva. Ejemplos prácticos:

- Una buena práctica es usar una página personaliza para los errores o bien una leyenda amable pero simple para indicarlos.

- Otro ejemplo es una ayuda textual para las búsquedas fallidas. Es necesario ayudar al usuario a encontrar lo que busca o a diagnosticar por qué no lo encuentra.

Principio 10: Ayuda y documentación(Help and documentation). A pesar de que es mejor si el sistema puede ser utilizado sin la documentación, puede ser necesario para proporcionar ayuda y documentación. Dicha información debe ser fácil de buscar, centrada en la tarea del usuario, arrojar una lista de medidas concretas que deben llevarse a cabo y no ser demasiado grande. Ejemplos prácticos:

- El ejemplo más conocido es el enlace de “condiciones de uso” y el de “FAQ” o “preguntas frecuentes”

- La opción de ayuda deberá estar siempre presente en todo proceso que involucre llenado de datos o bien interacción a preguntas con el usuario, o cuando la interfaz sea demasiado nueva o haya cambiado de forma radical.

b. Directrices

a. Directrices – Brown. (ISO/IEC 11581, 2000) Las recomendaciones de este autor son las siguientes:

- "Doble click" quiere decir clic + acción
- No poner botones de cerrar en diálogos modales
- Colocar botones "OK" y "CANCEL" en cualquier caja de diálogo modal
- Utilizar verbos en la barra de título en cuadros de diálogo de funciones
- Aprovechar la experiencia práctica
- Difundir e incorporar experiencia experimental aplicable
- Incorporar reglas de sentido común
- Promover consistencia entre los diseñadores responsables de partes diferentes de la interfaz
- Las directrices a veces pueden provocar conflictos, y por tanto siempre es importante aplicar test de usabilidad para tratar de resolverlos.

c. Estándares

Desarrollar estándares para la IGU significa lograr que los desarrollos de software sean más fáciles y seguros, estableciendo unos requisitos mínimos de fabricación, eliminando inconsistencias y variaciones innecesarias en las interfases.

Dentro de los más utlizados y populares se encuentran: Estándar de jure y Estándar de facto

Estándar de jure

Los estándares de jure son generados por un Comité con estatus legal y gozan del apoyo de un gobierno o una institución para producir estándares. En el caso del diseño ergónomico, el más popular de estos estándares es el establecido en la norma ISO.

La norma ISO 9241 es un estándar de *jure* relacionado con los requisitos ergonómicos para trabajar con terminales de presentación visual (VDT), tanto de hardware como de software. Las tareas de la oficina, procesamiento de texto y datos están cubiertos por la ISO 9241. Esta norma se enfoca a la calidad en usabilidad y ergonomía tanto de hardware como de software, fue creada por la ISO y la IEC.

La ISO 9241 se refiere a la colocación correcta de los equipos informáticos y periféricos, de modo que el trabajador pueda pasar el número máximo de horas de trabajo con el mínimo de molestias. La norma comprende 17 partes, de la 1 a la 9 tienen que ver con los equipos, el entorno y los puestos de trabajo. Las partes de la 10 a la 17, tratan de la ergonomía del software.

ISO 9241-10: Principios para diálogos

Esta parte describe principios generales de ergonomía juzgados importantes para el diseño y evaluación de diálogos entre el usuario y los sistemas de información (adaptación a la tarea, carácter auto descriptivo, control por parte del usuario, conformidad con las expectativas del usuario, tolerancia a errores, aptitud a la individualización, facilidad de aprendizaje). Estos principios pueden ser aplicados durante la especificación, el desarrollo o la evaluación de software como línea directriz general, y son independientes de cualquier técnica de diálogo específico. En este documento, cada principio está acompañado de una descripción seguida de ejemplos de puesta en práctica.

ISO 9241-11: Guía de especificaciones y medidas de usabilidad

Esta parte define la usabilidad y explica cómo identificar la información a tomar en cuenta para especificar o evaluar la usabilidad, en términos de desempeño y satisfacción del usuario. Este documento proporciona directrices para la descripción del contexto de usabilidad del software y las medidas pertinentes relativas a la usabilidad (medida de la eficacia y de la eficiencia).

ISO 9241-12: Presentación de la información

Esta parte proporciona recomendaciones ergonómicas relativas a la presentación y a las propiedades particulares de la información presentada en pantallas de visualización. Las recomendaciones proporcionadas tienen como objetivo permitir al usuario ejecutar tareas de percepción de manera eficaz y satisfactoria.

Aquí se aborda por lo tanto la organización de la información (ubicación de la información, adecuación de las ventanas, zonas de información, zonas de entrada/salida, grupos de información, listas, tablas, etiquetas, campos, etc), los objetos gráficos (cursores y punteros, etc), y las técnicas de codificación de la información (codificación alfanumérica, abreviación de códigos alfanuméricos, codificación gráfica, codificación por colores, marcadores, etc). Es proporcionado un ejemplo de procedimiento de evaluación y conformidad.

ISO 9241-13: Guía del usuario

Esta parte proporciona recomendaciones relativas a la ayuda del usuario. Las recomendaciones presentadas en esta parte están relacionadas al prompt, el feedback, el estado del sistema, la gestión de errores y la ayuda en línea. Las recomendaciones presentadas en esta parte deberían facilitar la interacción de un usuario con un programa, favorizando el uso eficaz del programa, evitando la carga de trabajo mental inútil, proporcionando a los usuarios un medio de gestión de errores y un asistente a los usuarios con niveles de conocimiento diferente.

ISO 9241-14: Diálogos de menús

Esta parte proporciona recomendaciones para el diseño ergonómico de los menús, es decir tipos de interacción en el que se presentan opciones a los usuarios bajo diferentes formas (ventanas de dialogo con casillas a marcar, botones, campos, etc). En esta parte, numerosas recomendaciones son condicionales, es decir que sólo deberían ser aplicadas en contextos específicos (ej. Tipo particular de usuario, de tarea, de entorno, de tecnología, etc). La aplicación de estas recomendaciones debería estar subordinada a un conocimiento de las tareas y de los futuros usuarios. Aquí se aborda entonces la estructura de los menús, la navegación en los menús, la selección y ejecución de las opciones y la presentación de los menús.

ISO 9241-15: Diálogos de tipo lenguaje de órdenes

Esta parte proporciona recomendaciones para el diseño y evaluación de los diálogos de tipo lenguaje de órdenes. Recordemos que en este tipo de diálogo, el usuario ingresa comandos completos o abreviados respetando la sintaxis del lenguaje de ordenes y el ordenador los ejecuta. Este documento aborda la estructura y la sintaxis del lenguaje de ordenes, la representación de los comando (nombres, abreviación, teclas de función, etc), los aspectos relativos a los modos de entrada y salida, el feedback y la ayuda.

ISO 9241-16: Diálogos de manipulación directa

En los diálogos de tipo de manipulación directa, los usuarios efectúan operaciones manipulando objetos que aparecen en pantalla como si manipularan entidades físicas (ej. Puntear, desplazar, etc.).

Esta parte aborda las metáforas graficas, la apariencia de los objetos utilizados en la manipulación directa, el feedback, los dispositivos de entrada de datos, la manipulación de objetos, el punteo y la selección, el dimensionamiento, la manipulación directa de las ventanas y los íconos, etc.

ISO 9241-17: Diálogos por cumplimentación de formularios

Los diálogos por cumplimentación de formularios, son diálogos en los que el usuario rellena, selecciona las entradas o modifica los campos indexados dentro de un formulario o de una ventana de diálogo, presentada por el sistema. Las recomendaciones dadas en esta parte tienen que ver con la estructura de los formularios, los campos y etiquetas, las entradas (textuales alfanuméricas, de opción, los controles, las validaciones, etc), el feedback y la navegación en el formulario.

ISO 14915: Ergonomía del software para interfaces de usuario multimedia

Esta norma se compone de 4 partes. La primera es la introducción y no contiene recomendaciones. La segunda parte proporciona recomendaciones acerca del diseño de controles y la navegación (ej. controles de audio, funciones como “play”, “stop”, “pausa”, etc). La tercera parte proporciona recomendaciones sobre medios específicos y sobre su articulación. En cuanto a la cuarta parte, esta concierne dominios de aplicación específicos como la formación asistida por ordenador, los bornes interactivos, etc.

ISO 13407 proceso de diseño centrado en el usuario para sistemas interactivos

Esta parte proporciona recomendaciones relativas a procesos de diseño centrados en el usuario a través de toda la vida útil de los sistemas interactivos informáticos. Esta norma está dirigida a los responsables de los procesos de diseño (los jefes de proyecto) y proporciona una guía de fuentes de información y normas que tratan del enfoque centrado en el usuario. Por lo tanto, esta norma tiene que ver con la planificación y la gestión del diseño centrada en el usuario. Aborda únicamente los aspectos técnicos del factor humano y de la ergonomía en la medida en que los jefes de proyecto necesitan comprender la adecuación e importancia de estos datos en relación al proceso de diseño en su conjunto.

Aquí están descritas las etapas de comprensión y especificación del contexto de uso, de especificación de las exigencias relacionadas al usuario y a la organización, de la producción de soluciones y de evaluación. Además, se indica al jefe del proyecto cómo evaluar la conformidad del proceso de diseño que ha puesto en práctica, con la norma 13407. Así, éste debe probar que los objetivos de usabilidad han sido objeto de test, y que estos han sido realizados utilizando métodos validos, y además que una cantidad apropiada de usuarios ha participado en él y son representativos de los futuros usuarios, y que los datos producto de estos test han sido tratados de manera apropiada.

ISO/TR 16982: Métodos de usabilidad que soportan diseño centrado en el usuario

Este es mas bien un reporte técnico (TR) y no una norma. Este documento presenta una lista de métodos ergonómicos que pueden ser aplicados a las diferentes etapas del ciclo de diseño, precisando sus ventajas y desventajas. Aquí se presentan los métodos implicando directamente a los usuarios finales (ej. La observación, la medida del desempeño, la técnica de los incidentes críticos, los cuestionarios, las entrevistas, las técnicas de diseño y evaluación participativa, etc) y los métodos que no implican directamente a los usuarios finales (ej. El análisis de documentos, guías de estilo, cuadros de evaluación, criterios ergonómicos, los métodos formales {ex., KLM, GOMS, MAD*}, etc). En resumen, esta es una presentación de métodos que son reconocidos en el dominio del diseño centrado en el usuario.

Estándares de facto

Son estándares que nacen a partir de productos de la industria que tiene un gran éxito en el mercado o desarrollos hechos por grupos de investigación en la Universidad que se divulgan rápidamente. Su definición se encuentra en manuales, libros y artículos. Son aceptados como tales por su uso generalizado. Por ejemplo: Lenguaje C y Normas CUA y IUSR.

Es fundamental para los desarrolladores basar sus diseños en un conjunto de principios y directrices para tener consistencia. Por este motivo es tan importante para las organizaciones que desarrollan software, disponer de una guía que puedan seguir sus desarrolladores. Estas guías se denominan guías de estilo y varían mucho en sus objetivos. Dependiendo del sistema hay guías de estilo para Macintosh, OS/2, Windows, UNIX y Java.

CUA

Fueron publicadas en 1987 por IBM juntamente con Microsoft fruto de una colaboración común. Se adoptó universalmente por la fuerza de IBM. Windows, OS/2 i Motif, son los estándares más importantes que siguen esta norma.

IUSR & CIF

A fin de facilitar el intercambio entre consultores y también entre consultores y universitarios de datos de test realizados a usuarios y para que los consumidores estén mejor informados sobre la calidad ergonómica de los programas interactivos, el National Institute of Standards and Technology ha puesto en marcha un proyecto llamado “The Industry Usability Reporting Project” (IUSR, http://www.nist.gov/iusr) (Scholtz & Morse, 2002). El objetivo de este proyecto es ofrecer una mayor visibilidad a la ergonomía de software. Para ello, el grupo ha definido un formato de presentación (The Common Industry Format, CIF) de los test realizados a los usuarios.

Este formato precisa los puntos que deben ser abordados en los reportes de test realizados a los usuarios, a saber: las tareas propuestas a los usuarios, el entorno del test (hardware y software), el protocolo del test, los métodos de recojo de datos, las técnicas de análisis puestas en practica y los resultados de la evaluación. Los miembros de este proyecto piensan extender el CIF a las evaluaciones ergonómicas de páginas web, de material informático, así como la evaluación de la accesibilidad. La normalización de la presentación de los resultados de los test realizados a los usuarios debería continuar a nivel internacional.

Finalmente es importante destacar que aunque se sigan estrictamente las normas de la guía no hay garantía de que la interfase sea usable. Es mejor seguir las guías que no seguirlas. Sólo cabe resalte que quizá considerando estos principios y aspectos técnicos, podamos hacer un diseño mejor.

Para concluir esta parte podemo referir a Fenoulière (2002) que indica que las normas pueden ser utilizadas según dos perspectivas:

(1) como ayuda que puede ser aportada por expertos que han transcrito lo que la profesión admite como algo ya establecido en la profesión,

(2) como una referencia cuya aplicación debe ser demostrada a organismos de certificación.

2.9 Métodos para el modelado de una Interfaz Gráfica

En realidad no existe un método único para modelar o diseñar una interfaz gráfica de usuario con propósitos específicos. El diseño de la interacción toma prestados muchos conceptos y modelos de las disciplinas de Ergonomía, Semiótica, Inteligencia Artificial y Ciencia Cognitiva. Los especialistas en Ergonomía de Software han acuñado dos neologismos para medir la adecuación de la aplicación a las capacidades y limitaciones de los usuarios: la Usabilidad y la Accesibilidad.

Jakob Nielsen, considerado el padre de la Usabilidad, en 1993 la definió como el atributo de calidad que mide lo fáciles de usar que son las interfaces. La Usabilidad de un sistema es una medida de su utilidad, facilidad de uso, facilidad de aprendizaje y apreciación del usuario para una tarea, un usuario y un contexto dado. Es fundamental seguir principios ergonómicos para minimizar la carga cognitiva. Una aplicación usable es la que permite que el usuario se concentre en su tarea y no en la aplicación.

La usabilidad garantiza que en el análisis hayan sido incluidos los usuarios finales, sus necesidades, experiencia previa, costumbres, capacidades y limitaciones. Además les permite un uso más eficiente de la aplicación y la hace más fácil de aprender, reduciendo tiempos y costos

de formación. Esto favorece también a los usuarios menos favorecidos en cuanto a capacidades y experiencia. Y como se mencina previamente, se disminuye también la carga mental del usuario ya que una aplicación usable es aquella que no requiere de esfuerzos de memorización por parte del usuario.

Con respecto a la Accesiblidad, podemos decir que toda interfaz debe evitar poner barreras para interactuar con ella, esto incluye desde las dificultades físicas para manipular los dispostivos y las barreras cognitivas para entender los procedimientos y la navegación. Estudios realizados con usuarios evidencian la necesidad de desarrollar interfases adaptables que permitan el control del sistema en entornos inteligentes.

Por otro lado, los seres humanos son diferentes entre sí y todas las interfaces de usuario deberían acomodarse a esas diferencias de tal modo que cualquier persona fuera capaz de utilizarla sin problemas. El objetivo a lograr en este caso es un Diseño más universal, que pretende que nadie se vea limitado en el uso de algo por causa de esas diferencias. Puesto que una gran cantidad de los esfuerzos en interfaz actuales se apoyan en elementos gráficos, resulta lógico ofrecer a los usuarios con visión reducida la opción de utilizar esos elementos en la medida que sea posible. La reducción de la carga visual es el objetivo a lograr.

Dicho de otra manera, el diseño de las IGUs debe concentrarse en el conocimiento del usuario y de sus capacidades, ya que los sietemas de información deben solucionar problemas al usuario. El sistema tiene que pensar en las tareas del mismo y en que esas tareas son el objetivo de la propia interacción hombre-máquina. Por ello se insiste en la participación del usuario en las tareas de diseño de la interfaz.

Para diseñar una interfaz es necesario pasar por cuatro etapas:

- Análisis de requerimientos del producto, análisis de las tareas. Conocimiento del usuario. Generación de posibles metáforas y análisis del tipo de diálogo. Revisión.
- Generación de prototipos virtuales (layouts) o físicos para investigar desde lo general hasta el detalle. Desarrollo de la aplicación, del sitio o del sistema.
- Planificación (desarrollo del plan, definición de las medidas, selección de participantes, formación de observadores, preparación de los materiales). Test (prueba piloto, test con usuarios).Conclusión (análisis de los datos, elaboración del informe, resultados y recomendaciones).
- Comparación con estándares (internos y/o externos), versiones anteriores del mismo producto y productos competidores. Verificación de las diferencias. Generación de nuevas metas.

Adicionalmente a este método práctico para el diseño de la IGU, existen otros métodos o modelos para orientar su diseño. El primero de ellos es parte del Modelo de Ingeniería de Requerimientos que refiere como fundamental establecer los requerimientos de la interfaz y que a su vez subdivide en un proceso de desarrollo que podemos definir de la siguiente manera:

Los requerimientos de interfaz son todos aquellos elementos que debe proveer el sistema para permitir la interacción entre el usuario y las funcionalidades que este tiene, con el fin de que en el proceso de diseño se tenga claridad de las interfaces que se deben crear y la relación que debe existir entre ellas.

Para la definición de los requerimientos de interfaz se deben identificar lo siguientes elementos:

- Id: Identifica de manera única una interfaz gráfica
- Descripción: Indica los elementos que debe tener la interfaz.
- Requerimientos asociados: Indican las funcionalidades asociadas a la interfaz gráfica.

En este nivel, no se va definir de manera detallada la interfaz, solo se pretende tener una primera aproximación a los elementos que deben ser tenidos en cuenta en el desarrollo de estas.

Otro modelo utilizado, particularmente en las interfases Web, es el modelo heurístico de Jakob Nielsen (2005). Las 10 heurísticas de usabilidad para el diseño de interfaz de usuario, se han enlistado previamente, porque se consideran principios, directrices y normas, así que bien tienen cabida en más de un apartado y se recuperan particularmente al integrar la interfaz del sistema con el EV.

Y finalmente describiremos rápidamente el Modelo SSOA O Modelo objeto–acción sintáctico–semántico de Schneiderman (1992), que se trata de un modelo del conocimiento del usuario. Provee un marco de definición sobre las diferentes formas y grado de conocimiento que un usuario puede emplear al interactuar con un sistema. La distinción entre las sintaxis y semántica, surgió a partir de la definición de compiladores, donde se separó el proceso de compilación de la entrada de texto o comando, denominado análisis sintáctico, del proceso de interpretación del mismo, denominado análisis semántico.

Los diseñadores de los sistemas interactivos también pueden contar con un modelo sintáctico semántico, pero aplicado al conocimiento del usuario. Este conocimiento sintáctico se refiere al entendimiento de detalles dependientes al uso de los dispositivos. El conocimiento semántico, incluye información conceptual concerniente a la aplicación -objetos y acciones del dominio de la

tarea- y al uso general de los sistemas interactivos.

De acuerdo al grado de conocimiento semántico y sintáctico que el usuario posea, se puede realizar una clasificación general de los mismos en tres categorías, tal como lo muestra la siguiente tabla:

Tabla 2.4 Clasificación del grado de conocimientos de los usuarios

Clasificación SSOA de los usuarios	Explicación
Novato	• No tienen ningún conocimiento sintáctico. • Cuentan con un conocimiento semántico pobre sobre el uso de sistemas interactivos. • Medianamente tienen un entendimiento de la aplicación.
Intermedio	• Tienen buen conocimiento semántico tanto del uso de sistemas de software como del dominio de tareas. • Presentan inconvenientes a nivel de cómo se llevan a cabo las tareas en el sistema, con problemas en recordar detalles de conocimiento sintáctico.
Experto	• Están muy familiarizados con todos los aspectos sintácticos y semánticos del sistema.

Es necesario aclarar que los conceptos de conocimiento sintáctico y semántico tienen una correspondencia directa con las capas que presenta el software. El corazón funcional procesa información semántica mientras que el procesamiento de información sintáctica pertenece a las capas propias de la interfaz del usuario.

Conclusiones del capítulo

Como se puede apreciar en lo hasta este momento abordado, existe un gran número de esfuerzos, investigadores e instituciones y organismos que están contribuyendo al desarrollo de los Entornos Virtuales y las interfaces gráficas. Esto, por una parte, resulta enormemente positivo, pues enriquece la visión que se tiene de estos sistemas. Por otra parte, sin embargo, presenta serios retos que son cada vez más urgentes de resolver a medida que crece la cantidad y complejidad de los mismos y que avanzan el desarrollo tecnológico.

Este capítulo nos da la visión global de lo que se ha desarrollado y propuesto en distintos momentos en los campos que nos ocupan. El análisis permite partir de múltiples y complejos conceptos que serán considerados en la propuesta del sistema. Podemos decir que la revisión general de la teoría y los distintos esfuerzos e investigaciones que se analizaron a lo largo de

esta sección han logrado que se respalde de una forma más completa el resultado que será presentado en la propuesta.

Se han revisado no sólo elementos teóricos y conceptuales, también estándares y normas que permitirán establecer una propuesta más *ad hoc*. Mediante el estudio, exploración y puesta en práctica de estas normas y recomendaciones de diseño, se logrará mejorar significativamente la propuesta, considerando desde las funciones y arquitectura, hasta la comunicación e interacción con el usuario final, optimizando la calidad del sistema en general.

El análisis de los entornos virtuales y su consecuente función educativa, permite dirigir los esfuerzos que se plantean como objetivos de este proyecto y que serán tomados en cuenta al momento de proponer el sistema. Adicionalmente se ha podido identificar la importancia de los componentes y funciones que deberán formar parte sustantiva de la propuesta y sobretodo, que serán estructurados, diseñados y presentados con los atributos que se han venido refiriendo desde el inicio. Así la arquitectura, con todo lo que se ha revisado a lo largo de este apartado, deberá establecer relaciones estructurales y conceptuales partiendo de los estándares y elementos que aquí se han estudiado.

La interfaz, por su parte, tal y como se ha visto se construye por y para los/las usuarios/as, por lo que debe tener en cuenta desde aspectos de visualización, servicios, estilo de interacción, formas de diálogo, comportamiento que debe ser acorde a sus necesidades. La interfaz del usuario constituye un puente, un nexo indispensable y único entre el usuario y la componente funcional del sistema. Es un concepto amplio y complejo, puede tomar diferentes dimensiones y se ve afectada por múltiples factores que se intentó analizar y explorar a lo largo de todo este material.

1 Java es un lenguaje de programación orientado a objetos desarrollado por Sun Microsystems a principio de los años 90´s.

2 VRML (sigla del inglés Virtual Reality Modeling Language. “Lenguaje para Modelado de Realidad Virtual”) - formato de archivo normalizado que tiene como objetivo la representación de escenas u objetos interactivos tridimensionales; diseñado particularmente para su empleo en la web.

CAPÍTULO 3:

PLANTEAMIENTO DE LA PROPUESTA

CAPÍTULO 3. Planteamiento General de la Propuesta.

El propósito de este capítulo es argumentar el por qué se seleccionaron y consideraron los Entornos Virtuales de Enseñanza-aprendizaje con Tutoría Inteligente, así como la razón de utilizar agentes en el diseño de su arquitectura y para su posterior implementación. Adicionalmente se explica, la motivación de enfocar su desarrollo desde el punto de vista de la arquitectura y el diseño de la interfaz gráfica de usuario (IGU). Este capítulo se complementa con el capítulo 4, que aborda todo lo relativo a la Construcción del Entorno Virtual.

3.1 Sistema de Tutoría Inteligente y Entornos Virtuales

Los Entornos Virtuales (EV) son un tipo de sistema software de aparición relativamente reciente. Como ya se ha mencionado, los primeros ejemplos se pueden situar a finales de los años 70, fundamentalmente se trataba de aplicaciones textuales enfocadas principalmente hacia el entretenimiento. A mediados de los 80 se les empieza a dotar ya de interfaz gráfica, y a principios de los 90 surgen las primeras aplicaciones en tres dimensiones (3D). A partir de ese momento, los EVs han sido utilizados con gran variedad de fines, incluyendo entretenimiento, formación, trabajo colaborativo, simulación o marketing.

Considerando sus características, el usuario de un sistema de Realidad Virtual se encuentra asistido por un conjunto de dispositivos que le permiten interactuar con el Mundo Virtual, y que refuerzan la sensación de inmersión dentro del mismo. Dependiendo del tipo de configuración realizada, basada en un conjunto determinado de componentes auditivos, visuales, táctiles y un software de simulación, puede hablarse de distintos niveles de inmersión en el sistema:

La denominada Realidad Virtual de Escritorio que es la que proporciona los niveles más bajos de inmersión, ya que sólo permite interactuar con el mundo a través de un teclado, un *joystick*[1] o control o un ratón, y donde el mundo se visualiza en un monitor de computadora normal común y corriente.

---

1 Traducido al español significa palanca de mando. Es un dispositivo con una palanca especial para ser tomado de manera ergonómica con una mano, y una serie de botones integrados en la palanca que controlan en la pantalla los movimientos y acciones de los objetos en los videojuegos.

La Realidad Virtual Inmersiva, en cambio, emplea dispositivos tales como HMD (Head Mounted Displays, cascos de Realidad Virtual), guantes, sistemas de rastreo del movimiento, etc. La sensación de presencia en el EV es mucho mayor en este tipo de sistemas, pero debido a que su costo es muy elevado, quedan fuera del alcance de la mayor parte de los usuarios.

La inmersión en un mundo virtual hace desaparecer la interfaz hombre-máquina, de forma que nuestras experiencias en el mundo virtual pueden llegar a ser de la misma calidad que nuestras experiencias en el mundo real. El conocimiento que se genera es directo, personal, subjetivo y en primera persona, frente al conocimiento en tercera persona que se adquiere cuando nos ha sido enseñado por alguien más.

Existen gran cantidad de documentos y estudios que indican que la tecnología de la Realidad Virtual es una poderosa herramienta para la enseñanza, debido fundamentalmente a su capacidad para proveer entornos inmersivos, multisensoriales, creíbles y basados en la experiencia, entre otras características (Bricken, 1990a; Bricken, 1990b; Byrne, 1993; Winn, 1993; Larijani, 1994; Pantelidis, 1996; Youngblut,1998; Hughes y Moshell, 1997). Todos estos trabajos sugieren que la aplicación de esta tecnología en los procesos de enseñanza/aprendizaje, ya sea en la educación o en el entrenamiento, puede generar beneficios superiores con respecto a los obtenidos mediante la utilización de tecnología no basada en Realidad Virtual. Sin embargo, debido al actual estado de desarrollo de la Realidad Virtual, muchas de las aplicaciones que se construyen en instituciones de investigación y universidades se encuentran todavía en proceso de desarrollo o quizá en pruebas firmes, pero aún no se piensa en ellas para su comercialización o su distribución masiva.

Dentro de los EVs educativos se pueden encontrar tres tipos de aplicaciones bien diferenciadas. En primer lugar, existe un tipo de EVs que sirven de soporte al e-learning, como define Kuljis (2002) en INVITE. En este tipo de aplicación, el EV se utiliza como lugar de encuentro para charlas, impartir lecciones o establecer discusiones. En segundo lugar se encuentran los EVs para enseñanza, como NICE y REP de Johnson y otros (1999a y 1999b) que son aplicaciones que reproducen un entorno más o menos real, y que se utilizan, en la mayor parte de los casos, para enseñar conceptos sobretodo a niños en edad escolar. Aunque existe un número amplio de aplicaciones de este estilo, los propios autores reconocen que su utilidad no es concluyente, pues los resultados obtenidos son similares a los de situaciones en que no se usan estos entornos. Adicionalmente a este fenómeno y quizá por su novedad, los niños tienden a dar mayor importancia a lo aprendido al usar el EV que a lo explicado por sus profesores, de manera que, si se aprende algo de manera incorrecta en estos ambientes, resulta mucho más difícil lograr que los niños desechen esa idea y aprendan el concepto correcto.

Y en tercer lugar tenemos los Entornos Virtuales de Entrenamiento (EVE). Este tipo de EV suele reproducir un entorno real de manera bastante fidedigna, y suelen estar enfocados a la enseñanza de tareas fundamentalmente procedimentales que posteriormente habrían de llevarse a cabo en la realidad, razón por la cual es fundamental que el EV sea lo mías parecido posible al escenario real, tanto en apariencia como en funcionamiento. Hasta la fecha son éstos los que han dado mejores resultados en cuanto a su eficacia.

Los EVEs han experimentando un gran avance durante los últimos años, siendo utilizados fundamentalmente en tres áreas: medicina (Korze y otros, 2001), industria (Méndez y otros, 2001) y entrenamiento militar (Rickel y otros, 2002b).

Dentro de los EVEs podemos establecer dos grupos claramente diferenciados:

En algunos casos, el EV no posee ninguna característica que lo oriente específicamente hacia el entrenamiento, sino que el uso que se le da es el que hace que sea considerado un EVE, por ejemplo en la educación o bien en la capacitación empresarial.

En otros casos se han incorporado sistemas de tutoría inteligente que apoyan el proceso de entrenamiento o aprendizaje.

Dentro de este último grupo se puede encontrar una amplia gama de aplicaciones, en función de su inteligencia y funcionalidad. Estas aplicaciones varían desde aquellas que controlan totalmente las acciones del estudiante, permitiendo realizar únicamente las acciones correctas para completar la actividad, hasta otras que proporcionan tutores con representación física en el EV y que permiten que el alumno realice acciones erróneas y haga preguntas para saber qué tiene que hacer.

En estos tutores, tal como refieren Lester y otros (1997c) se identifica el llamado efecto persona, el cual, según los autores de la investigación referida, parece tener un impacto considerable en el aprendizaje. Este trabajo muestra que los alumnos obtienen mejores resultados cuando reciben las explicaciones de un tutor con presencia física en el EV y con apariencia humana, en lugar de recibirlas de un tutor con apariencia no humana o de un tutor sin representación física. Por esta razón, es cada vez más habitual que los EVEs proporcionen tutores virtuales que acompañan al alumno durante todo el proceso.

Otro aspecto a destacar de este tipo de entornos es la posibilidad de realizar entrenamientos o instrucción en grupo, ya que habitualmente las actividades que se requieren suelen llevarse a cabo por más de una persona.

Como se ha mencionado anteriormente, se requiere que el EVE sea lo más parecido a lo que el alumno se encontraría en el entorno real, y resulta cada vez más importante que más de una persona pueda tomar parte en el entrenamiento. Considerando las características enumeradas a lo largo de esta sección, los sistemas que van a centrar la atención de este trabajo son los Entornos Virtuales de Entrenamiento con capacidad para soportar múltiples usuarios y que incorporan tutores inteligentes, con o sin presencia física en el entorno, aunque al cien por ciento la función particular del sistema no sea exclusivamente el entrenamiento sino la capacitación y el aprendizaje.

Como se ha podido observar en los trabajos e investigaciones referidas en este documento, existe una amplia gama de funcionalidades que este tipo de aplicación puede ofrecer. Por lo tanto, resulta conveniente delimitar la funcionalidad que debe ofrecer la arquitectura que se va a proponer en esta tesis. No se pretende elaborar una descripción detallada de la funcionalidad de los sistemas objeto de estudio a la manera de una especificación de requisitos. Esto requeriría, por una parte, cierta dependencia del dominio concreto en el que se quisiese trabajar, y por otro, incluiría numerosos detalles que no tendrían mayor impacto en la arquitectura. Además, como indica Clements (2002), no es necesario disponer de toda la información detallada acerca del sistema para poder comenzar el diseño de la arquitectura y como expresa J. James Garret (2009) las funciones básicas de la arquitectura se pueden traducir sin problemas en el planteamiento de una iterfaz consistente sin requerir el máximo detalle, sólo la traducción al diseño simbólico y se pueden cotejar con esquemáticas básicas.

Por lo tanto, lo que aquí se presenta es una descripción de la funcionalidad que se espera obtener en una aplicación que utilice la arquitectura propuesta como solución del problema que se plantea. Por tanto la mayor parte de la funcionalidad descrita se ha obtenido a partir del resultado del análisis de las investigaciones descritas en el capítulo 2 y puesto que uno de los objetivos que se persiguen es la posibilidad de intercambiar componentes de la arquitectura, la descripción de la funcionalidad se va a organizar basándose en los módulos que componen los Sistemas de Tutoría Inteligente, lo cual no obligaría a que la arquitectura de la aplicación responda a la estructura clásica de un STI.

3.1.1 Funciones de Tutoría

El sistema debería plantear distintas actividades para que el alumno sea capaz de resolverlas dentro del EV, ya sea de manera supervisada por el STI o sin supervisión alguna. Si las actividades se realizan con supervisión, el alumno podría realizar preguntas acerca de lo que tiene que hacer y el STI podría decidir si las responde o no, así como el nivel de detalle con el que ofrece la

respuesta.

Además, en el caso en que se detecte que el estudiante realiza acciones incorrectas, se podría decidir si dejarle continuar o si detenerlo e indicarle dónde está el error para que intente realizar una acción diferente. Tanto en este caso como si se queda parado sin saber qué hacer, el sistema podría ofrecerle al estudiante pistas que le ayuden a continuar.

Lo ideal sería que la tutoría se adaptara a las características de aprendizaje del alumno, lo cual se conseguiría a través del modelo que se realice del estudiante. En cualquier caso, la tutoría también debería poder realizarse de una manera estándar, sin tener en cuenta las características del estudiante.

3.1.2 Funciones del Experto

La responsabilidad básica del experto como parte del sistema es poseer el conocimiento que debe ser enseñado al estudiante. Dentro de este conocimiento se encuentra el necesario para resolver los tipos de ejercicios que se le plantean al alumno para que los lleve a cabo dentro del Entorno Virtual. Para poder realizar una tutoría más personalizada y flexible, el experto debe ser capaz no sólo de proporcionar una solución a los ejercicios, sino que también debería poder resolver las situaciones planteadas por el alumno cuando resuelva los ejercicios de manera distinta a la propuesta por el experto. Además, debería facilitar a los estudiantes explicaciones sobre los ejercicios, tendría que ser capaz de responder a sus preguntas y proporcionar información para darles pistas que les ayuden con la resolución de un ejercicio.

Con el objetivo de personalizar la tutoría, debería facilitar al tutor la información que éste le solicite, por ejemplo, para saber si es correcta la acción que quiere llevar a cabo un estudiante, de manera que, de no ser así, el tutor pueda decidir si le deja continuar.

Puesto que el objetivo de la propuesta es posibilitar la capacitación y aprendizaje en un Entorno Virtual, el experto debe tener conocimiento no sólo de la materia a impartir y de cómo resolver ejercicios, sino también de la estructura física de los escenarios y su contenido, de manera que sea posible saber cómo moverse por un escenario o contestar una pregunta respecto a la situación de un objeto.

En cualquier caso, debería ser posible realizar una tutoría menos sofisticada en caso de que el experto no proporcione alguna de las funcionalidades mencionadas.

3.1.3 Funciones de Modelado del Estudiante

El propósito de realizar un modelo del estudiante es adaptar el proceso de tutoría a las características del alumno. Para ello, es necesario que la estrategia de tutoría empleada haga uso de los datos que se vayan recopilando sobre el estudiante.

La información que se recopila deberá estar relacionada tanto con las acciones que el estudiante realiza, como con sus movimientos por el EV. También será necesario registrar el tiempo que tarda en realizar una determinada acción o grupo de acciones, la cantidad de preguntas que realiza, de qué tipo son, cuántas pistas hay que darle y su nivel de detalle o, incluso, hacia dónde enforcarse, para poder inferir información acerca de su atención.

Ya se ha mencionado que el modelo del estudiante está bastante relacionado con la estrategia de tutoría, por lo que la información concreta que contenga deberá ir en relación con la estrategia seleccionada. Esto no obstaculiza que el modelo pueda incluir información que no utilice directamente la estrategia de tutoría, ya que puede servir para ser analizada posteriormente por un tutor humano.

Por otro lado, la estrategia de tutoría debería ser capaz de tomar acciones por omisión si el modelo del estudiante no contiene alguna información que sea necesaria para una mayor personalización de la tutoría.

3.1.4 Funciones Relativas al Entorno Virtual

La naturaleza y funcionalidad de los Entornos Virtuales que se pueden utilizar junto con un STI son bastante variadas, por lo que, para una integración entre ambos con unas mínimas garantías de éxito, al menos debería ser posible que el EV informase de las acciones realizadas por el usuario, así como de las posiciones de los distintos personajes y objetos que lo integran.

En primer orden y en cuanto a las entradas que debería aceptar el EV, al menos debería ser posible cambiar la posición y orientación de los objetos, así como manejar un tutor con representación física dentro del EV.

Respecto a la funcionalidad, podemos decir que ésta depende enormemente del dominio en el que se realice el entrenamiento. No obstante, teniendo en cuenta que lo que se pretende es realizar un entrenamiento disciplinar y procedimental, debe manipular de alguna manera los objetos que forman parte del procedimiento que se entrena, adicionalmente a considerar los aspectos básicos de la disciplina. Y adicionalmente, cuando las características del escenario y del

entrenamiento así lo requieran, también se debería poder navegar por el entorno.

En cuanto a la forma de interacción entre el usuario y el EV, no existe ninguna restricción al respecto, por lo que son aceptables desde sistemas de escritorio, con un teclado, un ratón y un monitor, hasta sistemas totalmente inmersivos, que utilicen un dispositivo similar a un CAVE[2], gafas polarizadas, un casco de realidad aumentada, un dispositivo similar a una varilla (wands) o manos virtuales (virtual hands) . Tanto es así que debería ser posible sustituir el EV por cualquier otro elemento que produzca las mismas salidas (como una consola o una interfaz basada en ventanas) sin que el STI se vea afectado. Esto puede resultar de especial utilidad en el proceso de desarrollo, en el que no sería necesario poner en funcionamiento el EV para poder probar el STI.

3.1.5.Funciones de la Interfaz

La interfaz de un elemento es el medio por el cual éste interactúa con su entorno, considerando la interacción como cualquier cosa que hace un elemento que puede afectar a otro. La información acerca de una interfaz depende de la vista donde se documente dicha interfaz. En general, las interfaces son bidireccionales, por lo que es importante documentar no sólo lo que un elemento ofrece al exterior, sino también lo que ese elemento necesita del exterior. Además, dentro de una misma vista, un elemento puede tener más de una interfaz para, por ejemplo, restringir los servicios a los que otros elementos pueden acceder. Además, la existencia de míltiples interfaces facilita la evolución de un elemento.

Cuando se utiliza una interfaz gráfica clásica, la interacción con la aplicación se realiza a través de menús, botones y controles similares, pero el fin no es aprender a utilizar esos controles, sino que son un medio para comunicarse con la aplicación. De forma contraria, cuando se utiliza un EV, el fin que se persigue sí es aprender a manipular lo que aparece dentro de él. Por este motivo, es necesario que el STI tenga conocimiento de lo que ocurre dentro del EV, su estado y las posibilidades de interacción.

El EV constituye la principal forma de interacción del usuario con la aplicación. Para este caso se plantea de origen una interfaz bidimensional pero sin descartar algunos elementos de interfaz tridimensional que reproduzcan el entorno real en el que se realiza el aprendizajo o entrenamienteo, con las licencias pertinentes debidas a restricciones técnicas o a ayuda, como función prioritaria (Mendez, 2008). A través de la navegación por el escenario y la manipulación de los objetos que

2 CAVE (Cave Automatic Virtual Enviroment) Son dispositivos en los que el usuario se encuentra dentro de un habitáculo rodeado de pantallas que proyectan un espacio virtual altamente inmersivo y muy real. Con ello se pueden crear entornos de trabajo para entrenamiento o simulación.

en ella aparezcan, el estudiante intentará realizar lo que se plantea o resolver el trabajo o tarea que desea completar.

3.1.6.Funciones de Simulación

En determinados dominios, como sucede con el entrenamiento médico o en centrales nucleares por ejemplo, se puede dar el caso de que el sistema deba funcionar junto con un simulador que reproduzca el comportamiento de la central o la evolución del estado del paciente. Este simulador puede estar integrado con el EV o puede funcionar por separado, pero en cualquier caso debe ser posible que el STI se comunique con él para obtener información relativa a la simulación que sea relevante para el proceso de tutoría.

3.2 Sobre el Uso de Agentes Pedagógicos

Como se ha podido ver en el capítulo 2, el término agente posee distintas definiciones, que se reflejan en los diferentes usos que les dan los grupos de investigación que los utilizan, así como las particularidades de las áreas de investigación en que se emplean.

Por este motivo, no existe aún una definición comúnmente aceptada, así como sus características definitorias. Esta situación provoca que el término de agente corra el peligro de perder cualquier tipo de significado que pueda resultar de utilidad, como afirman los autores Wooldridge y Jennings (1998). Dentro del área de aplicación de la tesis se maneja el concepto de Agente Pedagógico Híbrido, con el cual se denomina a un “personaje” que habita un Entorno Virtual, que generalmente interactúa con el usuario y que exhibe un comportamiento más o menos complejo en función de las necesidades e intereses de quien lo desarrolla. Se plantea esta posibilidad, ya que es posible concebir sistemas heterogéneos cuyo comportamiento se derive de los dos tipos de agentes, reactivos y cognitivos y tenga características de ambos. Es decir, es posible dotar a los agentes cognitivos de capacidades de reacción a los eventos.

Estos Agentes Pedagógicos pueden estar diseñados e implementados de muy diversas maneras, y no existe una correspondencia directa entre ellos y los agentes que, considerados como elementos software, se utilizan para desarrollar aplicaciones de manera similar, por ejemplo, a los objetos.

Son éstos últimos, los agentes vistos como elemento de desarrollo de software, los que se considerarían en esta tesis para diseñar una Arquitectura para Entornos Virtuales. A través de la interacción entre estos agentes se dotaría a la Arquitectura de un comportamiento que simule a

un tutor, el cual, a su vez, podría estar representado como un Agente Virtual Inteligente dentro del Entorno Virtual.

Existen diversos trabajos que ponen de manifiesto las diferencias y analogías entre objetos y agentes, como en la propuesta de Odell (2002), donde se resalta la relativa similitud de ambos con una cierta ventaja para los agentes cuando se trata de construir sistemas altamente interactivos, distribuidos o descentralizados.

Zambonelli (2003) por ejemplo, realiza una comparación entre desarrollos utilizando objetos, componentes y agentes para mostrar sus diferencias y, en última instancia, resaltar las ventajas de los últimos para el desarrollo de determinado tipo de sistemas modernos.

Para poder razonar sobre la utilización de agentes es necesario, en primer lugar, adoptar una definición de agente. Se van a considerar las definiciones proporcionadas por Wooldridge (1997), Wooldridge y Jennings, (1995) y Zambonelli (2003), que establecen que un agente es una entidad software que posee las siguientes características:

1. *Autonomía:* capacidad para tomar el control de sus propias acciones.
2. *Situación:* capacidad de percibir el entorno en el que se encuentra y actuar sobre él.
3. *Reactividad:* capacidad para responder a estímulos externos.
4. *Proactividad:* capacidad de tomar la iniciativa para conseguir sus metas.
5. *Capacidad social:* habilidad para relacionarse con otros agentes.

Esta definición de agente proporciona un nivel de abstracción mayor que el de los objetos o los componentes según Wooldridge (1997), a pesar de que la evolución de ambos apunta a una paulatina convergencia de características como lo exponen Zambonelli y sus colaboradores (2003). Este mayor nivel de abstracción, como herramienta de diseño, ayuda a afrontar la complejidad que presentan los sistemas objeto de estudio.

Y sobre estas capacidades hace referencia el trabajo descrito en Medinilla y Gutiérrez (2007), donde los autores describen que las abstracciones son una herramienta para combatir la creciente complejidad del software. Esto es posible porque, según los mismos autores, las abstracciones son un recurso de simplificación que introduce ambigüedad en los diseños. Esta característica conduce a su vez a una mayor facilidad para realizar cambios en los diseños y, como consecuencia, a una mayor modificabilidad de las aplicaciones, allí donde la introducción de ambigüedad se haya utilizado como recurso de diseño.

Los Entornos Virtuales son aplicaciones cuya complejidad es notable por varias razones: el alto grado de interactividad con el usuario que requiere soportar, el nivel de adaptación al estudiante que se requiere de estos entornos, el gran esfuerzo que requiere modelar y simular el comportamiento de un tutor humano a través de una aplicación o de la computadora misma y la variedad de perfiles y elementos involucrados en su desarrollo. Esta complejidad es la que provoca que los agentes, como herramienta para combatirla, constituyan la mejor alternativa a explorar a la hora de elegir una forma de afrontar la construcción de estos sistemas.

Por sus características de autonomía, reactividad y proactividad, y la flexibilidad que éstas les proporcionan en su comportamiento, los agentes parecen estar especialmente indicados para sistemas que requieren un alto nivel de interacción y adaptabilidad a los usuarios, como es el caso de los EVs.

La característica de situación también los hace especialmente indicados para su utilización en un EV, el cual deben percibir y sobre el cual deben actuar. La capacidad social los hace adecuados para trabajar en sistemas formados por más de un agente, ya que deberían relacionarse entre ellos para lograr un comportamiento que simule el de un tutor humano.

Existe, para analizar, otra característica, que aunque menos relevante, hace que el uso de agentes constituya una alternativa razonable de desarrollo de software, como es la existencia de plataformas como JADE, Zeus o FIPAOS, que proporcionan ya una infraestructura básica para poder implementar los agentes, de manera que se eliminan algunas dificultades inherentes a ellos, tales como las que se señalan Wooldridge y Jennings (1998). En sistemas basados en agentes, gran parte del tiempo de desarrollo se consume en el diseño e implementación de infraestructuras, como mecanismos de comunicación, mensajes, directorios o la arquitectura de los propios agentes, por lo que la existencia de plataformas como las mencionadas, constituye un valor añadido que facilita su uso y desarrollo.

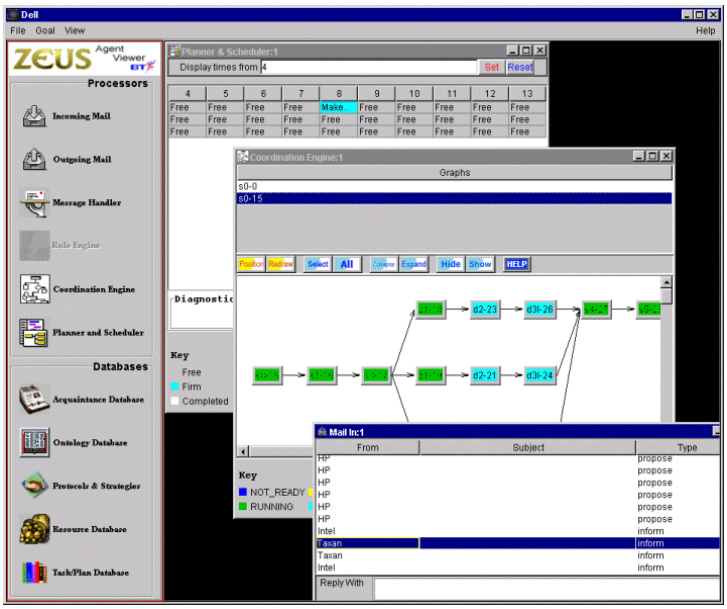


Figura 3.1: Vista general de la Interfaz de ZEUS

Figura 3.2: Interfaz de Interacción con plataforma JADE

3.3 Arquitectura Software

Según se menciona en Wooldridge y Jennings (1998), existen bastantes sistemas desarrollados por medio de la utilización de agentes, y se dispone de cierta cantidad de resultados tanto teóricos como prácticos que atestiguan su validez. Sin embargo, aunque parece que se están produciendo avances en el aspecto más teórico de los agentes, no son tantos los trabajos realizados que ayuden a entender cómo diseñar y construir sistemas basados en agentes y sobretodo que

presenten detalladamente una propuesta de interfaz gráfica que cumplan con las funciones y presentación que realmente se requiere.

Por estos motivos, inicialmente se consideró como parte de la propuesta, la creación de una metodología como extensión de Senda3D (Sánchez, 2001), metodología para el desarrollo de Entornos Virtuales y se le adiciona el aporte más significativo, desde la óptica de una diseñador de interfaz, que es la traducción simbólico-metafórica, tanto en la construcción de la interfaz, como en la presentación del Agente. Además de que se retoman elementos para presentar y argumentar las ventajas de la modificabilidad para la Arquitectura, de igular forma como metodología.

Se puede considerar que el propósito de una metodología de desarrollo es establecer un conjunto estructurado de actividades para obtener un sistema software de alta calidad con costos aceptables en un tiempo determinado (Pressman, 2004; Sommerville, 2007). Llevado al problema que nos ocupa, ninguna metodología existente establece criterios para determinar qué agentes deben existir, cuáles deben comunicarse entre ellos o cómo deben comunicarse, sino que el diseñador debe utilizar otros criterios para llegar al resultado final.

Este hecho, unido a la actual existencia de diversas metodologías para el desarrollo de sistemas basados en agentes, ha llevado a descartar la necesidad de crear otra metodología más, aunque sea específica para Sistemas de Tutoría Inteligente basados en agentes.

Se considera entonces, que resultará de mayor utilidad centrarse en uno de esos aspectos donde las metodologías dejan al diseñador a merced de su criterio y experiencia, como es la arquitectura software. De esta forma, un objetivo adicional que se plantea en este trabajo consiste en entender mejor cómo diseñar y construir un Entorno Virtual en el que el Tutor Virtual Inteligente se desarrolle a través de un sistema basado en un Agente Pedagógico. De esta manera, la metodología que se esté utilizando, en caso de que se use alguna, puede completarse con el diseño de la arquitectura que se propone como solución en este trabajo y enriquecerse con la metodología para el diseño de la intefaz gráfica.

3.3.1 Justificaciones de la Arquitectura.

Una posible definición al término arquitectura software, similar a la ofrecida en el capítulo 2, es la que proporcionan los autores Kroll y Kruchten (2003), quienes establecen que este concepto comprende los principales bloques del sistema junto con sus interfaces, los componentes más relevantes y sus respectivas interfaces, y los llamados mecanismos arquitectónicos, soluciones

efectivas para problemas comunes en el diseño de la arquitectura, similares a los patrones de diseño en la orientación a objetos.

Los mismos autores afirman que un buen diseño de la arquitectura de la aplicación resulta crucial para el éxito de un proyecto, por lo que debe ser uno de los principales objetivos al inicio del mismo. La arquitectura software conforma el esqueleto de cualquier sistema software, y es la principal responsable de los atributos de calidad del sistema.

Una arquitectura adecuada, correctamente diseñada, documentada y evaluada, constituye la base para que un proyecto finalice con éxito (Bass y otros, 2003). Esta definición implica que la arquitectura de un sistema define componentes y la interacción existente entre ellos, pero no detalles internos a esos componentes, que se puede considerar que no pertenecen a la arquitectura de la aplicación. Además, la arquitectura del sistema se puede ver desde un número no determinado de perspectivas, todas ellas válidas siempre que valgan para algún fin: análisis, comunicación o comprensión.
Se han desarrollado diversos métodos para diseñar, documentar y evaluar arquitecturas software, como pueden ser el Attribute-Driven Design (ADD), Attribute-Based Architectural Styles (ABAS), Quality Attribute Workshop (QAW), Active Reviews for Intermediate Designs (ARID), Architecture Tradeoff Analysis Method (ATAM), Cost-Benefit Analysis Method (CBAM) y Views and Beyond (V&B). Resultan especialmente interesantes porque no son técnicas independientes, sino que se proporcionan formas de combinar unas con otras (Nord y otros, 2004).

Y continuando con los objetivos de la propuesta, el último que se persigue con la presente tesis doctoral es conseguir que los distintos componentes que forman estos sistemas se conviertan en elementos fácilmente sustituibles e intercambiables, teniendo en cuenta que el uso del término fácilmente no implica que sea una sustitución de tipo "*plug and play*"[3], pero sí poco costosa. Así, por ejemplo, se espera poder utilizar distintas estrategias de tutoría, diferentes maneras de calcular trayectorias dentro del EV y el desarrollo de un tutor inteligente que esté dotados de características muy similares a las humanas.

Con este objetivo en mente, y considerando las definiciones de arquitectura software que se han visto hasta el momento, resulta de especial utilidad plantear la arquitectura que tienen que tener estas aplicaciones, de manera que se identifiquen claramente sus subsistemas y la comunicación entre ellos para, en último término, poder sustituirlos y minimizar el impacto de estas modificaciones tanto en la arquitectura como en el sistema resultante.

---

3 Conocido también como PnP. Tecnología que permite a un dispositivo conectarse y ser usado en una computadora sin configurarlo; para ello el sistema operativo también debe soportar el dispositivo.

3.3.2 ¿Cómo se Evalúa la Arquitectura?

No resulta fácil medir y evaluar las características de una arquitectura software. Esto es debido a varios factores:

1. No existe una identificación clara de las características que debe tener una arquitectura software,
2. No hay una definición unánime o generalizada de los atributos de calidad de las mismas, Determinados atributos de calidad se solapan con otros,
3. La mejora de unos atributos de calidad interfiere con otros
4. Hay factores externos a la arquitectura que también interfieren en las medidas

Sin embargo, lo que sí parece cierto como apunta el trabajo desarrollado por Bass y otros (2003) es que las características con las que se trabaja para evaluar las arquitecturas se refieren a la calidad del software, entendiendo por calidad la adecuación al uso previsto para el sistema software (IEEE, 2002). Algunos de los atributos de calidad más habituales son :

1. Rendimiento: capacidad de ejecución; está relacionado con la velocidad de ejecución y el tiempo de respuesta.
2. Flexibilidad: también llamada modificabilidad, es la facilidad para incorporar cambios.
3. Fiabilidad: tolerancia a fallos, es decir, respuesta correcta del sistema ante situaciones erróneas o inesperadas.
4. Seguridad: resistencia a distintos tipos de ataques, como denegación de servicio o *spoofing*[4].
5. Disponibilidad: capacidad de servir peticiones.
6. Usabilidad: facilidad de uso.

Aunque todos los atributos de calidad mencionados son importantes de cara al resultado final, ya que todos ellos influyen en la calidad final del sistema, las características de los EV, al no ser sistemas críticos, hacen que la fiabilidad, seguridad y disponibilidad sean elementos que no resultan esenciales para los objetivos del presente trabajo.

Por otra parte, la usabilidad, mucho más relacionada con la interacción del usuario con el EV, es un aspecto relevante para este trabajo, pero será abordado fundamentalmente en relación con la interfaz, y para ese punto se un planteamiento a profundidad. En el caso de la propuesta metodológica de la Arquitectura no se revisará detenidamente, porque adicionalmente la funcionalidad propia de los STI para adaptar la tutoría a las características de los estudiantes ya incluye aspectos relacionados con este atributo de calidad en la arquitectura.

4 **Spoofing**, en términos de seguridad de redes hace referencia al uso de técnicas de suplantación de identidad generalmente de forma maliciosos o de investigación.

Y finalmente se podría resaltar que las características más relevantes a considerar en este trabajo son: la modificabilidad, pues es el objetivo principal que queremos conseguir, y el rendimiento, pues presenta interacciones con los restantes atributos de calidad y puede verse afectado al intentar aumentar la modificabilidad, constituyendo además uno de los aspectos clave para la consecución del éxito en el desarrollo de sistemas de basados en EVs.

Para las dos características de interés, se pueden identificar los estímulos externos que las afectan y la manera de medir su impacto. En el caso de la modificabilidad, el estímulo externo serían los cambios en el sistema, y la forma de medir el impacto sería fundamentalmente a través de la complejidad del cambio. Para el caso del rendimiento, los estímulos serían los eventos desde el exterior, y la manera de medir el impacto sería por la velocidad de respuesta. Sin embargo, estas medidas sólo se pueden tomar cuando el diseño se encuentra en un estado avanzado o con el sistema ya implementado, y existen cuestiones adicionales que se deben tener en cuenta.

Como recomiendan Kazman y otros (2000) el análisis y evaluación de la arquitectura puede iniciarse en etapas tempranas del diseño de la misma, siendo los resultados tanto más fiables cuanto mayor detalle tenga ésta. Además, se considera poco práctico producir análisis detallados sobre los resultados de las medidas de los atributos de calidad. En su lugar, resultará más productivo identificar tendencias y puntos sensibles.

Los atributos de calidad, por sí mismos, no son suficientes a la hora de realizar un diseño o de evaluarlo, sino que sería necesario concretarlos a las características del proyecto en desarrollo (Barbacci y otros, 2003). Por ejemplo, en el caso de la modificabilidad, se puede discutir que un sistema sea modificable si permite cambiar con facilidad la interfaz de usuario pero es dependiente del sistema operativo. La respuesta dependería del tipo de modificaciones que se espera que experimente el sistema a lo largo de su existencia.

Por ello, resulta necesario describir situaciones más específicas, junto con la respuesta esperada, a través de la utilización de escenarios, como por ejemplo:

Se desarrolla una calculadora que permite la incorporación de operaciones aritméticas sencillas, y el sistema permite la sustitución del antiguo por el nuevo en menos de dos semanas-hombre. Como se puede observar, este escenario define claramente un estímulo (la existencia de una nueva calculadora que se quiere incorporar) y la respuesta del sistema (sustitución en menos de dos semanas-hombre), lo cual da un criterio claro para poder realizar una evaluación.

Por lo tanto, para estudiar la calidad de la arquitectura, no se van a utilizar medidas numéricas sobre su modificabilidad, ya que estas técnicas de evaluación aún están poco desarrolladas y ofrecen una información limitada sobre la calidad de la arquitectura resultante. Por el contrario, lo que se propone es la realización de una evaluación de la arquitectura software del sistema utilizando un sistema ya probado llamado ATAM (*Architecture Trade-off Analysis Method* [5]) (Clements y otros, 2002b) y tomando como atributos de calidad fundamentales la modificabilidad y el rendimiento.

El resultado sería un informe que refleje si la arquitectura diseñada cumple con los requisitos de modificabilidad y rendimiento, fijados inicialmente, así como una lista de puntos sensibles que pueden ser origen de conflictos cuando se produzcan modificaciones.

3.3.3. ¿Cómo se Utiliza la Arquitectura?

Detrás de los objetivos de este trabajo se encuentra la intención de proporcionar herramientas que sean de utilidad a la hora de desarrollar EVs basados en agentes. Si bien, como ya se ha dicho, la elaboración de una arquitectura para estos sistemas puede constituir una herramienta valiosa allí donde las metodologías de desarrollo no ofrecen ninguna indicación, también es cierto que una arquitectura, por sí sola, constituye una ayuda que se queda coja si carece de un conjunto de indicaciones que muestren cómo utilizarla y cómo es posible hacerla evolucionar.

Por este motivo se ha pensado que, como aportación complementaria de este trabajo, resulta conveniente elaborar un conjunto de recomendaciones metodológicas que sirvan de ayuda a quien pretenda utilizar la arquitectura resultante, ya sea para implementar un Entorno Virtual a partir de ella o para realizar modificaciones sobre la misma y adaptarla a sus propias necesidades.

Estas recomendaciones metodológicas se deben interpretar como criterios a utilizar para modificar la arquitectura propuesta, lo cual permitiría realizar modificaciones sobre ella conservando la integridad conceptual de la misma. De esta manera se conseguiría que la arquitectura no se degrade por efecto de los sucesivos cambios que se realicen sobre ella, lo cual facilitaría su mantenimiento y prolongaría su existencia, sin necesidad de tener que rediseñarla por completo.

---

5 El Método de Análisis de Acuerdos de Arquitectura, es un método de evaluación de arquitectura de software desarrollado e impulsado por el Instituto de Ingeniería de Software, (Software Engineering Institute, SEI), este centra su actividad de evaluación en la interacción entre los diferentes atributos de calidad arquitectónica y basa sus evaluaciones sobre los escenarios desarrollados por los involucrados y un equipo de evaluación.

3.4 Objetivos de la propuesta de Arquitectura

Como se ha podido ver, existen algunas carencias fundamentales en el desarrollo actual de Entornos Virtuales de Enseñanza-Aprendizaje con Tutoría Inteligente (EVEATI) que impiden que estos sistemas puedan alcanzar su madurez.

En primer lugar, la inexistencia de arquitecturas adecuadas provoca que los EVEATIs no sean lo suficientemente modificables. Por mencionar algo, se están realizando notables avances para dotar a los tutores virtuales de rasgos propios de los humanos, como mecanismos de comunicación utilizando lenguaje natural o arquitecturas cognitivas para proporcionarles rasgos de personalidad e incluso estados de ánimo, así como distintos mecanismos de percepción para que sepan lo que pasa a su alrededor. Sin embargo, todavía resulta complejo dotar a los sistemas existentes de estos nuevos avances, ya sea implementándolos desde cero o añadiendo módulos ya implementados.

En segundo lugar, la falta de estándares para el desarrollo de estos sistemas dificulta enormemente el intercambio no sólo de tutores inteligentes y de EVs, sino de los elementos que los componen. Así, resulta bastante complejo adoptar distintas estrategias de tutoría, mecanismos de planificación y cálculo de rutas o utilizar dispositivos de interacción de distinta naturaleza de los que fueron diseñados para funcionar inicialmente con el EV.

Con el este trabajo se pretende definir una arquitectura de software para EVEATIs que facilite la modificabilidad de los mismos y la reutilización de sus componentes. Para ello, es necesario tener en cuenta qué elementos pueden formar parte de un EVEATI y cuáles de estos interesa que puedan ser sujetos a modificación, ya sea ampliación, sustitución o incluso eliminación. Entre estos elementos deben encontrarse:

Entorno Gráfico e Interfaz Gráfica: es uno de los elementos cuya sustitución se debe poder llevar a cabo con facilidad. Debido a la complejidad de su desarrollo, resulta difícil de implementar para equipos no especializados en ello. Por este motivo, disponer de un EV externo que pueda integrarse con el software propio podría dar como resultado un sistema mejor construido.

Módulo de Tutoría: al trabajar con sistemas de enseñanza, uno de los elementos que es previsible que evolucionen con frecuencia son precisamente los tutores, ya sea incorporando nuevas estrategias de tutoría o haciendo uso de nuevos elementos para valorar las aptitudes de un alumno.

Modelo del Estudiante: está muy relacionado con el anterior, ya que cuanto más rico y completo sea el modelo que se realiza del estudiante, mayores y mejores herramientas tendrá el tutor para trabajar con los alumnos. Por esta razón, también es probable que sufra ampliaciones a medida que se avance en el estudio de las actitudes y aptitudes de los estudiantes.

Módulo Experto: maneja el conocimiento que posee el STI sobre el dominio en el cual se realiza el entrenamiento. Resulta, por tanto, necesario para ofrecer explicaciones acerca de las actividades a realizar, para decidir la adecuación de las acciones llevadas a cabo por los estudiantes y para contestar sus preguntas.

Planificador: el papel de un planificador resulta básico para poder plantear ejercicios al alumno, responder sus preguntas o estudiar las soluciones que los alumnos dan a los ejercicios propuestos. Los distintos dominios de aplicación de un EVEATI pueden requerir diferentes tipos de planificación, incluyendo la posibilidad de plantear tareas paralelas, realizar operaciones básicas o planificar hacia delante o hacia atrás.

También puede ser conveniente incorporar algoritmos de planificación más eficientes o utilizar planificación cooperativa. Así, éste es otro de los elementos susceptible de variar dependiendo del dominio de aplicación y de los avances en el campo de la planificación.

Planificador de Rutas: unida a la planificación de tareas, la planificación de rutas se hace necesaria para que el sistema pueda realizar planes en los que intervengan desplazamientos dentro del EV, así como para ejercicios en los que, por ejemplo, haga falta seguir la ruta más corta entre dos puntos o aquella que cumpla con alguna otra restricción. Dependiendo de la estructura del EV utilizado y de las necesidades, es posible utilizar distintos métodos de cálculo de rutas, como el algoritmo de Dijkstra (1959), el A* (Nilsson, 1971) o diagramas de Voronoi (1907), por mencionar algunos de los más utilizados y difundidos. Por tanto, también es necesario que este elemento pueda ser sustituido con facilidad.

Simulación: el control de la simulación se requiere cuando es necesario reproducir el comportamiento de algún elemento presente en el EV tras la acción de un estudiante. Así, por ejemplo, teclear una clave de acceso puede requerir la comprobación de la corrección de la clave y la anotación del suceso en el sistema simulado. Atendiendo al dominio de aplicación y las características de lo que se quiera simular sería necesario

disponer de distintos mecanismos de simulación, basados, por ejemplo, en reglas o en autómatas finitos. Por tanto, es otro de los elementos susceptibles de ser sustituido con relativa frecuencia, en función de las necesidades.

Para lograr que los elementos anteriores se puedan sustituir fácilmente, resulta fundamental que la arquitectura de software del sistema sea lo suficientemente flexible como para permitir el intercambio de agentes, así como la presencia o ausencia de algún agente en función de las necesidades de tutoría de una determinada actividad.

También resultaría de utilidad la existencia de criterios que guíen el desarrollo de sistemas con características análogas al considerado, tanto para su diseño como para la posterior implementación. Por ello, se elaborará un conjunto de recomendaciones metodológicas que den soporte a la utilización y modificación de la arquitectura propuesta.

El cumplimiento de los objetivos expuestos a lo largo de esta sección se llevaría a cabo mediante las aportaciones científicas y tecnológicas que se enumeran a continuación:

Principales aportaciones:

Diseñar una Arquitectura de software basada en agentes pedagógicos para Entornos Virtuales con Tutoría Inteligente que permita sustituir con facilidad los siguientes elementos: entorno gráfico; módulo de tutoría; modelo del estudiante; módulo experto; planificador; planificador de rutas; y simulador.

Plantear una serie de recomendaciones metodológicas para la construcción de Entornos Virtuales con Tutoría Inteligente utilizando la arquitectura basada en agentes descrita anteriormente.

Plantear las recomendaciones metodológicas para el diseño de la interfaz del EVEATI, así como la interfaz y modelado del Agente.

Implementar una infraestructura básica con una célula simple, que utilice la arquitectura diseñada y presente el modelado del interfaz gráfica de usuario y el agente y que sirva como punto de partida para poder implementar EVEATIs a partir de ella.

3.5 Propuesta de Arquitectura

Como se ha mencionado en los capítulos anteriores, uno de los principales objetivos es diseñar

una arquitectura de software para Entornos Virtuales de Enseñanza-Aprendizaje con Tutoría Inteligente (EVEATI).

En el presente apartado se describe, sin entrar en excesivos detalles, una propuesta inicial de arquitectura con estructura jerárquica, obtenida a partir de la descomposición de los elementos que conforman la arquitectura tradicional de un Sistema de Tutoría Inteligente (STI) y en otros elementos más especializados para dar paso a mayor sobre el planteamiento inicial.
Como se podrá ver al final del capítulo, la arquitectura resultante presenta solo algunas propiedades respecto a la facilidad de modificación que se deseaba obtener. Para diseñar esta arquitectura preliminar, se utilizó un enfoque organizacional propuesto por diversas metodologías de desarrollo de software orientado a agentes, siendo la metodología Gaia (Zambonelli y otros, 2003) el referente principal para el diseño de la arquitectura que se describe a continuación.

Gaia parte de una estructura inicial sobre la que se trabaja para obtener el resultado final. Esta estructura inicial es la de la organización (metáfora organizacional) para la que se desarrolla el sistema. En el caso que nos ocupa, se ha decidido adoptar como estructura de partida la de un STI tradicional (Sleeman y Brown, 1982; Wenger, 1987), debido a la inexistencia de otra organización que tomar como referencia y también a la inexperiencia previa de la autora con estos sistemas.

Por esta razón, se comenzará presentando una visión de los Sistemas Inteligentes de Tutoría, para continuar desarrollando la arquitectura de un EVEATI a partir de la de un STI, terminando con el diseño de una arquitectura conceptual del sistema basada en la propuesta de Hofmeister y otros (2000).

3.5.1 Arquitectura de Alto Nivel

Con la aparición de la Inteligencia Artificial (IA) a principios de los 60 y su aplicación al desarrollo de sistemas de enseñanza se acuñó el término Sistema de Tutoria Inteligente (STI). Un STI, según Sleeman y Brown (1982), tiene como objetivo actuar como un tutor humano inteligente, de manera que sea capaz de orientar y enseñar a un estudiante en el proceso de aprendizaje de una materia dada, detectar los errores que el estudiante comete, tratar de determinar el punto en que éste falla para corregirlo y aclarar confusiones o dudas que se le presenten, considerando las peculiaridades de cada estudiante y permitiendo, de esta forma, una enseñanza más individualizada.

Para poder alcanzar dicho objetivo, durante el desarrollo de un sistema de este tipo deben abordarse cuestiones como la representación en el sistema de la materia objeto de estudio, la representación del conocimiento acerca del estudiante, la estrategia de comunicación con el estudiante, y las estrategias de tutoría a seguir. La arquitectura más general de un STI (Wenger, 1987), que puede verse en la Figura 3.3, comprende cuatro componentes principales, cada uno de los cuales viene a dar respuesta a una de las cuestiones anteriores:

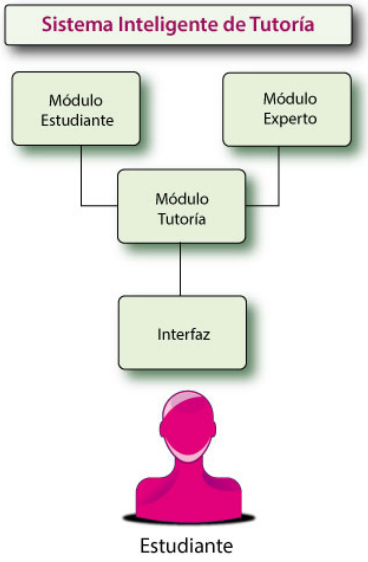


Figura 3.3: Arquitectura tradicional del un STI

La manera más clásica de construir un STI utilizando esta arquitectura se puede asimilar con una típica estructura en tres capas, donde el Módulo de Comunicación hace las veces de interfaz de usuario, el Módulo de Tutoría contiene la lógica de la aplicación y los Módulos Experto y del Estudiante serían ambas bases de datos, una con el conocimiento del experto y otra con la información que se posee sobre el estudiante.

Esta arquitectura, aunque suficiente para un STI tradicional, no se ajusta a los requisitos de un EVEATI por varios motivos:

Un EVEATI debe servir para poder enseñar o entrenar a un grupo de estudiantes al mismo tiempo, y no a uno solo, y debe ser capaz de adaptarse a las necesidades de cada alumno, lo cual no está contemplado en la arquitectura de la Figura 3.4.

El Módulo del Estudiante debe modelar el conocimiento de cada estudiante, pero también el del grupo completo. El estudiante no queda fuera de los límites del sistema, ya que la interacción con el EV se realiza a través de un avatar que se encuentra dentro del propio EV. Además, cada estudiante tiene una visión distinta del entorno dependiendo del punto en el que se encuentre.

Aunque sería muy intuitivo considerar que el EV forma parte de la interfaz gráfica de usuario, existe una razón que impide hacer esta consideración. Cuando se utiliza una interfaz gráfica clásica, la interacción con la aplicación se realiza a través de menús, botones y controles similares, pero el fin no es aprender a utilizar esos controles, sino que son un medio para comunicarse con la aplicación. Por el contrario, cuando se utiliza un EV, el objetivo que se persigue sí es aprender a manipular lo que aparece dentro de él. Por este motivo, es necesario que el STI tenga conocimiento de lo que ocurre dentro del EV, su estado y las posibilidades de interacción.

Teniendo en cuenta las diferencias anteriores, si se desea que un STI interactúe con un EV multiusuario es necesario de entrada modificar la concepción del Sistema y pasar de un sistema inteligente básico a el fin último que es desarrollar una tutoría inteligente y extender la arquitectura clásica para incorporar las nuevas características que de él se requieren. Los cambios que son necesarios llevar a cabo son:

Ampliar el Módulo de Comunicación para que de soporte a varios estudiantes de manera simultánea y que permita y exiga una interfaz más robusta, amigable, intuitiva y capaz de presentar un entorno completo.

Ampliar el Módulo del Estudiante para que mantenga un modelo para cada estudiante, así como del grupo completo.

Ampliar el Módulo Experto para incluir el conocimiento de lo que ocurre en la simulación cuando el estudiante realiza alguna acción.

Incluir un nuevo módulo, el Módulo Global o del Mundo, para mantener información geométrica y semántica acerca del estado de los objetos y habitantes del EV.

Este nuevo módulo debe comunicarse con el de Tutoría, para que éste a su vez pueda tener acceso al estado global o del mundo para, por ejemplo, indicar dónde se ubica un objeto que el estudiante no es capaz de encontrar, y con el de Comunicación, para poder actualizar el estado del mundo en respuesta a las acciones de los estudiantes.

Con estas modificaciones, la arquitectura resultante es la que se muestra en la Figura 3.4.

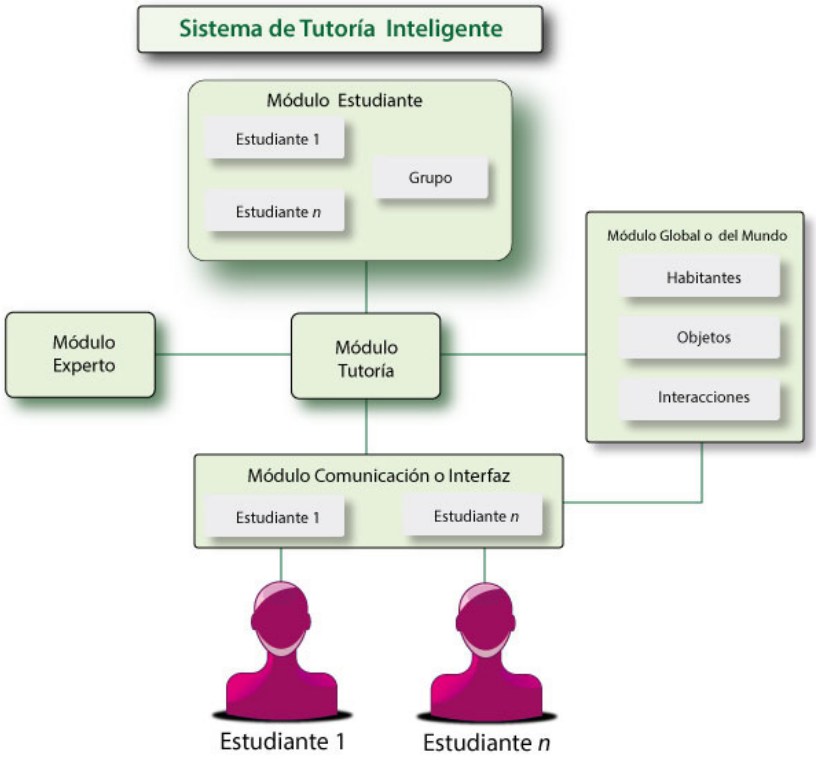


Figura 3.4: Sistema de Tutoría Inteligente con Arquitectura Ampliada

3.5.2 Arquitectura Conceptual con Agentes

Aunque para el desarrollo de la arquitectura conceptual de un sistema software no es necesario especificar la forma en que se construirá (e.g. utilizando objetos o agentes), en este punto se considera pertinente y necesario introducir los agentes en el diseño arquitectónico. Esto se debe a que sus características particulares pueden obligar a la modificación de la arquitectura si su utilización se deja para más adelante.

Para el diseño de la arquitectura, los distintos autores ofrecen diferentes alternativas, que acaban reduciéndose a dos junto con las distintas variantes que pueden surgir entre ellas.

Zambonelli (2003), a través de la metáfora de la organización, sugiere tanto una estructura jerárquica como una de pares, en la que todos los componentes se sitúan al mismo nivel. En la propuesta de Bass (2001) hay una fuerte inclinación por el modelo jerárquico, a partir de una descomposición y refinamiento iterativo de cada elemento de la arquitectura, de manera casi independiente al resto de elementos.

Considerando lo expuesto por ambos autores, y dada la clara separación de responsabilidades de los módulos que forman un STI, se decidió adoptar un enfoque jerárquico para el diseño de la arquitectura y mantener la modificación para hacer de este sistema un sistema dónde si se verifique la tutoría y seguimiento de los estudiantes de manera inteligente.

3.5.1.1 Agentes de Primer Nivel

Estos agentes, como se puede ver en la Figura 3.5, se derivan directamente de la arquitectura de alto nivel, y corresponden a los módulos del STI más el Módulo Mundo, que queda representado por el Agente Global o del Mundo. Las responsabilidades de los agentes son las que se describen a continuación:

Agente de Comunicación o Interfaz.

El agente de comunicación es el encargado de gestionar todo el intercambio de información existente entre el sistema y el exterior. De esta manera, de cara al exterior del STI, debe encargarse de recibir las acciones que realizan los estudiantes en el EV y cualquier otro evento que pueda suceder en él y transmitírselos, según corresponda, al agente de tutoría y al agente global. También debe gestionar las conexiones de los distintos EVs para poder informar al agente de tutoría del número de estudiantes conectados.

Finalmente, debe enviar las respuestas del STI al EV en forma de acciones, explicaciones, respuestas o cualquier otro elemento que pueda ser tratado en el EV.

El STI, como se ha mencionado, es el encargado de distribuir los eventos del EV al agente correspondiente, ya sea al agente de tutoría, si son acciones o preguntas del estudiante, o al agente mundo, si son cambios en la configuración de los objetos o habitantes presentes en el EV. En sentido contrario, debe recibir las instrucciones que el agente de tutoría desea comunicar a los estudiantes y enviarlas al alumno correspondiente.

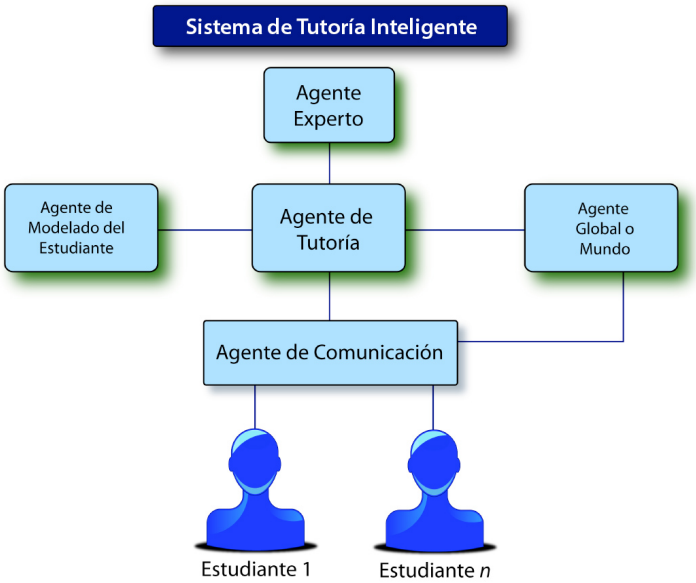


Figura 3.5: Arquitectura ampliada con Agentes

Agente de Tutoría

Este agente constituye la pieza central del STI, y es responsable de muchas de las decisiones que afectan al proceso de enseñanza. Así, en última instancia, este agente es el que decide qué se hace ante las distintas acciones que realicen los estudiantes, así funcionará bajo los siguientes criterios: dejándoles continuar cuando realizan alguna acción incorrecta, dándoles algún tipo de pista o permitiéndoles cambiar el nivel de detalle de las explicaciones y las respuestas a sus preguntas. Esto se realiza adoptando diferentes estrategias de tutoría, estrategias que se pueden apoyar en la información que se posee sobre los alumnos y que están en posesión del agente de modelado del estudiante.

La comunicación entre este agente y los que forman el resto del sistema es bastante amplia, debido a la relevancia de su papel. Del agente de comunicación recibe las acciones y preguntas de los estudiantes, y le devuelve respuestas a las preguntas, pistas, explicaciones, y eventos que deben suceder en el EV en respuesta a las acciones de los estudiantes.

Del agente experto recibe la forma de resolver las actividades que sirven de entrenamiento al alumno, así como el conocimiento que se le debe transmitir al alumno en forma de respuestas a algunas preguntas, explicaciones y pistas. Se le proporcionan las acciones que realiza el alumno para poder actualizar la solución al ejercicio planteado.

Agente de Modelado del Estudiante

El agente de modelado del estudiante es, posiblemente, el agente más complejo de encajar en la arquitectura, porque su función puede variar enormemente tanto en tamaño como en complejidad. En la versión más simple, no hace más que recibir las acciones que realiza el estudiante desde el agente de tutoría y las almacena para mantener una traza de cada sesión. En la versión más compleja, sirve de punto de apoyo de la estrategia de tutoría, de manera que realiza un diagnóstico cognitivo del alumno: mantiene un perfil de su estilo de aprendizaje, su atención, estado de ánimo o el conocimiento que el STI asume que el estudiante posee, en función de los cuales se pueden seleccionar distintas estrategias de tutoría y distintas respuestas ante una acción del alumno, así como el nivel de detalle de una explicación.

Recibe del agente de tutoría la información necesaria para mantener el modelo del estudiante, como las acciones realizadas, sus movimientos por el EV o hacia dónde mira, y le proporciona los datos necesarios para tomar decisiones de tutoría, como su nivel o facilidad de aprendizaje.

Además, puesto que posee la información relativa a todos los estudiantes, también debe realizar un modelo del grupo en su conjunto, de manera que se pueda medir su capacidad organizativa, su coordinación o la corrección de la distribución de funciones.

Agente Experto

Este agente posee el conocimiento sobre la materia que se quiere enseñar al alumno. En el caso del entrenamiento procedimental, es capaz de plantear situaciones que el alumno debe resolver y de actualizarlas en función de las acciones del estudiante para replantear la solución en caso necesario. Por tanto, debe tener capacidades de planificación y replanificación, que serán aprovechadas por el agente de tutoría para utilizar estrategias más o menos flexibles. Así, se podrán plantear ejercicios con objetivos similares partiendo de situaciones distintas o, incluso, plantear ejercicios en respuesta a una petición del alumno.

Por otro lado, posee el conocimiento necesario para simular el comportamiento de los objetos que aparecen en el EV, de forma que si se requiere la manipulación por parte del estudiante se pueda controlar el comportamiento de los mismos.

Proporciona al agente de tutoría el planteamiento de los ejercicios a resolver, así como la información para determinar si una acción es o no correcta. De él recibe peticiones para crear un nuevo ejercicio, las acciones realizadas por el estudiante para que las valide y peticiones para replantear la solución de un ejercicio si las acciones del estudiante se dirigen por un camino distinto al propuesto por el experto.

Agente Global o del Mundo

Este agente se encarga de mantener una representación del EV que puedan utilizar los demás agentes. Posee información acerca de los objetos presentes en el EV, como su posición, su orientación y su geometría. También posee conocimiento sobre la utilidad de los distintos objetos, como las acciones que se pueden realizar con ellos y sus condiciones de uso. Además, maneja información sobre los habitantes del EV, como su posición, la orientación o los objetos que poseen.

Finalmente, también conoce la estructura del EV, de manera que es capaz de calcular rutas entre dos puntos distintos del mismo, evitando colisionar con los objetos que se encuentran presentes en él.

Recibe toda la información geométrica a través del agente de comunicación, y le proporciona al agente de tutoría la información necesaria acerca del estado del mundo para que la pueda utilizar en el proceso de tutoría.

3.5.1.2 Agentes de Segundo Nivel

Estos agentes surgen para hacerse cargo de las responsabilidades de los agentes descritos en la sección anterior, quienes se encargarán de coordinarlos y de canalizar las comunicaciones entre ellos. Así, la existencia de los agentes de primer nivel introduce un nivel que contribuye a aumentar la flexibilidad de la arquitectura (Bass y otros, 2000), ya que facilita la introducción de nuevos agentes subordinados a ellos y la eliminación de alguno de los existentes, atendiendo a la funcionalidad requerida.

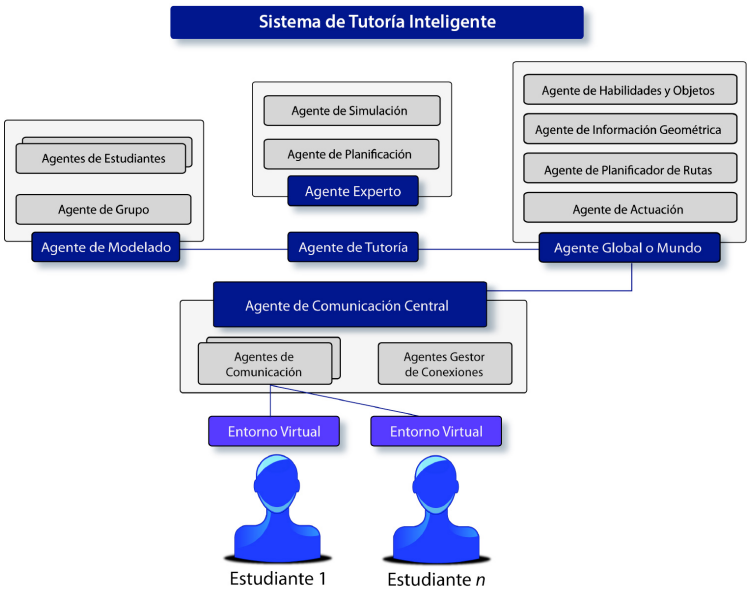


Figura 3-6: Arquitectura Basada en Agentes

Por otra parte, los agentes de segundo nivel se ocuparán de las responsabilidades de carácter variable y para las cuales se pretende dotar de flexibilidad a la arquitectura. Además, se ha adoptado un enfoque cooperativo para realizar la planificación y replanificación de las soluciones a los ejercicios. Dicho enfoque permite la incorporación de nuevos agentes con conocimiento distinto al de los agentes existentes, lo cual facilita la adaptación a nuevos dominios. Todos los agentes que participen en la planificación se van a comunicar por medio de una pizarra, lo cual también contribuye a la modificabilidad del sistema, ya que la pizarra permite la incorporación y eliminación de agentes sin que el resto deba sufrir ningún tipo de modificación.

A continuación se describen las responsabilidades de cada uno de los agentes de segundo nivel, así como las de los agentes de primer nivel, tanto las nuevas como las que ya poseían.

Agente de Comunicación Central

Este agente es el equivalente al agente de comunicación del nivel anterior. Sin embargo, al añadir como subordinados a los agentes de comunicación y al agente gestor de conexiones, este agente se mantiene sólo por motivos de flexibilidad. Sus responsabilidades quedan prácticamente reducidas a cero, salvo por las recién adquiridas como mediador.

Su papel principal en la arquitectura es, por tanto, el de mantener la modificabilidad. Esto permitirá, en caso de que sea necesario, eliminar los agentes de comunicación sin afectar al resto de los

agentes, lo cual puede ser conveniente, por ejemplo, para mantener el STI funcionando sin EVs conectados a él. También facilitará la inclusión de nuevos agentes, lo que puede ser útil si se precisa manejar algún tipo de información especial relativa, por ejemplo, a los dispositivos de realidad virtual, ya que esta información podría no ser proporcionada directamente por el EV.

Aunque puede constituir un cuello de botella, el hecho de que no realice ningún tipo de procesamiento complejo, puesto que sólo redirige mensajes entre el exterior y el interior del STI, hace poco probable que vaya a generar problemas de rendimiento.

Agentes de Comunicación

Estos agentes se encargan de la comunicación entre el STI y cada uno de los EVs. En ejecución habrá tantos agentes de comunicación como estudiantes hayan iniciado una sesión de entrenamiento, y cada agente será responsable de recibir la información de su EV correspondiente, traducirla al formato que maneje el resto de los agentes y enviársela al agente de comunicación central para que éste, a su vez, se la envíe al agente adecuado.

La labor de traducción se le asigna a estos agentes por dos motivos. El primero es que de esa forma se puede realizar simultáneamente la traducción de los mensajes enviados desde distintos EVs, lo cual redunda en un mejor rendimiento de la aplicación (aunque el hecho de tener que hacer la traducción es ya una carga para el STI). El segundo es que permite cambiar el formato de los mensajes entre el EV y el STI sin necesidad de modificar ningún otro agente, lo cual va en beneficio de la modificabilidad del sistema. Por supuesto, sería mejor que la traducción se realizase en el EV, antes de enviar la información, pero ya que uno de los propósitos es poder interconectar diferentes EVs, no se puede asegurar que esto siempre vaya a ser así.

De igual manera, estos agentes se deben encargar de realizar la labor contraria: recibir información del agente de comunicación central, traducirla y enviarla al EV del cual son responsables.

Agente Gestor de Conexiones

La labor de este agente es, sobre todo, de soporte. Así, es el encargado de recibir las conexiones entrantes de cada alumno que desea unirse y crear un agente de comunicación que se encargue del intercambio de información entre el nuevo EV y el STI.

También se encarga de informar al agente de tutoría, a través del agente de comunicación central, del número de estudiantes que se encuentran conectados, para que se pueda iniciar el entrenamiento cuando haya suficientes estudiantes conectados.

Agente de Tutoría

El agente de tutoría es el único que, de momento, no tiene asignado ningún agente subordinado, por lo que conserva todas las responsabilidades que se le asignaron en la sección anterior.

Aunque se ha manejado la posibilidad de incorporar un agente de tutoría individualizado para cada estudiante, esto podría provocar problemas de coordinación y visualización, por lo que, de momento, se ha preferido mantener un único agente de tutoría común para todo el grupo.

También se ha estudiado la posibilidad de incluir agentes que pongan en práctica distintas estrategias de tutoría. Sin embargo, esto presenta varios problemas que no aconsejan seguir ese camino.

A pesar de su aparente utilidad desde el punto de vista pedagógico, requeriría establecer un proceso para decidir qué estrategia de tutoría aplicar en cada momento. Este proceso de negociación entre distintos agentes de estrategia de tutoría podría conducir a un cambio continuo, lo cual podría llegar a confundir al alumno.

Desde el punto de vista computacional, la negociación para realizar el cambio de estrategia puede suponer un costo alto si se realiza con demasiada frecuencia. Además, dependiendo de las estrategias de tutoría que deban convivir, puede suponer un incremento significativo de la complejidad del agente de modelado de estudiantes. En cualquier caso, la arquitectura permitiría dotar al sistema de ambos mecanismos.

Agente de Modelado de Estudiantes

Al igual que sucedía con el agente de comunicación central, el agente de modelado de estudiantes corresponde al agente de estudiante descrito en la sección anterior, cuyas responsabilidades se han delegado en sus agentes subordinados, por lo que su papel ahora es el de mediador por motivos de flexibilidad. Así, se encargará de redirigir la información y las peticiones que le transmita el agente de tutoría al agente adecuado, y enviar las respuestas de estos agentes al de tutoría. En caso necesario deberá encargarse de realizar labores de traducción o interpretación.

Agentes de Estudiante

De igual manera que los agentes de comunicación, los agentes de estudiante se van a encargar de realizar un modelo de cada estudiante conectado al sistema. En ejecución existirá un agente de estudiante por cada alumno conectado al sistema.

El modelo de estudiante puede incluir información acerca de la forma de aprendizaje del alumno, de su personalidad o de su atención. También incluirá las acciones realizadas por el alumno a lo largo de la sesión, así como indicaciones de su corrección o incorrección y cualquier otro tipo de evaluación que pueda ser de interés para el desarrollo de la estrategia de tutoría seleccionada.

La evaluación de la atención del alumno puede necesitar información en poder del agente global o del mundo, como la dirección de la mirada del estudiante. De igual manera, el diagnóstico de por qué el estudiante ha cometido un error necesita información en posesión del agente experto. Por tanto, puede existir flujo de información entre los agentes de estudiante y el agente experto y el agente global o del mundo.

Por el momento se propone que este tráfico de información se lleve a cabo a través del agente de tutoría, pues estas acciones se realizarán a petición suya. Si llegase a suponer un tráfico excesivo de información y un cuello de botella, habrá que diseñar una comunicación directa. Esto supondría una merma en la modificabilidad del sistema, al tiempo que se da un aumento en el rendimiento de la aplicación, ya que la comunicación no pasaría por intermediarios.

De manera adicional, la evaluación podría realizarse sin esperar la petición del agente de tutoría, por lo que éste tampoco tendría que esperar a que sea elaborada la información que necesite y el rendimiento de la aplicación también se vería incrementado.

Agente de Grupo

Este agente es el encargado de realizar la misma labor que los agentes descritos anteriormente, pero para el grupo en conjunto, sólo que al contrario de los anteriores, no utilizará la información proveniente del exterior, sino que hará uso de la que ya ha sido elaborada por los agentes de estudiante.

El modelo a realizar por este agente incluye aspectos de coordinación entre los miembros del grupo, colaboración entre ellos o posibles fallos en algún aspecto concreto del trabajo en equipo.

Agente Experto

El agente experto es otro de los que han visto sus responsabilidades reducidas a un papel mediador, en este caso entre el agente de tutoría y el de planificación. Al contrario que en el caso del agente de comunicación central o el de modelado de estudiante, carece de un grupo de agentes a su cargo, por lo que su carga de trabajo no es muy grande, ni en cantidad ni en complejidad. Y tomando en consideración esto, se puede decir que quizá sea prospecto a

desaparecer de la arquitectura, de momento se ha mantenido por motivos de homogeneidad.

Agente de Simulación

El agente de simulación está a cargo del conocimiento del dominio relacionado con los efectos de las acciones del usuario sobre algunos objetos. En concreto, tal como indica su nombre, está encargado de simular el comportamiento de determinados objetos del EV, especialmente cuando un alumno actúa sobre ellos. Por ejemplo, si se quisiese enseñar a un estudiante el procedimiento a seguir para aspirar una alfombra, el agente de simulación estaría encargado de simular lo que sucede en la aspiradora una vez que ésta se pone en marcha: los distintos estados por los que pasa y los efectos que tienen sobre la ropa.

Este agente es uno de los mejores ejemplos de la necesidad de modificabilidad en la arquitectura, ya que dependiendo del dominio de aplicación puede suceder que este agente no sea necesario, funcione con un sistema basado en reglas, con un autómata finito determinista o por cualquier otro medio.

Agente de Planificación

El agente de planificación constituye unos de los puntos críticos de la arquitectura, y es uno de los principales motivos por los que es deseable que el sistema sea altamente modificable.

Este agente tiene uno de los papeles más relevantes dentro de un EVEATI, ya que es responsabilidad suya plantear la solución a un ejercicio e ir encontrando soluciones alternativas a medida que las acciones de los estudiantes no se ajustan a las especificadas en el plan inicial.

El motivo de la importancia de este agente es doble. Por un lado, el proceso de planificación y replanificación es bastante laborioso, y constituyeuno de los elementos que más afecta al rendimiento del sistema. Por otro lado, el hecho de haber optado por una planificación cooperativa hace que haya más de un agente involucrado en la misma, lo cual puede afectar a la modificabilidad de la arquitectura. Esto es debido tanto a que, dependiendo del dominio, pueden ser necesarias distintas técnicas de planificación y distinto tipo de representación de la información, y también a que la necesidad de comunicación puede dar lugar a acoplamientos no deseados.

Para evitar el problema que supone la planificación y replanificación la solución consiste en utilizar planificadores eficientes en los que la replanificación no suponga partir desde cero. Este es uno de los motivos que hace que sea necesario facilitar el cambio de este agente.

El otro, como se ha mencionado anteriormente, es el hecho de que distintos dominios de aplicación pueden necesitar distintos tipos de planificación. Para evitar el acoplamiento entre los distintos agentes que toman parte en la planificación se ha optado por realizar la comunicación a través de una pizarra. De esta forma, el agente de planificación se encarga de realizar su función con la información que le proporcionan el resto de los agentes, pero no necesita saber cuántos ni cuáles son. Por otro lado, el resto de agentes que toman parte en la planificación no necesitan saber quién la realiza ni quién más participa. Simplemente, si encuentran en la pizarra alguna situación que se puede satisfacer con el conocimiento que poseen, lo escriben en ella y se olvidan de cualquier otro detalle.

En cuanto al problema de representación de la información, no es algo que se pueda resolver con la arquitectura del sistema, sino que dependerá de una buena implementación de la misma.

En la arquitectura que se plantea, la planificación la realiza el agente de planificación junto con el de simulación, el de actuación y el planificador de rutas. Con este enfoque resultaría sencillo hacer que otros agentes también participacen en ella.

Agente Global o del Mundo

Como ocurre con la mayoría de los agentes descritos en la sección anterior el agente global o del mundo queda reducido a labores de intermediario para establecer un punto que dote a la arquitectura de mayor flexibilidad. Al contrario de lo que sucedería con el agente experto, este agente tiene la suficiente entidad como para que no surjan dudas acerca de su necesidad, ya que este agente sí tiene como misión la supervisión de un grupo de agentes.

Por otra parte, es susceptible de recibir nuevos agentes subordinados a él, ya que los avances en modelos de percepción, personalidad y comportamiento para agentes inteligentes hacen bastante probable que haya que incorporar nuevas características al funcionamiento del tutor virtual dentro del mundo.

Este agente va a interactuar, por tanto, con el agente de comunicación central para recibir las actualizaciones sobre el estado del EV, las cuales trasladará a sus agentes subordinados. También se comunica con el agente de tutoría para proporcionar la información que éste necesite para dar explicaciones al alumno o contestar sus preguntas.

Agente de Habitantes y Objetos

El agente de habitantes y objetos posee información semántica acerca de los objetos presentes en el EV, de forma que este conocimiento es utilizado para indicarle a los alumnos qué es un objeto o para qué se puede utilizar. Esta información se la proporcionará al agente de tutoría a través del agente global o mundo.

Agente de Información Geométrica

El agente de información geométrica mantiene un mapa geométrico del EV, incluyendo habitaciones, objetos y personajes. Aunque toda esta información podría recuperarse del EV, resulta más eficiente mantener una réplica en el STI, la cual se mantendrá actualizada a medida que lleguen cambios procedentes de cualquiera de los EVs que se encuentren conectados al STI.

Este agente maneja información acerca de la situación y orientación de todos los habitantes y objetos presentes en el EV, así como de los objetos que se encuentran en posesión de cada uno de los habitantes del EV. De esta manera, por ejemplo, proporciona información acerca de dónde se encuentra un objeto en el EV para poder indicárselo al alumno.

Agente de Actuación

El agente de actuación es uno de los agentes más importantes de cara al proceso de planificación. Este agente maneja información acerca de las posibles acciones que se pueden realizar con los objetos presentes en el EV, de manera que, con la configuración actual, es el agente que más contribuye a la planificación.

Por otro lado, la información que maneja este agente es de utilidad para indicarle al alumno las consecuencias de realizar una determinada acción dentro del EV.

Agente Planificador de Rutas

El agente planificador de rutas se encarga de calcular los posibles caminos que existen para llegar de un punto a otro del EV, evitando chocar con obstáculos. Este cálculo se realiza, fundamentalmente, aplicando un algoritmo de cálculo de trayectorias como A* al mapa del EV realizado por el agente de información geométrica.

Los caminos calculados por este agente se pueden utilizar, por ejemplo, para mover agentes virtuales dentro del EV o, en los casos en los que sea aplicable, para comparar la ruta obtenida con la que ha seguido el estudiante, de manera que se pueda ver si ha seguido el camino más conveniente o no.

3.6 Análisis de la Arquitectura

Como se ha mencionado al comienzo del capítulo, la arquitectura aquí descrita presenta una serie de problemas que la hacen poco adecuada para los fines que se persiguen. En concreto, algunas partes de la arquitectura son excesivamente rígidas para la modificabilidad que se quiere obtener. En otros casos, las decisiones que se han tomado, aunque razonables en su momento, parecen no conducir hacia los fines deseados.

A continuación se describen los dos grandes motivos que han llevado a descartar o más bien mejorar y reacondicionar esta arquitectura, junto con unas reflexiones sobre el trabajo que se debe mejorar y las modificaciones que deben desarrollarse.

3.6.1 Efectos de la Planificación en la Arquitectura

Una de las responsabilidades más complejas que soporta la arquitectura descrita en este capítulo es la relativa a la planificación. Uno de los requisitos que se impusieron es que se deseaba que el sistema de planificación implementado fuese muy flexible y admitiese la adición de nuevos operadores de manera dinámica, esto con el fin de que cualquiera que la tomara como punto de referencia, no importando la temática o contenido, pudiera partir de la modificabilidad y flexibilidad como principales atributos. Por esa razón se decidió utilizar una planificación cooperativa en la que distintos agentes pudiesen aportar operadores relativos a lo que exclusivamente ellos saben hacer. Esto permite la introducción de nuevos agentes, incluso de manera dinámica, que aportaran nuevos operadores que ayudaran a realizar un plan donde antes no se podría, o a realizar un plan más eficiente que el que se conseguiría sin su ayuda.

En este sentido, se debe decir que la arquitectura presenta un alto grado de modificabilidad, ya que permite la introducción de nuevos agentes y nuevos operadores para realizar la planificación de manera dinámica, sin modificar el resto del sistema. Sin embargo, esta característica que incluye tanta flexibilidad impone también fuertes restricciones. En primer lugar, involucra a muchos de los agentes del sistema en la planificación. Como cada agente es responsable de sus

propios operadores, los cambios que se realicen en el planificador que afecten el formato de los operadores pueden afectar a todos los agentes que formen parte del proceso de planificación.

Por otra parte, el hecho de realizar la planificación usando una pizarra que utilicen todos los agentes se salta en cierta manera la filosofía de utilizar una estructura jerárquica en la que la comunicación entre agentes se realice a través de los agentes supervisores. Uno de los aspectos fundamentales en una arquitectura es que exista la denominada integridad conceptual (Bass y otros, 2003), que consiste en mantener una homogeneidad de criterios en el diseño de la arquitectura. El hecho de introducir soluciones distintas para cada una de las funciones que debe cumplir la arquitectura redunda en una mayor complejidad de la misma y, a la larga, en una mayor dificultad para entenderla, manejarla y modificarla.

Además, uno de los factores que no se tuvo en cuenta es que en algún momento podría ser necesario sustituir el tipo de planificador empleado en la realización de la arquitectura. El planificador utilizado en primera instancia, basado en STRIPS[6], resultó ser bastante sencillo, y no proporcionaba algunas capacidades que se hacía necesario utilizar. Llegado el momento, el problema que surgió es que no fue posible encontrar un planificador que admitiera seguir utilizando la planificación cooperativa como se hacía hasta el momento. El resultado fue que, al cambiar el planificador, hubo que desmontar gran parte del sistema para volver a rediseñarlo, lo cual demostró que la arquitectura no estaba preparada para sufrir esa modificación.

6 El planificador STRIPS (STanford Research Institute Problem Solver), surge a principios de los 70. Se trata de un Planificador automático usado en aplicaciones de robótica. Aunque hoy día ya casi no se utiliza como tal, lo que se ha popularizao es el uso de STRIPS como un lenguaje clásico para la planificación. Utiliza un formalismo basado en lógica para representar estados y operadores(Fikes y Nilsson 1971).

Reflexiones

Existe una tendencia natural a intentar los diseños, más que a evaluarlos, para emitir dictámenes acerca de si son buenos o malos. A pesar de algunos inconvenientes que se presentaron en el desarrollo, la arquitectura descrita es adecuada, si se entiende como adecuado el hecho de que es modificable, aunque sea sólo en unos determinados aspectos, y desde luego también es adecuada porque dió lugar a un sistema implementado que ofrecía las funcionalidades que de él se requería.

Por lo tanto, lo más que se puede decir es si la arquitectura es modificable en uno u otro aspecto, y en alguno de ellos no lo era tanto como se esperaba. Pensando en el proceso de diseño de la arquitectura seguido en este capítulo, y sobre todo después de haber obtenido un resultado de la primera evaluación, cabe plantearse la incorporación y revisión de elementos, funcionalidades y estructura para llegar más allá.

La principal conclusión que se saca de la experiencia mostrada en este capítulo es la necesidad de complementar la primera etapa de diseño, con las consideraciones para la construcción del entorno virtual, especificando cada una de las funciones y elementos que debe contener, así como los elementos de la interfaz. Por estas razones en el capítulo subsecuente se expresan todas las consideraciones para la construcción del entorno, paa posteriormente detallar funcionalidades tanto del STI, como de la interfaz.

CAPÍTULO 4:

CONSIDERACIONES PARA LA CONFIGURACIÓN DEL ENTORNO VIRTUAL

CAPÍTULO 4. Consideraciones Generales para la Configuración del Entorno Virtual.

Como se ha venido refiriendo, el objetivo de esta tesis es proporcionar una solución viable al problema del desarrollo de entornos de aprendizaje con apoyo de tutoría inteligente. Basándose en esta implementación, el presente capítulo describe los detalles de creación y construcción del Entorno Virtual de Enseñanza-Aprendizaje. Este proceso de construcción y las consideraciones y recomendaciones más significativas expuesatas, se constituyen como parte del proceso de diseño y construcción y también integran las recomendaciones para el diseño de entornos semejantes.

4.1 Configuración del entorno

El núcleo del marco de trabajo contiene los elementos propios de todo STI: un modelo del dominio, un modelo del alumno, un módulo de instrucción y una interfaz. Durante el proceso de aprendizaje el alumno irá seleccionando conceptos del currículo y realizando tareas que le ayuden a asimilarlo. La construcción de un entorno de aprendizaje consistirá básicamente en determinar qué materia se va a estudiar y cómo. Para ello será necesario modelar el dominio, seleccionar los Recursos de Instrucción (RI) que formarán parte del entorno, establecer para qué conceptos del dominio es válido cada RI, es decir, qué tareas se podrán realizar para estudiar cada tema y las tareas más adecuadas para un determinado tipo de alumno.

4.1.1Creación del dominio

Esta tarea corresponde a un experto que debe estructurar el dominio, es decir, descomponerlo en unidades de conocimiento (conceptos) y establecer las relaciones entre ellas. El marco de trabajo establece el conjunto de atributos disponibles para definir cada concepto (*nombre, objetivo, nota mínima* y *tipo de evaluación*) así como los tipos de relaciones disponibles (*es_un, parte_de, prerrequisito* y *pertenece_a*…).

Con el propósito de facilitar esta tarea y como parte de la propuesta, se ha construido una herramienta de edición de dominios, EDITOR-DOM, que permite definir un dominio utilizando un entorno visual y generar el fichero OXML correspondiente a partir de un grafo. La Figura 4.1 muestra el aspecto de la interfaz de EDITOR-DOM, en el apartado de la interfaz se explicará tanto el estilo como las directrices para el diseño de este editor. En primer plano podemos ver una red de conceptos y en la esquina inferior derecha el fichero OXML que se genera a partir de la misma.

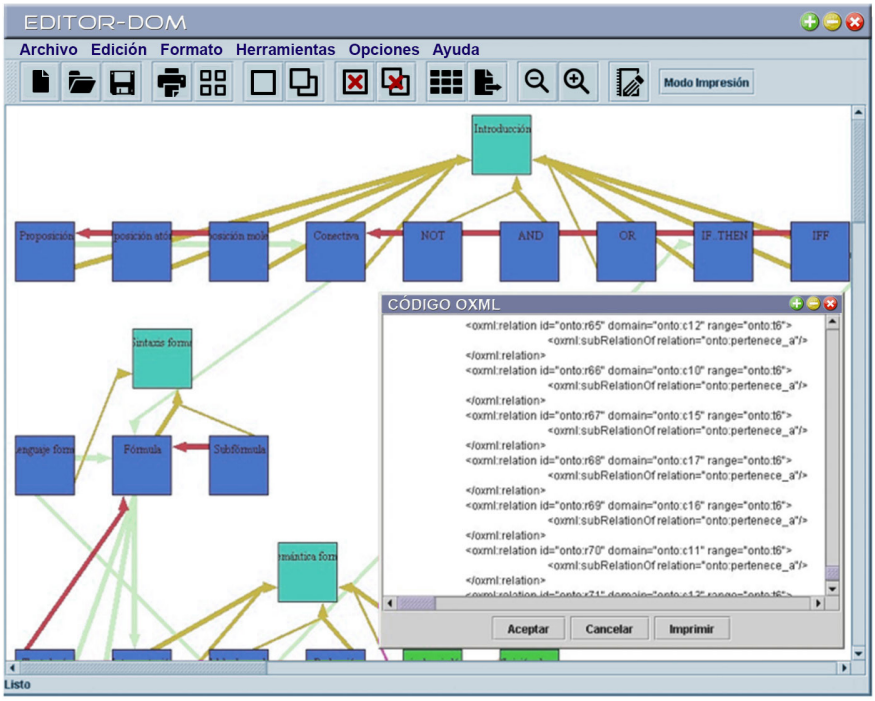


Figura 4.1 Editor de dominios EDITOR-DOM

4.1.2 Selección del dominio

El profesor seleccionará uno de los dominios ya creados para construir su entorno de aprendizaje. Éste constituirá la base sobre la que se irá configurando la estrategia de enseñanza y las distintas tareas que puede realizar el alumno.

4.1.3 Selección de los recursos de instrucción

Como parte de su estrategia de enseñanza, el docente debe asignar a cada concepto del dominio una lista con las tareas que el alumno puede llevar a cabo. La realización de una de ellas provoca la ejecución del recurso de instrucción correspondiente. En este punto surge el problema del aspecto semántico de la comunicación entre dos sistemas inteligentes.

Un Recurso de Instrucción (RI) inteligente se define como una aplicación externa e independiente que posee su propio modelo del dominio. En DECANO se establecieron dos vocabularios comunes al núcleo y al RI, es decir, se comparte ontología del dominio, de modo que, si el alumno solicita realizar una tarea sobre un concepto concreto, el RI será capaz de entender acerca de qué le están pidiendo que le enseñe. Del mismo modo, cuando el RI comunique los resultados

obtenidos, el modelo del alumno del núcleo podrá interpretarlos. Por ejemplo, supongamos un entorno de aprendizaje para operaciones aritméticas simples cuyo dominio contiene el concepto *multiplicación*. Supongamos también que existe un recurso de instrucción que permite realizar operaciones simples cuyo dominio está formado por los términos *suma*, *resta*, *producto* y *división*. Es evidente que para que el recurso pueda usarse a través del entorno alguien debería indicarle que cuando se dice *multiplicación* en realidad se quiere decir *producto*.

Para solucionar esto es necesario establecer una equivalencia de términos entre ambos dominios (estas relaciones constituyen el grueso de la descripción semántica. Como primer paso el profesor solicita al RI, a través del servicio *Exportar modelo del dominio*, la lista de términos que componen su dominio. A continuación establecerá las correspondencias con los conceptos de su propio dominio.

En primer lugar puede ocurrir que para un concepto no exista su homólogo en el dominio del recurso de instrucción. En este caso el concepto no podría ser estudiado usando esa tarea.

Puede darse la situación inversa, que exista un concepto en el dominio del RI para el que no haya uno equivalente en el dominio del WILE. En este caso existiría material del RI que nunca se mostraría al alumno por estar fuera del alcance o el ámbito del dominio de su instrucción.

Otra circunstancia que puede darse es que no exista una correspondencia clara uno a uno, sino que un término del dominio sea equivalente a varios del RI o viceversa (1-n, n-1). Por ejemplo en el entorno que se describe más adelante, el experto definió el concepto *Introducción* que más tarde el profesor asoció a los términos ***Índice***, *Contextualización*, *Presentación, Inicio, Prefacio y Prólogo*.

Una vez seleccionados los RI y establecidas las correspondencias entre dominios, el entorno de aprendizaje estará disponible para su uso. En este punto del desarrollo las estrategias de instrucción son comunes a todos los alumnos. El profesor puede refinar el conocimiento de instrucción estableciendo distintas estrategias para distintos perfiles de alumnos. En el siguiente punto se aborda esta cuestión.

4.1.4 Adaptación de cada tarea a los perfiles del alumno

Un RI es usado por un *alumno* para estudiar un *concepto* del dominio. Puede que algunos de ellos sirvan para configurar aspectos relativos al desarrollo de la tarea y por lo tanto tengan influencia en la estrategia de enseñanza. El integrador del recurso proporciona para cada uno un valor por defecto que, de no indicarse lo contrario, será el empleado por todos los alumnos.

Como parte de la metodología de desarrollo se define que "el profesor puede establecer, como parte de su estrategia, qué tareas son mas adecuadas que otras para un tipo determinado de alumno". Para ello el profesor establecerá correspondencias entre una instancia del modelo de aptitudes y un conjunto de valores de los parámetros de ejecución o un conjunto de atributos de la descripción semántica de un RI (*Learning Type*,...). Para aclarar este punto en la siguiente tabla se muestran dos ejemplos de adaptación realizados respectivamente en el entorno para el recurso Metodología de la Investigación y *Mostrar contenidos* y en el de Taxonomía de Objetivos para el recurso *Test de aprendizaje (Figura 4.2).*

DOMINIO	RECURSO DE INSTRUCCIÓN	PERFIL	PARÁMETRO/VALOR
Metodología de la Investigación	Mostrar contenidos	Desarrollo cognitivo = Bajo	Generic
Taxonomía de objetivos	Test de aprendizaje	Desarrollo cognitivo = Medio/Alto	Adaptive

Figura 4.2: Ejemplos de adaptación de tareas a perfiles para los entornos de Investigación y Taxonomía de objetivos.

La idea es con el WILE adaptar páginas existentes en la web para que el alumno estudie una materia. Estas páginas poseen enlaces que pueden ser externos o apuntar a otras páginas que pertenecen al mismo curso. El sistema permite regular el grado de libertad de navegación a través de esos enlaces. El profesor, al construir el entorno de Metodología de la investigación determinó que para los alumnos con un *desarrollo cognitivo bajo* y con *poca experiencia con la computadora* este parámetro tomará el valor 2 (no se permiten accesos a enlaces externos al curso). La justificación pedagógica es que alumnos poco acostumbrados a navegar "correctamente" por la web y que además no conocen mucho el dominio de instrucción corren un riesgo elevado de perderse en el hiperespacio y por lo tanto necesitan que el sistema les limite el número de caminos, además también consideró cuestiones como la búsqueda en otro tipo de recursos y la utilización de las sugerencias del tutor.

Para el entorno de *Taxonomía de objetivos* el profesor ha establecido que los alumnos con un *desarrollo cognitivo alto* y una *motivación baja* usen el método de selección de preguntas adaptativo. La justificación de esta elección es que para aquellos alumnos cuya habilidad para comprender conceptos abstractos sea elevada, pero estén poco motivados, así se disminuye el riesgo de aburrimiento y abandono, si realizan tests adaptativos que reducen sensiblemente el número de preguntas respecto a los test tradicionales.

El número de relaciones *perfil-parámetro(s)* no está limitado a uno, por ejemplo el profesor podría establecer, qué tipo de alumnos harán tests adaptativos, cuál será el perfil para los tests aleatorios y qué otros se ceñirán al orden establecido por el profesor. Si el perfil de un alumno no coincide con ninguno de los prototipos definidos usará los valores por defecto.

El resultado de este proceso de adaptación depende del modelo de aptitudes del alumno, de la documentación aportada por el desarrollador del recurso y de la experiencia del docente. En efecto, la precisión con la que el modelo de aptitudes permita describir a un alumno así como una exacta descripción de los parámetros de ejecución de un recurso, facilitarán una mejor adaptación. Así, estamos considerando que la experiencia docente del profesor le permitirá manejar toda esta información y aplicar sus conocimientos en favor de una adaptación lo más precisa posible de la instrucción.

4.1.5 Aplicación de distintos estilos de enseñanza

La forma en que diferentes alumnos procesan la información y aprenden una materia varía en función de sus preferencias y habilidades. Es lo que conocemos como estilos de aprendizaje. Un modelo de estilos de aprendizaje establece un conjunto de categorías relativas al modo en el que un alumno procesa la información, clasificándolo de acuerdo con sus preferencias a la hora de aprender (sensitivos/intuitivos, inductivos/deductivos, o bien auditivos, visuales, kinestésicos, etc.). Existen distintos modelos basados en diversas teorías psicológicas como las de Jung (1994), Gardner (1983) o Bloom (1956). De forma paralela a los modelos de aprendizaje y obviamente como parte del complemento del proceso, se han desarrollado modelos de estilos de enseñanza que determinan las técnicas de instrucción que mejor se ajustan a cada estilo de aprendizaje (Felder and Silverman, 1988; Kolb, 1984).

En este punto se presentan algunos de los estilos más usados en el campo de los sistemas inteligentes. El prototipo DECANO permite ajustarse a cualquiera de ellos e incluso simultanear varios, esto quizá sonará pretensioso, pero para el modelado del prototipo y el dominio de instrucción, mucho se ha cuidado este aspecto. A continuación se describe el modo de enfocar el proceso de creación de sistemas inteligentes (según lo visto en los apartados anteriores) para

aplicar cada uno de los modelos presentados.

Aprender haciendo (*Learning by doing*) referido en Schank y otros (1999). El objetivo de esta teoría es fomentar el desarrollo de habilidades y el aprendizaje de hechos en el contexto de cómo serán usados. Para ello se deben presentar al alumno actividades relacionadas con los objetivos de aprendizaje y en las que el alumno pase la mayor parte del tiempo practicando las habilidades que se esperan desarrollar. Existen diversas aproximaciones a este estilo de enseñanza como por ejemplo:

- *Aprendizaje basado en problemas* (*Problem-Based Learning*) definido por Barrows and Tamblyn (1980). Es el aprendizaje que resulta del proceso de trabajar en la resolución de un problema. Para crear un entorno basado en este estilo de aprendizaje usando la metodología que se propone sería necesario:

  Definir un dominio cuyos conceptos representen los distintos problemas que el alumno debe resolver para alcanzar un determinado objetivo de aprendizaje.

  Integrar herramientas para la resolución de los distintos tipos de problemas del dominio.

  Asociar a cada nodo del dominio el/los recursos apropiados para resolver el problema que representan.

- *Aprender explorando* (*Learning by exploring*) Según exponen Schank y Cleary (1995) es una situación cuando un alumno está desarrollando una tarea y se implica en el aprendizaje, genera preguntas y aprende de ellas. Esta teoría se basa en responder a esas preguntas conforme el alumno se las va haciendo y a la vez presentar ejemplos, explicaciones, etc., relacionados con las respuestas a las preguntas para que el alumno vaya recorriendo el espacio de conocimiento según sus necesidades.

Para crear un sistema exploratorio con este marco de trabajo el profesor deberá

1. Definir un dominio que incluya, además de conceptos propios del espacio de conocimientos, nodos que representen preguntas frecuentes.
2. Crear páginas cuyo contenido sean respuestas a esas preguntas, ejemplos que justifiquen las respuestas, explicaciones que las complementen, recursos multimediales que faciliten su comprensión, etc.
3. Asociar esas páginas a los conceptos del dominio correspondientes, por eso tanta insistencia en la documentación y que el sistema se alimente de los propios alumnos.

- *Aprendizaje basado en proyectos* (*Project-Based Learning*). Es un método de instrucción centrado en el alumno. A través de la construcción de un objeto significativo para el alumno (un juguete, una presentación multimedia, un poema, un ensayo, etc.) los alumnos representan lo que han aprendido. Para ello se le plantea al alumno una tarea cuyo resultado es la construcción de un objeto que puedan compartir con otros alumnos y se le facilitan recursos para llevarla a cabo (libros, herramientas, entre otros).

La traslación de este método de enseñanza al proceso de creación es prácticamente inmediata, quedando como sigue:

1. El dominio representaría el guión de desarrollo del proyecto. La granularidad del mismo podría variar desde un único nodo que represente la tarea a desarrollar hasta un conjunto detallado de pasos que hay que completar para lograr el objetivo.

2. A cada nodo se le asociarán los recursos necesarios para llevar a cabo esa parte del proyecto.

Aprendizaje basado en casos *(Case-Based Learning)* (Schank and Cleary, 1995). La idea de esta teoría es proporcionar al alumno la información que necesita en el momento que la necesita para que pueda aprender de sus errores. El alumno es ubicado en una situación concreta. Cuando el instructor detecta que se ha bloqueado o ha cometido un fallo deduce que está listo para aprender e interviene. Los alumnos pueden pedir ayuda cuando lo necesiten. El aprendizaje basado en casos complementa el estilo de *aprender haciendo* (*Learning by doing*) proporcionando al alumno las tareas que deber ir realizando.

La implementación de esta estrategia no es inmediata. La capa de la metodología de desarrollo indica que el alumno puede realizar la tarea que desee en el momento que considere más oportuno a lo largo de su proceso de instrucción. El sistema tiene control parcial sobre las acciones del alumno y sólo está preparado para intervenir en un momento concreto o a petición expresa del alumno. Sin embargo, durante la ejecución de una tarea se cede el control al recurso de instrucción asociado a la misma. De este modo, el profesor podría integrar recursos que implementen el aprendizaje basado en casos y asociarlo a aquellos conceptos de su dominio para los que considere que sería útil la aplicación de este método de enseñanza.

4.1.6 Creación del material didáctico

Los sistemas educativos inteligentes suelen tener arquitecturas fuertemente acopladas y poco modulares. Esto impide en muchas ocasiones reutilizar sólo el material didáctico ya que éste separado del resto del sistema carece de sentido. Esta es una de las razones por las que se

plantea como uno de sus principales objetivos integrar sistemas completos como partes de otro reutilizando, no sólo el material, sino las estrategias de enseñanza diseñadas alrededor de ese material. Para ello también en este punto, se adicionaron al sistema, como referencias de partida el modelo de STI propuesto por Ferreira, Moore y Mellish (2007) y el parser ELE-TUTOR desarrollado por Ferreira y Kotz (2010), los cuales determinan aspectos del STI tales como las estrategias de feedback a utilizar, los tipos de error a tratar, el dominio, los tipos de actividades y la representación del sistema de conocimiento del estudiante.

Existen recursos de instrucción para la enseñanza de dominios específicos, como ELM-ART de LISP, Algebrain-Tutor de álgebra o Medtec (Eliot y otros, 1997) de anatomía y otros genéricos (aplicable a varios dominios), como AHA, CALAT o CATGlobal. Independientemente de esto, los recursos pueden contener material ya creado por otros profesores y, en algunos casos, herramientas de autor para el desarrollo de nuevo material. En cualquier caso la creación de material didáctico es un proceso externo al sistema.

El marco de trabajo descrito incluye recursos de instrucción basados en dos aplicaciones complementarias que se diseñaron: El programa para *Mostrar contenidos (*CONTENT) y El Editor de Dominios (EDITOR-DOM).

4.1.6.1 Página web

CONTENT es un sistema para la construcción de recursos web mediante la indexación de páginas ubicadas en Internet. En este entorno el modelo del dominio es una jerarquía de conceptos con una o varias URL asociadas. El sistema permite establecer prerrequisitos entre los conceptos. El profesor puede definir distintos grados de libertad en la navegación del alumno.

Para ello puede a) deshabilitar todos los enlaces de las páginas incluidas en el curso, de modo que el alumno sólo pueda navegar a través del árbol de conceptos; b) dejar todos los enlaces contenidos en las páginas, permitiendo al alumno navegar libremente y c) habilitar sólo los enlaces que hacen referencia a páginas que pertenecen al curso.

4.1.6.2 Tests y evaluaciones

Como parte de las herramientas internas, el sistema cuenta con una herramienta de autor para la definición de ítems y tests. Un test tiene una estructura jerárquica formada por *tests*, *temas* e ***ítems***, donde se incluye uno o más temas, cada uno de los cuales contiene un conjunto de ítems. Un tema puede pertenecer a más de un test y una pregunta a diferentes temas.

Este recurso permite definir distintos tipos de preguntas (respuesta única, respuesta múltiple, respuesta libre, etc.) así como configurar algunas características de los tests referentes a su presentación, modo de evaluación y método de selección de ítems. Se genera y muestra el test del concepto del dominio seleccionado por el alumno desde el entorno de aprendizaje.

4.1.7 Actualización y mantenimiento del entorno de aprendizaje

Un entorno de aprendizaje siempre está sujeto a mejoras y modificaciones. Estas actualizaciones pueden afectar a cualquiera de los elementos creados por los actores que intervienen en el proceso de desarrollo: modelo del dominio, material, recursos de instrucción seleccionados y estrategias educativas resultado de la adaptación de tareas.

Actualización del material docente. La creación y/o modificación de material docente debe realizarse a través de las herramientas de autor de cada recurso, en caso de que estos lo permitan. Si el profesor modifica y/o crea material acerca de un concepto del dominio puesto en correspondencia con un concepto del recurso, la actualización es inmediata. Cualquier alumno que seleccione el recurso para estudiar el tema afectado dispondrá del nuevo material al momento. Si por el contrario, el profesor crea nuevo material para un concepto que en un principio no fue relacionado con otro del recurso, es necesario establecer esa correspondencia si se quiere que el alumno pueda acceder a dicho material.

Integración y/o eliminación de recursos de instrucción. El sistema permite la integración de recursos en el marco de trabajo de forma dinámica. Basta proporcionar su descripción funcional y automáticamente estará disponible para aquellos docentes que deseen añadirlo a sus entornos siguiendo un proceso documentado en la arquitectura. Para eliminar completamente un recurso de un entorno sólo será necesario desvincularlo del mismo sin que esto afecte a su disponibilidad para otros entornos.

Asignación de tareas a temas: El profesor puede modificar en cualquier momento las relaciones entre los términos de su dominio y los del dominio de un determinado recurso. Como consecuencia de ello el alumno podrá realizar nuevas tareas para estudiar algunos conceptos o dejará de tener a su disposición otras, según se añadan o eliminen correspondencias.

Modificación del dominio: La modificación del dominio puede afectar al modelo del alumno y por supuesto a las correspondencias establecidas con los dominios de algunos recursos. Si se eliminan conceptos del dominio, la información relativa a

los mismos será eliminada de los modelos de los alumnos. Del mismo modo serán eliminadas de la descripción semántica todas aquellas relaciones en las que dicho concepto participe. Por otro lado, si se añaden conceptos al dominio, el modelo del usuario no se verá afectado ya que la información relativa a los mismos se irá construyendo durante el proceso de aprendizaje. En cuanto a las descripciones semánticas, el profesor deberá establecer las correspondencias necesarias, si desea que los nuevos conceptos tengan tareas asignadas dentro de su sistema.

4.2 El Aprendizaje

La metodología desarrollada establece el modo en el que se llevará a cabo el aprendizaje en un WILE una vez construido. Según esto, a lo largo del proceso de aprendizaje el alumno irá seleccionando conceptos del currículum (microadaptación) y realizando tareas que le ayuden a asimilarlo (macroadaptación). Para ello contará con la ayuda del planificador de instrucción que en cada momento le recomendará el concepto y la tarea más adecuados para él, basándose principalmente en su modelo de conocimientos. Éste será actualizado después de cada tarea. Las actualizaciones se reflejan en la interfaz por el color de los iconos que acompañan a cada concepto del árbol curricular: el *rojo*, indica que el alumno no está preparado para abordar el aprendizaje de un concepto; el color *verde* quiere decir que el alumno tiene los conocimientos suficientes para empezar a aprender el concepto y el *azul* significa que el sistema cree que el alumno ya ha aprendido el concepto.

A pesar de que se permite a los alumnos ignorar las indicaciones del sistema y seguir su propio camino durante la instrucción, todas sus acciones estarán supeditadas a la estrategia global establecida por el profesor durante el proceso de construcción del entorno y serán monitoreadas permanentemente por el Planificador de rutas. Estas directrices se reflejan en:

El alumno sólo podrá usar aquellos recursos de instrucción que hayan sido integrados como parte del entorno, independientemente de los que estén disponibles en el marco de trabajo.

Cada concepto tendrá asociada una lista de recursos. Esta lista estará compuesta por aquellos recursos para los que el profesor ha establecido un vínculo entre un concepto de su dominio y el estudiante. Los alumnos que deseen estudiar este concepto sólo podrán realizar una tarea de las que aparecen en esa lista.

El diseño del entorno, principalmente el dominio y los recursos seleccionados por el profesor, pueden marcar implícitamente un estilo de enseñanza (aprendizaje basado en problemas, en casos, por exploración, etc.) tal y como se explicó previamente.

Cada recurso de instrucción inteligente es responsable de su material didáctico y de la forma en que se presenta, es decir, cada uno implementa sus propias estrategias de enseñanza. Es responsabilidad del profesor elegir los más adecuados. El conocimiento de instrucción aportado por el profesor durante la construcción del WILE influirá en mayor medida en el grado de satisfacción del alumno que las estrategias de planificación del núcleo, dado que estas sólo proporcionan ayuda y guía.

4.3 Requisitos

4.3.1 Del alumno

Con la finalidad de facilitar el acceso y el uso provechoso del entorno de aprendizaje, así como fomentar una cultura del conocimiento según los términos de Charles (2002), previamente hay que capacitar al estudiante brindándole estrategias de resolución de problemas y metaconocimiento acerca del propio conocimiento.

La forma de brindar este conocimiento podría ser, por ejemplo, a través de clases magistrales o entrega de notas. La meta es que el estudiante tome conciencia de las formas del conocimiento y cómo gestionarlo. Para brindar esta capacitación, se elaboró un material que resume los principales conceptos e ideas a manejar. Dicho material se intregró formalmente como conocimiento del dominio.

4.3.2 Del profesor

El conocimiento sobre la materia que estará en el entorno será provisto por el profesor y, o, experto del dominio. Será necesario capacitar al profesor en identificar los tipos de conocimiento, con la intención de estructurar esos conocimientos adecuadamente.

4.4 Funcionamiento esperado del entorno

Con el objetivo de brindar una rápida descripción de cómo funcionaría el entorno que se propone, se ofrece a continuación un relato ejemplificando un posible uso por parte de un estudiante, en forma algo similar a un caso de uso, según la terminología de Ivar Jacobson (1992).

Posteriormente, se presentará el modelo formal, en el cual se describirá y fundamentará cada uno de los elementos utilizados en el entorno. Se detallará el uso del entorno simplificado considerando la asignatura Metodología de la Investigación, que se enfoca en que el alumno aprenda los tipos de metodología que la UAQ permite para la estructura de sus tesis de licenciatura o bien de su reporte de práctica profesional, considerando estas dos modalidades como sólo dos de las formas de titulación, puesto que existen otros mecanismos alternos no tomados en cuenta para el sistema.

El estudiante ("actor"), quien previamente es capacitado en gestión del conocimiento y resolución de problemas, se registrará en el sistema y le aparecerá la lista de los temas disponibles (en este ejemplo simplificado): *Temáticas, Relaciones, Objetivos, Marcos de conceptos, Redacción de preguntas, etc*. El estudiante, sabiendo que desconoce o está aún confundido con el punto Temáticas, elige éste de la lista. A continuación le aparecerá la lista de los tipos de conocimiento disponibles sobre el tema, en este caso *Conocimiento descriptivo* (básicamente descripción o definición del conceptos disciplinarios relativos al Diseño y las Artes visuales, especialides, disciplinas y subdisciplinas, interdisciplinas, transdiciplinas) y *Conocimiento procedimental* (describe el procedimiento para construir un tema). Ante la selección de *descriptivo*, aparecerá la definición del concepto Temáticas, permitiendo también ver definiciones alternativas desde otras perspectivas. Se le ofrecerá a continuación la lista de tipos de conocimiento disponibles relacionados al concepto presentado, en el caso planteado podría ser *anecdótico* (que ofrece una historia del concepto), *procedimental* (cómo implementar o construir un tema), *ampliativos* (temas recomendados para continuar o extensivos a la disciplina), *aplicativos* o de *monitoreo* (ejemplos y ejercicios sobre el concepto), etc.

En otras palabras, partiendo del reconocimiento, por parte del alumno, de una carencia de conocimiento en cierto tema (o aún desde la percepción del propio sistema de dicha carencia), se le ofrecerá el conocimiento disponible en el entorno sobre él desde la perspectiva de los diferentes tipos de conocimiento, dejando principalmente en el alumno la propia gestión del desconocimiento, con la finalidad de fomentar el autodidactismo y favorecer su trabajo autónomo e independiente. Además, el alumno dispondrá del apoyo de una memoria institucional, a la que podrá recurrir en busca de respuestas a preguntas frecuentes, enterarse de las mejores prácticas y de las lecciones aprendidas, así como incluir nuevos elementos, permitiendo la integración de sus habilidades individuales al grupo u organización. Permanentemente, a través de un STI, se podrá hacer un seguimiento de las acciones del alumno, lo que permitirá ofrecerle sugerencias sobre las alternativas más adecuadas de acuerdo a sus preferencias mostradas durante el uso del entorno así como recomendaciones varias.

A su vez, el docente dispondrá de herramientas que le permitan la carga del dominio del conocimiento en el módulo de gestión del conocimiento, que incluye, como se verá, el listado, la

memoria institucional, entre otros componentes. Tendrá que completar una "ficha" o registro por cada concepto o procedimiento con los tipos de conocimiento. Así pondría que la definición del concepto de "diseño" es conocimiento descriptivo y su valor es " *«La fase del proceso productivo en la cual se definen todas las características de un producto (visuales, formales, tecnológicas, utilitarias, constructivas, materiales etc), su forma de producción, distribución y consumo, previo a la producción material».* (Chaves, N. 2004)". Además, como conocimiento procedimental asociado a ese mismo concepto podría estar la forma de crear o llevar a cabo las fases para «materializar» ese diseño.

4.5 Aportes del modelo

En relación al modelo propuesto por el estándar (IEEE P1484.1/D9, 2001), las diferencias o aportes fundamentales son:

1. aparece la relación directa entre la entidad aprendiz (*learner entity*) y los recursos de aprendizaje (*learning resources*), pues el estudiante tiene acceso directamente a ellos, gestionando su conocimiento y,o, desconocimiento;
2. se define una estructura para los recursos de aprendizaje (en el repositorio de conocimiento), centrándolos en los tipos de conocimiento a partir, como se verá, de una ontología.
3. El módulo de gestión del conocimiento contendrá, entre otros elementos, la memoria institucional que será utilizada y mantenida tanto por estudiantes como docentes;
4. se incorpora la entidad profesor;
5. a evaluación está contenida en el propio módulo de gestión del conocimiento;
6. el estudiante tiene acceso directo a sus registros; y
7. se incluye la evaluación del propio repositorio de conocimiento por parte del estudiante.

En resumen, de las diferencias citadas, podrían resaltarse como puntos fundamentales la definición de la estructura de los recursos de aprendizaje (según la terminología de la norma de IEEE referida), el módulo de gestión del conocimiento y la incorporación de la entidad profesor.

Si se compara con el modelo de Gil (2002) no se fija un orden preestablecido y se observa que se incorpora todo el módulo de gestión del conocimiento, aspecto no detallado en su propuesta. Se sigue las recomendaciones de Fernández - Guinea (2001), Ferreira y Kotz (2010) en cuanto es aprendizaje activo; le permite al alumno acceder a amplios recursos de aprendizaje y la toma de decisiones está en el alumno (Salinas, J., 1997) y los contenidos se seleccionarán a partir de las indicaciones y preferencias del estudiante, en forma dinámica como lo recomienda Carro, R. (2001). Se tiene en cuenta los estilos de aprendizaje, pues el material se presentará en

diferentes formas y medios, permitiéndose el acceso al mismo según las preferencias del propio alumno (Felder, R. *y otros*, 1988). Los contenidos del entorno son extensibles y actualizables como refieren Trikic, A. (2001) , López, A. (2000), Ferreira yKotz (2010).

Respecto a los criterios de clasificación de entornos, esta arquitectura se aplicaría prácticamente a cualquier área de conocimiento, de contenido intelectual y donde las destrezas de resolución de problemas tienen un lugar importante. Utilizando los criterios de Marqués (1998); también, permitirá desarrollar entornos que se pueden catalogar como de bases de datos abiertas, que integra múltiples medios (pues cada tipo de conocimiento estará representado en la forma deseada), con inteligencia (pues dispone de un proceso supervisor), orientado a todos los objetivos educativos (conceptuales, procedimentales, actitudinales) y a todas las actividades cognitivas (como análisis, síntesis, cálculo, etc). Es independiente del tiempo y de estudio independiente (Pohjonen, J., 1997) y propicia los recursos (Almeida, S. *y otros*, 1997).

Asimismo, permite, según los criterios de Yildirim y colegas (2001) los diversos tipos de conocimiento. Revisa y recupera los componentes del aula virtual de Azpiazu y colegas (2002), pues se definirán por completo los elementos y se incorporan otros no presentes en su propuesta (por ejemplo, glosarios, infografías, resultados gráficos de tareas).

También permite que el estudiante controle su proceso de aprendizaje, aspecto destacado por Follows (1999). Se evita la sobrecarga de información criticada por Liaw (2001), pues aquí es el propio estudiante quien guía el proceso, paso a paso.

El modelo podría considerarse como orientado a construir entornos constructivistas según las características citadas por Liaw (2001) y Leidner y colegas (1995), pues fundamentalmente provee varias representaciones de la realidad y enfatiza la construcción de conocimiento, en particular a través de la gestión del conocimiento.

4.5.1 Marco conceptual del modelo

Para el diseño del modelo presentado, se propone definir una ontología para analizar el dominio del conocimiento, profesor y estudiante y hacer explícitas las suposiciones. Más adelante se presentarán detallados los componentes del modelo.

Como indican Schreiber y colegas (1998), al construir un modelo de conocimiento en general debe: identificarse el conocimiento, especificarse el conocimiento y refinar el conocimiento.

Como sugieren Noy y colegas (2000), es valioso revisar las ontologías existentes con la intención de hacer reuso de ellas. Realizada la búsqueda (entre otros lugares, http://www.ksl.stanford.edu/software/ontolingua, www.daml.org/ontologies y www.unspsc.org), no se encontró al momento de escribir estas líneas una ontología que vincule o presente relacionados los conceptos de tipo de conocimiento, estudiante y profesor. Como metodología sugieren luego de enumerar los términos importantes, definir la jerarquía de clases, indicar las propiedades de las clases (atributos o "slots"), identificar las facetas de las propiedades (facets) y crear las instancias.

Se siguieron estas recomendaciones debido a su simplicidad. Se presenta la ontología elaborada luego de relacionar, integrar y unificar las propuestas y conceptos descritos en los capítulos previos. Se ofrecen los términos básicos y sus relaciones, como sugieren Neches y colegas (1991).

Esta ontología se podría catalogar en función del tipo de la estructura de la conceptualización como de modelado como refieren Van Heijst, G. y otros *(*1997), pues incluye términos e información sobre las estructuras de las bases de datos. Considerando la dimensión del sujeto de la conceptualización, podría clasificarse como de dominio. Sería terminológica aplicando la clasificación de Sowa (2001). En relación a la clasificación de Mizoguchi y colegas citada en Gómez, A. y otros (1997) sería una ontología dependiente del dominio, en particular del concepto.

Conclusiones

En este capítulo se ha descrito los requerimientos de los actores más importante dentro del proceso, el profesor que planifca el contenido del sistema y el alumno quien recibirá la información y tendrá interacción con el mismo.

El desarrollo de estos sistemas pone de manifiesto que la metodología es aplicable a un amplio rango de dominios. El modelo del dominio se estructura como un conjunto de conceptos. El marco de trabajo especifica como se van a describir estos conceptos (atributos) y las relaciones que se pueden establecer entre ellos. En el marco implementado para esta tesis se han definido relaciones muy genéricas aplicables a la mayoría de los dominios. La inclusión de relaciones más especificas, no limita el espectro de dominios, por el contrario, lo amplía. Aunque existan algunas relaciones imprescindibles para modelar ciertos dominios, para la mayoría este factor no es determinante. En efecto, el marco de trabajo es lo suficientemente flexible como para permitir modelar no sólo modelos declarativos, sino también habilidades o procedimientos. Para describir, por ejemplo, algún experimento de laboratorio (químico, físico, etc.) bastaría con identificar las acciones que el alumno debe conocer para poder desarrollar el experimento literalmente describiendo uno a uno y obviando el hecho de que un cambio en la descripción haría que el

experimiento real, si el alumno lo desarrolla a partir del EV, fracase.

Cada fases o etapa sería considerada un concepto y, mediante la relación de prerrequisitos, podría establecerse la secuencia de acciones correcta para alcanzar el objetivo. Cualquier dominio susceptible de descomponerse en unidades de conocimiento podría usarse para construir un WILE siguiendo nuestra metodología de desarrollo. En última instancia el grado de generalidad depende del marco de trabajo que se implemente.

Uno de los objetivos planteados al comienzo de esta tesis era proporcionar una solución viable a uno de los principales prblemas en la construcción de sistemas educativos inteligentes: su elevado costo de desarrollo. En efecto, y según se deduce de la revisión realizada en los Capítulo 2 y Capítulo 3, la mayoría de aplicaciones educativas inteligentes se construyen desde cero e implican a numerosos desarrolladores, muchos de ellos técnicos que intentan aplicar los conocimientos proporcionados por los docentes y los expertos en el dominio. Para subsanar estos inconvenientes la metodología propuesta en esta tesis aboga por la reutilización de sistemas ya construidos como medio para permitir a) reducir los costos de creación de material e implementación de estrategias educativas y b) la construcción de WILE por personal no especializado, por esto se hizo una revisión y análisis detallado de los que a juicio del equipo resultaba idea reconsiderar y tomar.

La reutilización de código proporciona ventajas que atañen no sólo al proceso de desarrollo, sino también al resultado desde el punto de vista del alumno. Es evidente que la posibilidad de usar otras herramientas, incrementa la riqueza del entorno y permite aprovechar sistemas existentes acerca de un dominio que han sido desarrollados *ad hoc* y que representan un compendio del conocimiento de numerosos expertos. Esto quedará de manifiesto una vez que ser revise la propuesta, haciendo hincapié que se ha solicitado el permiso de varios autores para retomar y ampliar sobretodo al arquitectura del sistema.

Aún a pesar de lo anterior, se insiste en que el resultado fue totalmente inesperado, muy agotador y pues cuestiones justamente de tiempo, dinero y estudiantes poco familiarizados con esta cuestión, el valor del producto crece. Ya a nivel personal, he de decir que actualmente hemos aplicado la célula para dos convocatorias que quizá de ser beneficiados, puedan ayudarnos a mejorarlo o quien sabe, incluso hasta terminarlo.

CAPÍTULO 5:

MODELO DE ARQUITECTURA

CAPÍTULO 5. Modelo de Arquitectura, Metodología y Criterios Mínimos de Desarrollo.

5.1 Consideraciones iniciales

Como se ha podido ver en capítulos anteriores, uno de los aspectos que suelen quedarse sin considerar son los atributos de calidad con los que se desea dotar al prototipo, como lo refieren Bass y otros (2003), al dar por sentado que no son necesarios, que aparecerán solos, o que el equipo de desarrollo será capaz de obtenerlos porque la metodología lo permite en gran medida.

Por este motivo, además del diseño de una arquitectura software para Entornos Virtuales de Enseñanza-Aprendizaje con Tutoría Inteligente, se ha pensado que este documento también aporte una herramienta que complemente a la arquitectura y, sobre todo, que ayude en el desarrollo de este tipo de sistemas allí donde las metodologías abandonan al equipo o al desarrollador un poco a su suerte o a lo que vaya resultado en el camino.

Así, y principalmente en la dinámica de trabajo colaborativo, otra aportación adicional que se proporciona con este trabajo son una serie de recomendaciones metodológicas, obtenidas a partir del diseño de la arquitectura, y que constituyen las bases que han permitido llegar hasta la solución descrita. Estas recomendaciones deben entenderse más bien como una serie de criterios de decisión que ayuden al diseño del tipo de prototipos objeto de estudio, allí donde otras metodologías no cubren estos aspectos.

Adelantando conclusiones, si se ha conseguido dotar al prototipo de la modificabilidad que se deseaba, y así se ha pretendido hacer y demostrar, entonces es de esperar que en un periodo de tiempo relativamente corto, el diseño de la arquitectura sea distinto del expuesto en este trabajo. Este hecho no podrá considerarse un fracaso. Muy al contrario, no será sino la demostración del éxito logrado en los objetivos de modificabilidad propuestos, los cuales se han conseguido, en gran parte, gracias a las recomendaciones que se recogen en el resto del presente capítulo.

5.2 Proceso de Diseño de la Arquitectura.

Antes de presentar la propuesta definitiva de arquitectura, es importante tener claro lo que se entiende precisamente por “arquitectura de un sistema software”. Para ello utilizaremos una metáfora, comparando la arquitectura del software con la arquitectura de edificios. Un edificio es habitualmente una sola unidad desde la perspectiva del cliente. El arquitecto del edificio puede encontrar útil hacer una maqueta a escala del edificio, junto con los dibujos, planos y demás vistas del edificio desde distintas perspectivas. Como un edificio, un sistema software es una única entidad, pero al arquitecto de software y a los desarrolladores les resulta útil presentar el

sistema desde diferentes perspectivas para comprender mejor el diseño. Estas perspectivas son vistas del modelo del sistema. Todas las vistas juntas representan la arquitectura.

La arquitectura software abarca decisiones importantes sobre:

1. La organización del sistema software.
2. Los elementos estructurales que compondrán el sistema y sus interfaces, junto con sus comportamientos, tal y como se especifican en las colaboraciones entre estos elementos.
3. La composición de los elementos estructurales y del comportamiento en subsistemas progresivamente más grandes.
4. El estilo de la arquitectura que guía esta organización: los elementos y sus interfaces, sus colaboraciones y su composición.

Sin embargo, la arquitectura software está afectada no sólo por la estructura y el comportamiento, sino también por el uso, la funcionalidad, el rendimiento, la flexibilidad, la reutilización, la facilidad de comprensión, las restricciones y compromisos económicos y tecnológicos, y la estética. Un sistema software es difícil de abarcar visualmente porque no existe en un mundo de tres dimensiones. Suele utilizar tecnología poco probada o una mezcla de tecnologías nuevas. Tampoco es raro que el sistema lleve a sus últimos límites la tecnología existente. Además, debe ser construido para acomodar gran cantidad de componentes que sufrirán cambios futuros. A medida que los sistemas se hacen más complejos, “los problemas de diseño van más allá de los algoritmos y las estructuras de datos para su computación”.

Una vez iniciada la etapa de diseño, el desarrollo de la arquitectura del prototipo utilizó el conjunto de métodos y técnicas proporcionados por el grupo de arquitecturas software del SEI (Software Engineering Institute, Carnegie Mellon UniverSTIy), surgidos en torno al año 1995.

Como principal herramienta para el diseño de la arquitectura software del prototipo se propone el método ADD (Attribute Driven Design) considerando a Bass y otros, 2001y a Wojcik y colegas, (2006). ADD es un método iterativo que se basa en la sucesiva descomposición de los distintos módulos del prototipo hasta obtener una arquitectura que permita conseguir los objetivos funcionales y de calidad deseados. Para descomponer cada módulo es necesario seleccionar los escenarios de calidad que le afectan y utilizando las tácticas y estilos arquitectónicos más indicados para conseguir los atributos de calidad perseguidos, como los definidos en Bachmann (2007). Donde sea preciso, también se utilizarán patrones de diseño orientados a agentes, como los definidos en Hayden (1999), ya que algunos de ellos están especialmente orientados a dotar a la arquitectura de mayor *Flexibilidad.*

El uso de ADD se complementa con QAW (Quality Attribute Workshop) (Barbacci y otros, 2003). Básicamente, QAW consiste en la realización de una reunión en la que participan las partes interesadas en el desarrollo de la aplicación, con el objeto de identificar los atributos de calidad y generar los escenarios de calidad. A través de distintas fases, se generan, refinan y priorizan una serie de escenarios que identifican los atributos de calidad deseados para el prototipo. QAW no es un método sistemático y objetivo, por lo que los resultados dependen de los participantes en las sesiones. Particularmente para este proceso, QAW se aplicó con sumo cuidado considerando que la calidad en un entorno de aprendizaje debe mediar entre estrategias y conocimientos que deben reflejarse en el prototipo proyectado.

Una vez obtenidos los escenarios de calidad con QAW, se utilizaron como entrada para el diseño de la arquitectura con ADD (Nord y otros, 2004). De los distintos métodos existentes para evaluar arquitecturas software, SAAM (Software Architecture Analysis Method), ARID (Active Reviews for Intermediate Designs) y ATAM (Architecture Trade-ofi Analysis Method), fueron los seleccioneados por las siguientes razones:

Los expertos involucrados en el desarrollos coincidieron en que ARID es más apropiado para estadios intermedios de la arquitectura, lo cual lo hace menos apropiado para nuestros intereses. También mencionaron que ATAM es un método más moderno y evolucionado que SAAM, al cual podría decirse que engloba, aunque SAAM es más específico para evaluar modificabilidad y eficiencia. La tercera razón es que existen precedentes de evaluación de arquitecturas basadas en agentes utilizando ATAM de acuerdo a Woods y Barbacci (1999), Ivezic y otros (2000), Boucké y otros(2006), lo cual no sucede con los otros dos métodos.

Al igual que QAW, ATAM se basa en la generación de una serie de escenarios que, en este caso, se utilizarán para evaluar la arquitectura. Los escenarios son tanto funcionales como específicos para evaluar los atributos de calidad y, al igual que sucedía con QAW, no sirven para dar una medida objetiva de la calidad de la arquitectura, sino que reflejan los intereses de los evaluadores en el momento de realizar la evaluación. Las salidas del método, por tanto, no son medidas absolutas de calidad, sino una lista de puntos neurálgicos de la arquitectura, podríamos decir que se trata de un marco de acción para evaluar la arquitectura y una lista de problemas, algunos resueltos y otros no resueltos, que finalmente y de manera práctica se convierten en recomendaciones a seguir o evitar.

Por los motivos anteriormente expuestos, los pasos a seguir para el diseño de la arquitectura y su posterior evaluación fueron los siguientes:

Utilización de QAW para generar los escenarios de los atributos de calidad que interesa contemplar.

Utilización de ADD para definir la arquitectura software.

Al finalizar la aplicación de ADD, se evaluará la arquitectura en función de un método de criterios enumerados (Wooldridge y Jennings, 1999), de manera que se cuente con unos criterios de evaluación específicos para sistemas basados en agentes.

Una vez finalizada la descripción de la arquitectura, se evaluará utilizando ATAM, para refinarla en caso necesario.

5.3. Atributos de Calidad

De los seis atributos de calidad que se mencionan al inicio (Bass y otros, 2003) como relevantes para el diseño de la arquitectura de un sistema software, hay dos, modificabilidad y eficiencia, que han sido identificados por las personas involucradas en el desarrollo de la presente arquitectura como de especial importancia.

La modificabilidad es el atributo es en el que se pondrá especial cuidado y atención, ya que el principal objetivo que se persigue es el desarrollo de una arquitectura que facilite el intercambio de determinados componentes de la misma.

La eficiencia en la ejecución se vuelve determinante, ya que una de las características deseables de un EVEATI es que pueda actuarse dentro de él de forma similar a como se haría en un entorno real, lo cual implica una velocidad de ejecución lo más cercana posible al tiempo real.

Los otros atributos: calidad, seguridad, disponibilidad y usabilidad, no serán descartados totalmente, si bien no suponen, en el momento en que se aborda el diseño de la arquitectura, factores que deban tenerse en cuenta a la hora de darle forma a la misma si tienen implicaciones considerables y por lo menos en cuando a la usabilidad, mucho se tocará en la propuesta de modelo de interfaz.

A continuación se describen los escenarios de calidad que se han identificado como más relevantes y que se tendrán en cuenta a la hora de diseñar la arquitectura. Se utilizará la notación sugerida por Bass, *y otros* (2003) tal y como se muestra en la Figura 5.1.

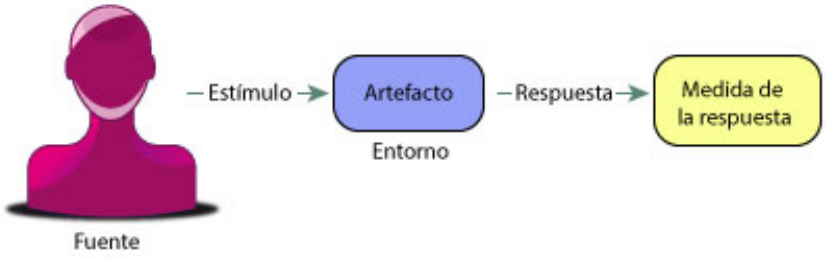


Figura 5.1 Escenario de calidad genérico

5.3.1. Propiedades de Modificabilidad

La modificabilidad del prototipo debe considerarse desde dos puntos de vista. Por una parte, como ya se ha mencionado, se pretende diseñar una arquitectura que posibilite la sustitución de algunos de sus componentes por otros que proporcionen funcionalidades similares. En presencia de un estándar, debería ser posible intercambiar componentes sin mayores consecuencias. No siendo así, es de esperar que sea necesario adaptar las interfaces de comunicación y el formato de los datos que intercambian los distintos módulos, aspecto que se refleja en el escenario de la Figura 5.2.

También se desea que, en función de las necesidades de la persona que diseña el curso o de quien realiza la enseñanza, haya algunas funcionalidades que se puedan añadir o quitar sin tener que modificar el diseño de la arquitectura o el código de la aplicación. Debe ser posible realizar estos cambios, para lo cuál se propone hacerlos a través de un fichero de configuración, una opción de un menú o algún otro medio similar, razón por la cual es un mecanismo que debería poder funcionar bajo demanda sin mayores complicaciones.

Estas necesidades se reflejan en los escenarios de las Figuras 5.2 y 5.3.

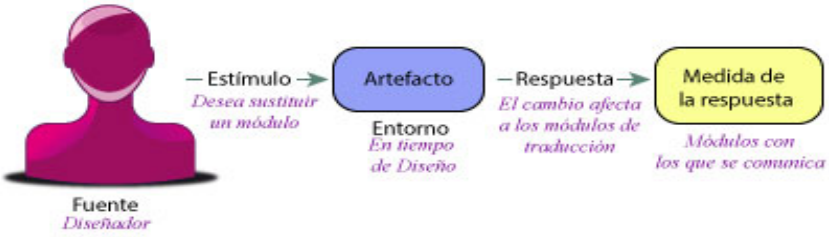


Figura 5.2: Escenario general de modificabilidad en diseño

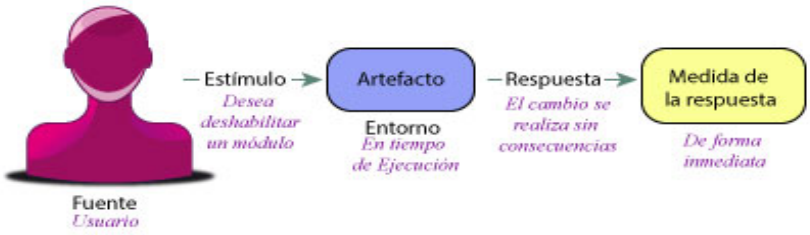


Figura 5.3: Escenario general de modificabilidad en ejecución

Los siguientes apartados describen las modificaciones concretas que se tendrán en cuenta.

Sustitución del Entorno Virtual

Uno de los problemas a los que comúnmente deben hacer frente los equipos que desarrollan STIs que funcionan junto a un EV es que ningún miembro del equipo suele ser experto en el desarrollo de EVs. Los Evs desarrollados suelen adolecer de defectos como escenarios mal estructurados, algoritmos poco eficientes o modelos 3D de aspecto bastante pobre. Todo ello hace que tanto el resultado final de la aplicación, como la velocidad en la ejecución hagan desmerecer la labor realizada por estos equipos.

Por lo tanto, una de las necesidades fundamentales en estos casos es la posibilidad de conectar EVs que han sido desarrollados por otros grupos más expertos y que, por lo general, no han sido diseñados *ex-profeso* para ser integrados con los STIs con los que van a funcionar.

Además, a medida que las tecnologías evolucionan y que las necesidades de los STIs se van incrementando, se vuelve a hacer necesario sustituir el EV que se ha utilizado hasta el momento por otro más sofisticado, por lo que ésta es una necesidad que nunca deja de estar presente.

Como resultado, es necesario que, durante el diseño de la aplicación, los diseñadores puedan integrar un EV desarrollado independientemente del STI. Aparte de ofrecer la posibilidad de acceder a la información que sea necesaria en cada caso, lo normal es que los distintos EVs trabajen con distintos formatos de la información (por ejemplo, criterios para interpretar cuáles son los ejes x, y, z) razón por la cual será habitual que se deba realizar un trabajo de traducción de los formatos de la información al integrar el EV con el STI.

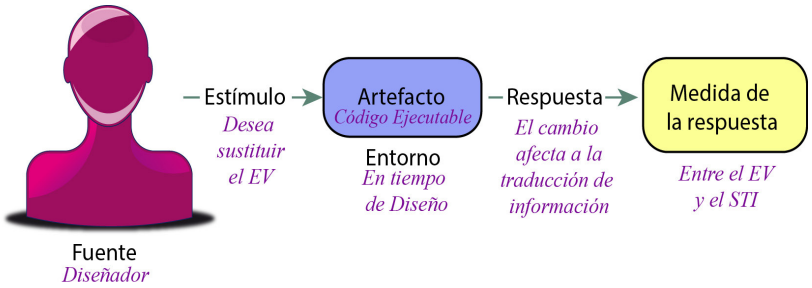


Figura 5.4: Escenario de sustitución del Entorno Virtual

Sustitución de la Estrategia de Tutoría

La estrategia de tutoría es una de las partes fundamentales del STI y una de las que más evoluciones sufrirán a lo largo de todo el ciclo de vida, pues prácticamente cualquier mejora en los demás elementos del prototipo podrá ser aprovechada para modificar la estrategia de tutoría. Por tanto, es bastante probable que cualquier cambio en otras partes del prototipo puedan afectar a la estrategia y, de forma inversa, que los cambios en ella puedan afectar su relación con otras partes del prototipo.

Mientras que, como se ha dicho, la estrategia de tutoría debería aprovechar las mejoras en otras partes del prototipo para mejorarse, lo deseable es que la situación no se reproduzca a la inversa, es decir, que estos cambios en la estrategia de tutoría no disparen a su vez cambios en otras partes del prototipo, ya que la modificabilidad quedaría seriamente afectada.

Lo que se va a tener en cuenta, por tanto, es que los cambios en la estrategia de tutoría se realicen dentro del marco definido por las funcionalidades ofrecidas por otras partes del prototipo. Lo que se pretende es que la sustitución de la estrategia de tutoría se pueda realizar, en tiempo de diseño, sin necesidad de modificar nada más que los módulos encargados de realizar la traducción de información con los demás módulos del prototipo.

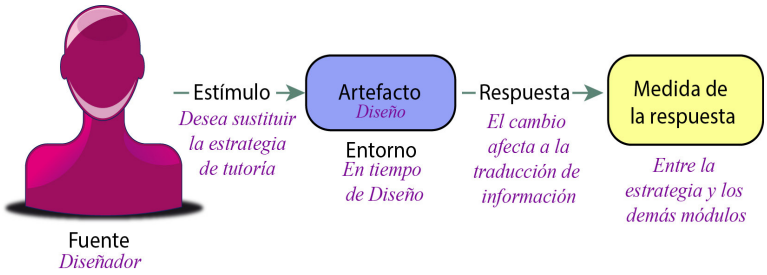


Figura 5.5: Escenario de sustitución de la estrategia de tutoría

Sustitución del Modelo del Estudiante

Una de las mejores formas de personalizar la tutoría, que también redunda en una mejora de la usabilidad del prototipo, es la realización y utilización de un modelo del usuario. En cuanto a la tutoría, es la manera de acercarla a la forma de aprendizaje del estudiante, además de servir para evaluar los conocimientos del mismo.

Así, se puede pasar de almacenar una traza de sus acciones, como medio de evaluarlo, a utilizar esa traza para decidir cómo responder a sus preguntas, si se le ofrece ayuda de manera proactiva o para valorar si está prestando atención a una explicación o si muestra señales de aburrimiento.

Los modelos de estudiante se pueden realizar de maneras completamente distintas, como pueden ser los sistemas basados en reglas o las redes bayesianas[1] (Russell y Norvig, 1995), en función de lo que se desee obtener con el modelado del estudiante y de la eficiencia de las operaciones que se le soliciten (una red bayesiana suele ser más eficiente que un sistema basado en reglas, pero lo normal es que el modelo del estudiante no sea tan rico en información).

Dependiendo de lo sofisticado de la estrategia de tutoría y del modelo del estudiante, este último puede variar entre ser un elemento completamente pasivo o constituir un componente que tome parte activa en la toma de decisiones de la tutoría. Por ello, un cambio en el modelo del estudiante puede ir de la mano de un cambio en la estrategia de tutoría, lo cual supone un una variante como para ser objeto de atención especial.

Por tanto, lo que aquí se considera es una sustitución del modelo del estudiante existente por otro que ofrezca una funcionalidad similar y que vaya acorde con el funcionamiento marcado por la estrategia de tutoría existente.

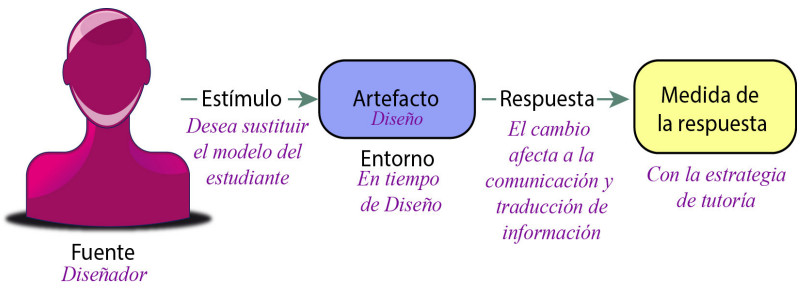


Figura 5.6: Escenario de sustitución del modelo del estudiante

1 Diríamos que una red bayesiana es un conjunto de variables, una estructura gráfica conectando estas variables y un conjunto de distribuciones de probabilidad condicional. Simplificando, podemos resumir que es un *grafo*; esto es, una representación gráfica de un problema.

Sustitución del Planificador

Otra forma de enriquecer la tutoría consiste en poder generar ejercicios de manera dinámica y, también, en resolver ejercicios o situaciones planteados por el alumno. Esto requiere disponer de un mecanismo de resolución de problemas en lugar de una batería de problemas resueltos, lo que en el caso de tareas procedimentales se traduce la necesidad de disponer de un planificador que proporcione la secuencia en la que deben realizarse las acciones.

En el caso más sencillo, el planificador puede consistir en la asociación de un problema con su solución. A partir de ahí, se pueden requerir mayores capacidades, como poder planificar un procedimiento lineal a partir de una situación inicial y unos objetivos. Posteriormente, se podrían añadir procedimientos con varias soluciones válidas, cuya diferencia sea, por ejemplo, el orden en que se realizan las acciones. Luego, se puede necesitar que se planifiquen acciones paralelas, imponer restricciones temporales o que el planificador sea capaz de realizar operaciones aritméticas sencillas.

También puede necesitarse un cambio de planificador por motivos de eficiencia. Hay implementaciones de algoritmos de planificación que son más eficientes que otras, y un planificador que sea capaz de realizar replanificaciones, en general, también resultará más eficiente que uno que no lo sea y necesite llevar a cabo una nueva planificación desde cero.

Algunas de las modificaciones descritas no implican cambios en la comunicación entre el planificador y quien use sus servicios, salvo en el formato de la información que se intercambia. Otros cambios, como la planificación de acciones paralelas, pueden implicar modificaciones en el tipo de información intercambiada, por lo que se requerirá que se afecte únicamente a quienes solicite los servicios del planificador (Figura 5.7):

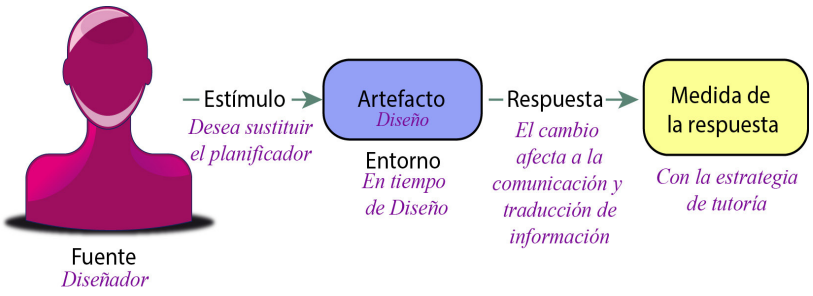


Figura 5.7: Escenario de sustitución del planificador

Sustitución del Planificador de Rutas

Un planificador de rutas es un elemento que cobra especial relevancia cuando se utiliza un EV como parte integrante de un sistema de enseñanza. Su uso sirve a varios fines, como el cálculo de un camino entre dos puntos cuando hay que mover un personaje por el escenario, o como medio para evaluar si el alumno está siguiendo la mejor ruta posible entre dos lugares dentro del escenario. Esto último es especialmente relevante en determinados escenarios, como una central nuclear, donde uno de los objetivos que se le pueden exigir a un alumno es aprender a moverse por las zonas de la central en las que las radiaciones recibidas sean mínimas, lo cual no siempre implica seguir el camino más corto entre dos puntos.

Los métodos de cálculo de rutas varían en función de las características de los escenarios en los que tiene lugar la enseñanza, especialmente en función de su tamaño y dinamismo y de las necesidades del contexto. Así, en los casos más simples, con un escenario pequeño, estático en el que siempre se busque el camino más corto, una representación del escenario en forma de grafo sobre el que se ejecuta un algoritmo simple como el algoritmo de Dijkstra (1959) que para este tipo de casos, suele ser suficiente. En escenarios de mayor tamaño o en los que existan elementos móviles que pueden bloquear un camino que previamente era factible, suele ser necesario optar por otro tipo de algoritmos como el sugerido que es el A* (Nilsson, 1971) o los diagramas de Voronoi (1907).

En cualquiera caso, lo que se le puede pedir a un planificador de rutas es que acepte las coordenadas que maneja el STI y que proporcione una solución de las mismas características, aunque puede ser necesario modificar la representación del escenario que usa el planificador de rutas.

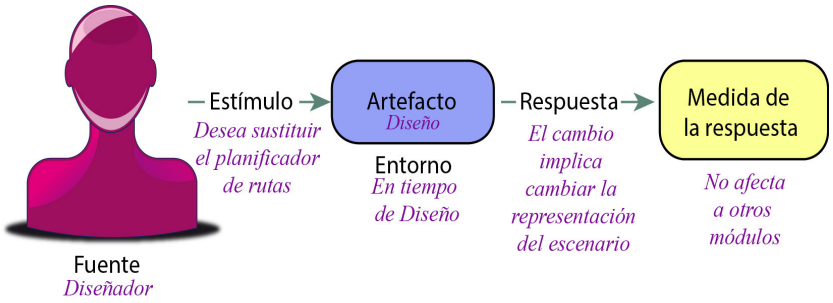


Figura 5.8: Escenario de sustitución del planificador de rutas.

Sustitución del Simulador

Cuando se promueve y se busca el aprendizaje en un entorno complejo, puede ser necesario hacer uso de un elemento que simule las características de dicho entorno, ya sea como respuesta a las acciones realizadas por el estudiante o como fruto de la dinámica habitual en un entorno de esas características. Así, por ejemplo, en un entorno de entrenamiento médico, será necesario simular la evolución del estado de un paciente, quien irá reaccionando ante las acciones del estudiante, irá empeorando su estado si no se le atiende o si las medidas adoptadas no son correctas, o cambiará su estado como resultado de condiciones impuestas por el STI para que el estudiante haga frente a determinadas situaciones.

A diferencia de lo que sucede con otros elementos descritos, el simulador es mucho más dependiente del contexto en el que se realice la enseñanza, por lo que resulta más complicado determinar las características que puede tener un simulador o qué es lo que se puede esperar de él. Además, se puede dar el caso de que el simulador esté disponible de manera separada o que se encuentre integrado ya en un EV.

Otra dificultad adicional es que, dependiendo del estado de la simulación en cada momento, puede darse el caso de que no se puedan llevar a cabo las acciones realizadas por los estudiantes, o que la situación que determine modifique el estado del EV e influya en el proceso de planificación.

Son tantas las variables que se presentan en el caso de los simuladores que habrá que intentar reducir al mínimo, el número de módulos que dependen de él, de manera que un cambio en el simulador afecte al menor número de módulos que sea posible. Figura 5.9:

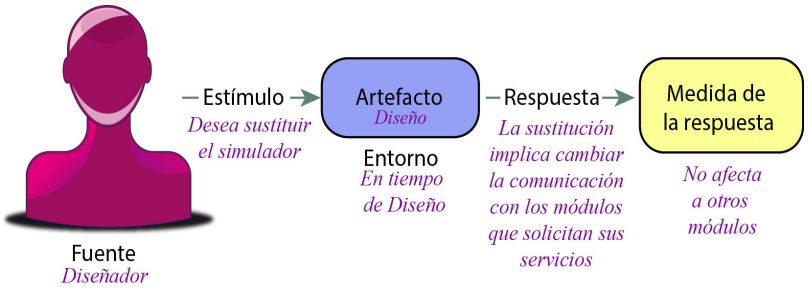


Figura 5.9: Escenario de sustitución del simulador

Deshabilitación de la Tutoría

Existen muchas razones que pueden llevar a que los usuarios soliciten la eliminación de las funciones de tutoría, al menos de manera visible. El caso más claro es quizá la realización de un

examen por parte de un alumno, situación en la que el diseñador del curso puede querer evaluar las aptitudes del alumno sin ningún tipo de ayuda como la que prestaría un tutor virtual. Una Situación similar podría darse si existen distintos niveles de dificultad para resolver un ejercicio, donde en el nivel más sencillo se prestaría al estudiante toda la ayuda posible y en el más complejo se le dejaría con completa libertad para hacer lo que desee. También puede darse el caso de que, previa a la realización de una actividad, se ofrezca al alumno la posibilidad de moverse por el escenario para familiarizarse con él.

Una funcionalidad más elaborada puede ser que el STI detecte si el alumno rechaza siempre la ayuda, lo cual podría llevarle a decidir que no va a ofrecerla más, salvo que el alumno lo demander. Puede suceder también que un alumno encuentre excesivamente molestas las interrupciones del tutor para hacerle preguntas o para ofrecerle ayuda, de manera que puede ser el propio alumno quien decida deshabilitar estas funciones, ya sea con este fin o para comprobar él mismo cuál es su nivel de destreza en la resolución de una determinada situación.

Para cualquiera de estas situaciones se hace necesario poder deshabilitar las funciones de tutoría, bien al arrancar la aplicación o con el prototipo ya trabajando. Esto debe poder realizarse prácticamente al instante y sin mucho esfuerzo, sin necesidad de modificar el código de la aplicación sino cambiando alguna opción de configuración o solicitándolo a través de un menú o algún otro mecanismo similar.

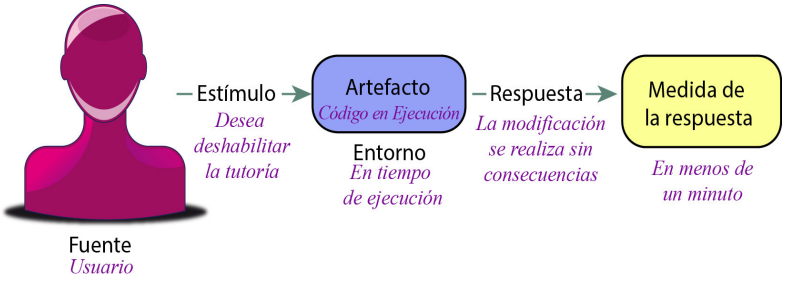


Figura 5.10: Escenario de deshabilitación de la tutoría

Deshabilitación del Cálculo de Trayectorias

La planificación o cálculo de rutas es un elemento de uso común en los EVs, tengan estos fines educativos o no. Sin embargo, ni todos los dominios requieren del cálculo de rutas, ni todas las actividades que se puedan plantear en un dominio tienen por qué requerirlo tampoco.

Por último, igual que en el inciso anterior, se le puede ofrecer al alumno la posibilidad de explorar un escenario antes de realizar una actividad, lo cual tampoco requeriría del funcionamiento

del cálculo de rutas. Por estas razones, resulta ideal que pueda deshabilitarse el cálculo de trayectorias, especialmente en función de las características de la actividad a realizar.

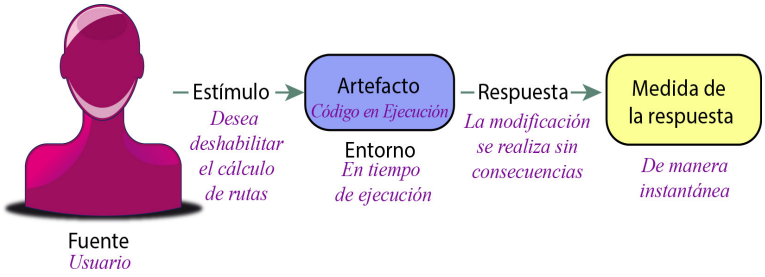


Figura 5.11: Escenario de deshabilitación del cálculo de trayectorias

5.3.2. Valores de Eficiencia

Escenarios de Eficiencia

Como se mencionó al inicio, en una aplicación de enseñanza basada en realidad virtual es de especial importancia que la aplicación responda a las acciones del usuario de manera bastante inmediata. De no ser así, la experiencia del usuario puede no resultar satisfactoria y el interés se perderá de forma rápida. Sin embargo, no todas las acciones que realiza el usuario pueden medirse de la misma forma y si bien es cierto que la navegación por el escenario debe realizarse de manera ágil, la respuesta a otras acciones sí puede admitir un retardo, aunque no debe ser excesivo.

Los escenarios que siguen la forma general mostrada en la Figura 5.12, recogen los aspectos mencionados.

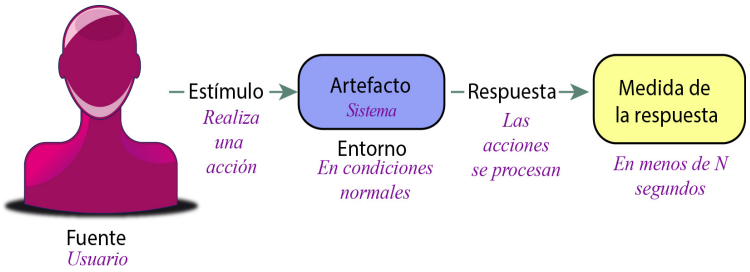


Figura 5.12: Escenario general de eficiencia.

Respuesta ante Acciones

Cuando un alumno tiene que realizar acciones dentro del EVEATI como parte de una actividad, éstas deben procesarse de distintas formas, ya que, por ejemplo, la estrategia de tutoría debe decidir si permite su realización y pueden causar algún efecto en el simulador. A su vez, habitualmente se debe generar una respuesta, como no permitir su realización, advertir al estudiante de algo o disparar algún evento dentro del prototipo.

Como el procesamiento de una acción puede ser complejo, existe la posibilidad de que no pueda darse una respuesta en tiempo real, lo cual no es estrictamente necesario, pero el retardo debe ser lo suficientemente pequeño como para que el alumno no piense que no se ha ejecutado la acción, pues esto le llevaría a repetirla cuando en realidad está siendo procesada. Tampoco debe darle la sensación de que la ejecución de la aplicación es lenta, ya que puede desmotivarlo para continuar utilizándola.

Por tanto se requerirá de la aplicación que, independientemente de lo que tenga que hacer con motivo de la ejecución de una acción, le de una respuesta al alumno en el menor tiempo, por ejemplo en menos de un segundo, sin importar que una misma acción pueda disparar más de una respuesta.

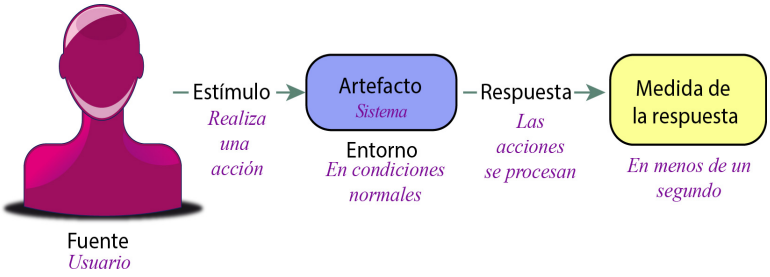


Figura 5.13: Escenario de eficiencia ante una acción

Respuesta en Navegación

Cuando un usuario se mueve por el EVEATI, es necesario registrar las coordenadas del tutor por si resulta necesario hacer un seguimiento de sus movimientos. Sin embargo, esto no debe afectar en absoluto al usuario, pues la percepción de un movimiento fluido es uno de los factores clave a la hora de que los alumnos tengan una buena experiencia con el prototipo y continúen utilizándolo para su formación. Por tanto, se requerirá que el muestreo de coordenadas se lleve a cabo sin disminuir la velocidad de ejecución del prototipo, de forma que el usuario no note que esta acción se está llevando a cabo.

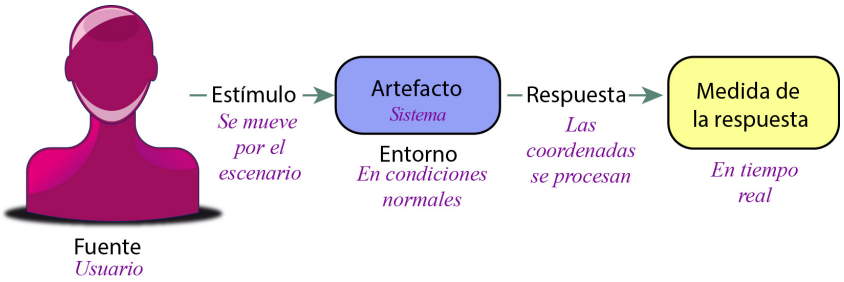


Figura 5.14: Escenario de eficiencia en navegación.

5.4. Documentación de la Arquitectura

5.4.1. Descripción de la Documentación Necesaria

De acuerdo a Clements y otros (2002a), la documentación de una arquitectura software es un elemento fundamental para la comunicación, tanto dentro del equipo de desarrollo como con otros individuos que, de una u otra manera, se ven afectados por el desarrollo del sistema, como pueden ser los usuarios o los diseñadores de software que se puede conectar con el prototipo en desarrollo. Por este motivo, resulta de especial relevancia elaborar una documentación adecuada a cada uno de los posibles afectados, lo cual implica reflejar los aspectos que necesita conocer cada uno de ellos con un nivel de detalle suficiente, pero no demasiado profundo para no dificultar la comprensión del documento.

Los autores del trabajo mencionado en el párrafo previo, proponen la documentación de una arquitectura software a través del uso de distintas vistas de la arquitectura, las cuales reflejan los diferentes aspectos que puede ser necesario conocer. Además, sugieren una serie de vistas que pueden ser utilizadas que permiten documentar una amplia gama de arquitecturas. Esto no impide que el arquitecto software genere nuevas vistas o combine algunas de las existentes si lo considera necesario. A continuación se describen brevemente las vistas propuestas.

Las vistas de los módulos reflejan aspectos estáticos de la arquitectura a través de distintos tipos de descomposiciones. Por este motivo, resultan útiles para obtener una primera visión de conjunto de la aplicación. Además, la descripción de las responsabilidades de cada módulo son útiles para realizar un análisis de trazabilidad de los requisitos. También pueden resultar de utilidad para realizar análisis de impacto, de forma que ayudan a ver qué elementos de la arquitectura se pueden ver afectados por una modificación. Los estilos arquitectónicos incluidos en este tipo de vista son:

Descomposición: representa la descomposición del prototipo en subsistemas, y de éstos a su vez en otros subsistemas, y así sucesivamente. Este estilo se utiliza para dar una visión general del prototipo y sus partes, y su uso está especialmente recomendado para empezar a entender la totalidad de proyecto y para la comunicación con los responsables del proyecto.

Uso: muestra la dependencia de unos módulos respecto de otros en cuanto a su relación de uso. Este estilo resulta de utilidad para planificar desarrollos iterativos, extensiones al sistema y análisis de impacto.

Generalización: muestra, de una forma similar a la herencia del diseño orientado a objetos, qué subsistemas extienden la funcionalidad de otros, por lo que es un estilo que suele utilizarse en el diseño de arquitecturas orientadas a objetos.

Capas: estructura los módulos del prototipo en forma de una serie de capas superpuestas, de tal manera que las capas más altas pueden utilizar los servicios proporcionados por las capas más bajas de acuerdo con unas reglas predefinidas (por ejemplo, que sólo pueden utilizarse los servicios de la capa inmediatamente inferior). Es un estilo adecuado para la creación de máquinas virtuales, las cuales facilitan la reutilización y la portabilidad.

Las vistas de componentes y conectores reflejan el comportamiento del prototipo mediante el intercambio de mensajes entre distintos componentes a través de determinados conectores. Dependiendo de la vista utilizada, los componentes puede corresponder a los módulos utilizados en las vistas de módulos, aunque no tiene que ser necesariamente de esta forma. Debido al aspecto dinámico que incorporan, estas vistas sirven para analizar atributos de calidad como fiabilidad, disponibilidad y rendimiento. Las vistas incluidas en este proyecto son:

Tuberías y filtros: es un tipo de vista que está especialmente indicado cuando existen unos datos que van a ir sufriendo sucesivas transformaciones.

Cliente-servidor: muestra la interacción de elementos a través de la solicitud de servicios de unos a otros. Los servidores proporcionan los servicios a través de distintas interfaces, y los clientes utilizan los servicios proporcionados por esas interfaces. Este estilo desacopla a los productores y a los consumidores de datos y servicios.

Peer-to-peer: se caracteriza por el intercambio de servicios entre los distintos elementos de la arquitectura; cualquier elemento puede solicitar los servicios ofrecidos por otro. En este estilo es habitual que los componentes dispongan de la información más actualizada

que necesitan, lo que facilita la distribución de los componentes, reduce la carga de los mismos cuando actúan como servidores y reduce la necesidad de comunicación para actualizar datos.

Procesos comunicados: se caracteriza por la interacción de distintos componentes que se ejecutan de manera concurrente. Sirve para comprender la dinámica de sistemas que trabajan en paralelo, por lo que también puede ser utilizado para realizar análisis de eficiencia y factibilidad. Se suele usar en combinación con otros estilos, como peer-to-peer o cliente-servidor.

Editor-suscriptor: se caracteriza por la existencia de elementos que interactúan a través del anuncio de eventos por parte de unos y la subscripción a dichos eventos por parte de otros. Es un estilo adecuado para desacoplar a los productores y a los consumidores de los eventos. Este desacoplamiento permite que los productores y los consumidores no se conozcan directamente hasta que el prototipo se está ejecutando, lo que facilita la modificación tanto de productores como de consumidores. Suele presentarse en combinación con el estilo peer-to-peer.

Datos compartidos: este estilo se centra en la representación de datos persistentes a los que acceden distintos subsistemas. Puede servir para desacoplar a los productores y a los consumidores de datos, ya que el acceso al repositorio de datos posibilita que los productores y los consumidores no se conozcan directamente. Si el almacén de datos es pasivo se le da el nombre de repositorio, mientras que si avisa a los consumidores de datos de los cambios que se producen se le da el nombre de pizarra (en este caso, se convierte en un estilo similar al editor-suscriptor, aunque en este último no se da la persistencia de los datos). Es una representación adecuada para sistemas de información y para sistemas basados en conocimiento. Este estilo suele aparecer en combinación con el cliente-servidor.

Las vistas de asignación, por su parte, describen la correspondencia entre elementos software y elementos del entorno:

Despliegue: establece una correspondencia entre los elementos software de la arquitectura y los elementos hardware que soportarán su ejecución, y su utilización es especialmente adecuada para la realización de análisis de rendimiento, seguridad y fiabilidad.

Implementación: establece la correspondencia entre los elementos software y la infraestructura de desarrollo, siendo adecuada su utilización, por ejemplo, para establecer el sistema de gestión de configuración.

Asignación de trabajo: se utiliza para indicar a qué miembros del grupo de trabajo les corresponde el desarrollo de los distintos elementos presentes en la arquitectura.

Además de las vistas anteriores, existe documentación adicional que también es conveniente tener en consideración, aunque queda a criterio del arquitecto software qué es relevante y qué no lo es, en función de a quién vaya dirigida, del tipo de prototipo en construcción o de los atributos de calidad considerados:

Diagrama de contexto: dentro de la documentación de la arquitectura software, un diagrama de contexto es aquél que muestra la visión de un elemento junto con los elementos con los que éste se relaciona. Un caso de especial relevancia es aquél en el que el elemento considerado es el propio sistema. En este caso, el diagrama de contexto muestra el alcance del prototipo, estableciendo qué es lo que queda dentro y qué es lo que queda fuera, así como las interacciones con el exterior. No tiene por qué existir un único diagrama de contexto con estas características, sino que puede existir uno por cada vista que proporcione una perspectiva diferente de la relación del prototipo con el exterior.

Decisiones de diseño: puesto que el diseño de la arquitectura es un proceso iterativo, en muchas ocasiones se contemplan soluciones que posteriormente se desechan por distintos motivos. Suele ser conveniente documentar las razones por las que se ha optado por una solución, así como las soluciones descartadas y los motivos para descartarlas, de manera que en posteriores revisiones de la arquitectura se cuente con esa experiencia como criterio para seguir descartando una solución u optar por ella en función de cambios que hayan podido suceder.

Mapeo de vistas: una de las tareas importantes a la hora de documentar la arquitectura consiste en decidir qué vistas se utilizan. Decidido esto, es conveniente documentar la relación existente entre las distintas vistas para que no dé la sensación de que se refieren a sistemas completamente distintos. Una forma de hacerlo consiste en combinar varias vistas en una sola, aunque muchas veces ésta no es la solución más aconsejable, ya que pueden aparecer diagramas demasiado complejos. En esos casos lo más conveniente es mantener las vistas por separado y elaborar una tabla que muestre qué elementos de una vista se corresponden con los elementos de otra de las vistas.

Guía de variabilidad: esta guía se escribe para documentar aspectos del prototipo que pueden cambiar en función de distintos factores. Estos cambios pueden ser de dos tipos: variabilidad y dinamismo. La variabilidad se da cuando algún elemento de la arquitectura puede sustituirse por otro. Este caso ocurre, por ejemplo, cuando todavía no se ha tomado

una decisión respecto a esa parte de la arquitectura, pero ya se han explorado distintas opciones, cuando se planea construir una familia de sistemas y esa parte de la arquitectura cambia en cada miembro de la familia o cuando se está diseñando un *framework* que contiene puntos en los que puede extenderse. En cualquiera de los casos, es necesario definir cuáles son los puntos de variación, cuáles son los elementos afectados por el cambio y en qué momento se produce (diseño, compilación, ejecución, etc.).

Por otro lado, el dinamismo, es una propiedad del prototipo en ejecución. En general tiene que ver con la creación y eliminación de distintos componentes, en función de los servicios requeridos por los usuarios. También se produce dinamismo cuando los componentes del prototipo pueden cambiar dinámicamente de procesador para, por ejemplo, mejorar el rendimiento de la aplicación.

Interfaces: la interfaz de un elemento es el medio por el cual interactúa con su entorno, considerando la interacción como cualquier cosa que hace un elemento que pueda afectar a otro. La información acerca de ella depende de la vista donde se documente dicha interfaz. En general, son bidireccionales, por lo que es importante documentar no sólo lo que un elemento ofrece al exterior, sino también lo que ese elemento necesita del exterior. Además, dentro de una misma vista, un elemento puede tener más de una interfaz para, por ejemplo, restringir los servicios a los que otros elementos pueden acceder. Además, la existencia de múltiples interfaces facilita su evolución.

Análisis: en este apartado se documenta cualquier tipo de análisis que se realice sobre la arquitectura, como los resultados de una evaluación de la seguridad o del impacto de realizar alguna modificación.

5.4.2. Selección de la Documentación a Generar

Clements y los otros autores (2002) proponen, a partir de la documentación descrita, un pequeño proceso para decidir qué documentos resultan de utilidad y cuáles no. En primer lugar, sugieren la creación de una tabla como la mostrada en la Figura 5.15, donde se observan los tipos de documentación que se pueden generar y los perfiles afectados por el prototipo considerado. En las casillas se representa la documentación que es necesario generar para cada perfil y el nivel de detalle adecuado para cada uno de ellos.

Los autores del citado trabajo sugieren combinar y eliminar distintas vistas, proporcionando unas guías adicionales para filtrar los resultados obtenidos a partir de la tabla, de manera que

la documentación generada no sea excesiva y continúe resultando de utilidad. Así, por ejemplo, indican que en sistemas de tamaño pequeño o medio, como el que nos ocupa, pueden no documentarse las vistas de implementación y asignación de trabajo, pues con toda probabilidad se pueden solapar con la de descomposición. Además, dependiendo del tipo de prototipo, habrá vistas que no resulten de utilidad, por lo que no será necesario generar la documentación correspondiente a las mismas.

Como tercer paso se propone realizar una priorización de la documentación según su importancia, en función de restricciones de tiempo o presupuestarias. Puesto que esta situación no es aplicable en el presente trabajo, se ha omitido la realización de este paso.

El resultado de la realización de este proceso genera la documentación que se va a explicar a continuación. Es importante señalar que a medida que se diseña y refina la arquitectura pueden aparecer nuevas necesidades en lo que a documentación se refiere, por lo que lo que aquí se presenta es el resultado final:

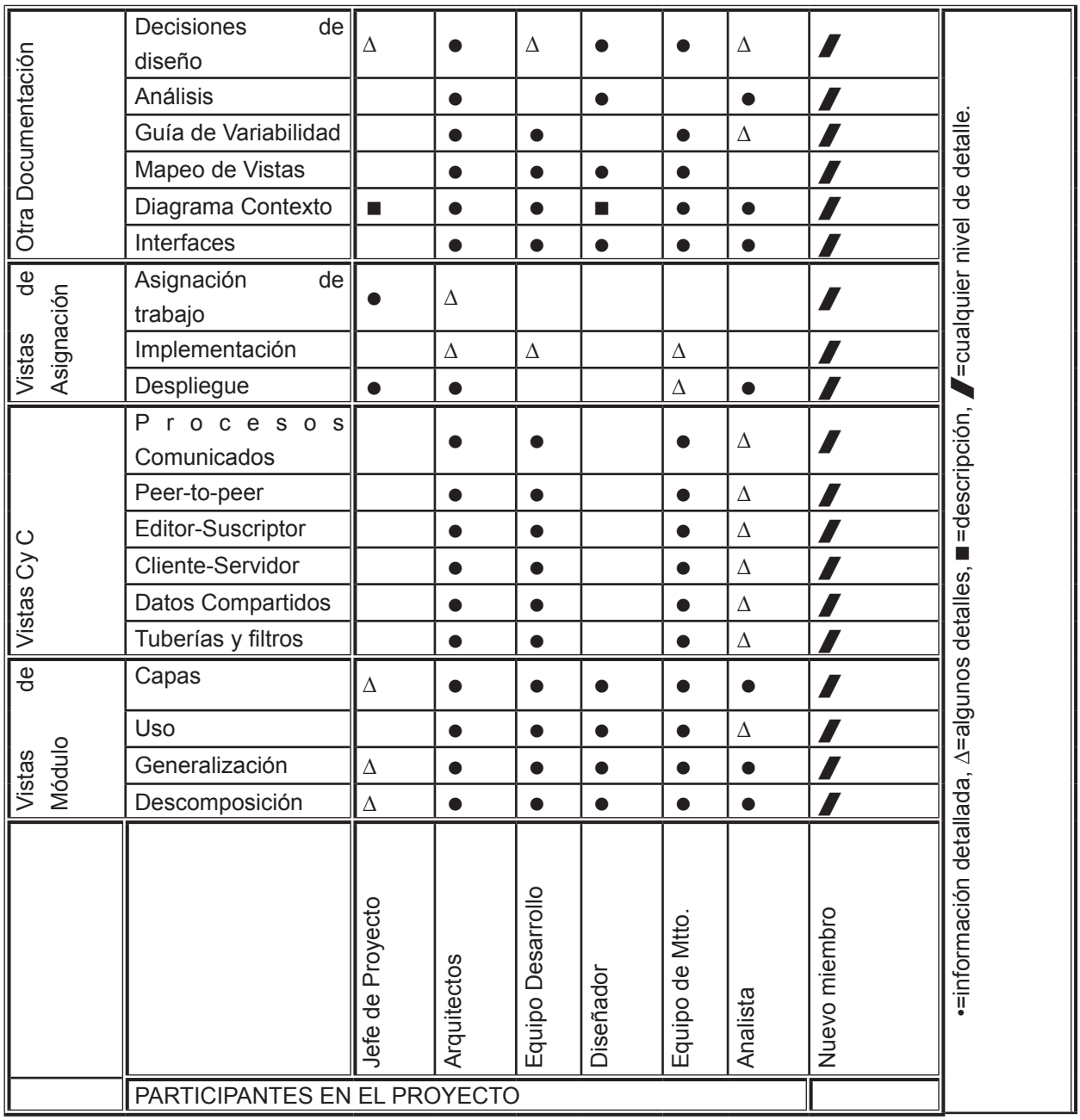


Figura 5.15 Documentación de las vistas

Vista de descomposición: con esta vista se mostrarán los elementos de los que se compone el prototipo. Se hace importante porque en ella se fijarán los principales criterios de descomposición del prototipo en distintos módulos, por lo que gran parte de la modificabilidad dependerá de lo adecuado de las decisiones adoptadas para realizar la descomposición.

Vista de uso: se muestran las dependencias existentes entre los elementos de la vista anterior. Esta vista resulta de especial utilidad para analizar la modificabilidad del prototipo, por lo que, se cuidará mucho su documentación con el fin de que contribuya a resaltar esa cualidad.

Vista editor-suscriptor: el uso de editores y suscriptores es una de las técnicas que mencionan los autores de (Bass, y otros, 2003) como adecuadas para conseguir aplicaciones modificables, y es una de las utilizadas para el diseño de la arquitectura software propuesta. Por ello, deberá ser documentada convenientemente a través de esta vista.

Vista peer-to-peer: como se explicará más adelante, la aproximación utilizada en esta arquitectura, no va a ser jerárquica, sino que cada agente solicitará los servicios de otros directamente, sin utilizar intermediarios. De esta manera, todos los agentes tendrán la misma categoría, por lo que se ha considerado que esta vista es la más adecuada para reflejar esta característica.

Vista de despliegue: esta vista se utilizará para proponer una posible instalación del prototipo resultante de utilizar la arquitectura propuesta, teniendo en cuenta las características de las aplicaciones que se le piensan dar al prototipo resultante.

5.4.3. Plantillas

Para documentar cada una de estas vistas se dividirá la documentación en paquetes. Esta división puede realizarse atendiendo tanto al tamaño de la parte del prototipo a documentar como al nivel de profundidad en el que nos adentremos en la arquitectura. La organización de la documentación de cada vista o paquete será la que se indica a continuación:

Presentación de la vista: muestra los elementos existentes en la parte de la vista representada por este paquete y las relaciones existentes entre ellos. Se propuso de manera gráfica.

Catálogo de elementos: contiene, al menos, los elementos mostrados en la presentación de la vista. De forma particular contiene:

Elementos y sus propiedades: enumera cada uno de los elementos de la vista o paquete y describe sus propiedades.

Relaciones y sus propiedades: cada vista representa una relación distinta entre los elementos que contiene. Si la presentación de la vista no contiene todas las relaciones o si existen excepciones a lo mostrado en la presentación de la vista, en esta sección se recoge la información pertinente.

Interfaces: en esta sección se documenta tanto lo que proporciona un elemento como lo que necesita del entorno.

Comportamientos: aquí se documentan interacciones complejas o comportamientos útiles de cara al análisis del prototipo.

Diagrama de contexto: muestra cómo se relaciona el elemento descrito con los elementos con los que debe interactuar.

Guía de variabilidad: muestra cómo puede tener lugar cualquier forma de variación o dinamismo previstas sobre el elemento descrito.

Información sobre la arquitectura: recoge información acerca de las decisiones de diseño que se han tomado para que la arquitectura haya dado el resultado que se muestra. El objetivo principal es justificar por qué se han tomado determinadas decisiones y mostrar que se ha hecho de manera razonada. Lo mínimo que debe considerar son:

Decisiones de diseño: se describen las decisiones de diseño adoptadas.

Análisis de resultados: recoge los resultados de las evaluaciones que se realicen sobre la arquitectura, como por ejemplo una lista de cambios necesarios para una determinada modificación.

Vistas relacionadas: en general, la vista o paquete del que se deriva la vista o paquete actual (padre), los que se derivan de él (hijos) o los que se derivan del mismo que él (hermanos). En cada una de las vistas sólo se documentarán, de entre los puntos anteriores, aquellos que se consideren aplicables o necesarios.

5.5. Diseño de la Arquitectura

A continuación se presentan las distintas vistas de la arquitectura del prototipo atendiendo a la selección de documentación realizada en la sección 5.4.2 y con el formato de la plantilla descrita en la sección 5.4.3. Como se refiere en el punto anterior, se ven los paquetes de la vista de descomposición, seguidos por los de la vista de uso, los de la vista editor-suscriptor, la vista peer-to-peer y, finalmente, la vista de despliegue.

Cabe señalar que para la representación de la Arquitectura, todos los diagramas que se muestran, se han realizado basados en la notación UML[2] aunque para mejorar su compresión se hecho la traducción y mejorado su aspecto.

5.5.1. Descomposición de la Arquitectura en vistas

En esta vista se muestra, de forma progresiva, la descomposición del prototipo en subsistemas y de éstos en módulos. No se reproduce el comportamiento de los módulos, el cual se podrá ver en las vistas editor-suscriptor y peer-to-peer

Sistema
Presentación de la vista

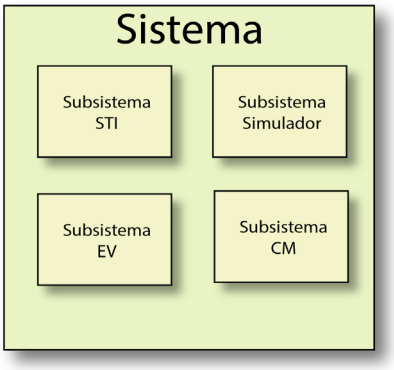


Figura 5.16: Descomposición del prototipo

Catálogo de elementos
B.1. Elementos y sus propiedades

2 El Lenguaje Unificado de Modelado (UML, *Unified Modeling Language*) es un lenguaje estándar, basado en una notación gráfica, que se utiliza para modelar software. Es un lenguaje de propósito general que pretende ser un estándar mundial y se utiliza para visualizar, especificar, construir y documentar las diferentes “piezas”de un prototipo.

Elemento	Propiedades
Entorno Virtual	El EV constituye la principal forma de interacción del usuario con la aplicación. Proporciona una interfaz tridimensional que reproduce el entorno real en el que se realiza el proceso de aprendizaje, con las licencias pertinentes debidas a restricciones técnicas o a ayudas inherentes. A través de la navegación por el escenario y la manipulación de los objetos que en él aparecen, el estudiante intentará realizar los ejercicios que se le planteen o bien trabajará en las actividades que requiere llevar a cabo.
Simulador	Reproduce el comportamiento de distintos elementos presentes en un EV, ya sea como respuesta a las acciones de los usuarios o como resultado de la evolución o funcionamiento normal de cada elemento en cuestión.
Sistema de Tutoría Inteligente	En este subsistema reside todo lo relativo al proceso de tutoría. Es el encargado de registrar y procesar las acciones del usuario, decidir si son correctas y dar una respuesta adecuada a las mismas. También se encarga de procesar las preguntas de los estudiantes y proporcionarles una respuesta adecuada en función de los parámetros manejados por la estrategia de tutoría. Es dentro de este subsistema donde se realizan la planificación y el cálculo de rutas.
Centro de Mensajes	Es el encargado de manejar la comunicación entre los otros subsistemas. Funciona con un mecanismo de suscripción. Cada subsistema se suscribe a los tipo de mensajes que quiere recibir, y los mensajes enviados por cada subsistema sólo los reciben aquellos que están suscritos a esos mensajes. El Centro de Mensajes no conoce *a priori* los tipos de mensajes que puede recibir ni de quién, por lo que admite todas las suscripciones que reciba.

Tabla 5.1: Elementos y propiedades del prototipo

Es importante señalar que el prototipo, como tal, no existe como elemento, por lo que todos los elementos descritos son visibles por otros elementos con las restricciones impuestas por la utilización del centro de mensajes.

B.2. Relaciones y sus propiedades

La relación que dirige la descomposición de esta vista es “forma-parte-de”. No hay excepciones ni añadidos a lo que define esta relación.

B.3. Interfaces

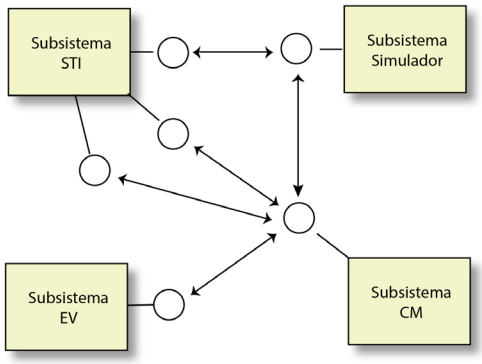


Figura 5.17: Interfaces de los módulos del prototipo

Elemento	Interfaces
Entorno Virtual	Recibe comandos para mover un avatar por el escenario, así como otros para realizar acciones dentro del mismo. Proporciona las posiciones y orientaciones actuales de los objetos presentes en un escenario, así como las acciones realizadas por los usuarios.
Simulador	Recibe las acciones que afectan al estado de la simulación. Proporciona los resultados de las acciones y permite consultar el estado de los elementos que forman parte de la simulación.
Sistema de Tutoría Inteligente	Recibe las notificaciones de conexión de estudiantes y, para cada uno de ellos, sus acciones, preguntas y movimientos. Proporciona distintas respuestas a las acciones de los estudiantes en función de los parámetros manejados por la estrategia de tutoría. También facilita respuestas a sus preguntas y proporciona pistas para ayudar a los estudiantes a resolver los ejercicios.
Centro de Mensajes	Recibe peticiones de suscripción y desuscripción a distintos tipos de mensajes. Proporciona a los suscriptores los mensajes recibidos de otros subsistemas conectados a él, siempre y cuando sean de alguno de los tipos a los que se han suscrito. La manera correcta de utilizar el Centro de Mensajes es suscribirse a los mensajes que se desee recibir y desuscribirse una vez no se desee recibirlos más.

Tabla 5.2: Elementos e interfaces del sistema

Todas las interfaces se definen de forma más concreta en el paquete correspondiente a cada módulo.

B.4. Comportamientos

Para el caso del prototipo en cuestión, este aspecto se considera no aplicable.

Diagrama de contexto

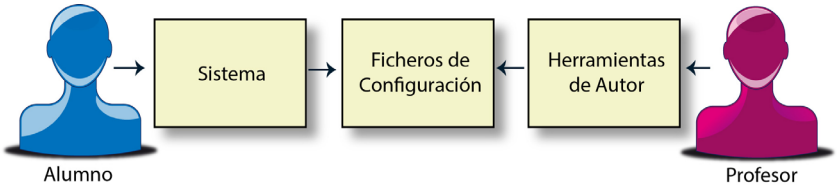


Figura 5.18: Diagrama de contexto del prototipo.

Guía de variabilidad

La conexión del simulador al prototipo es susceptible de variar en función de las características del mismo. Para mayor información, ver la guía de variabilidad en la sección 5.5.1.4. Existirá un único STI, un único simulador y un único Centro de Mensajes. Sin embargo, habrá tantos EVs como estudiantes estén realizando el proceso.

Información sobre la arquitectura

E.1. Decisiones de diseño

Para realizar esta descomposición se ha tenido en cuenta que algunos de los elementos mostrados, en concreto EV y Simulador, forman parte de los elementos susceptibles de sufrir sustituciones, por lo que se ha decidido que es mejor separarlos del resto de módulos del prototipo desde el principio, ya que constituyen, de por sí, aplicaciones con entidad propia. Para facilitar la independencia entre ellos y también su integración se ha pensado en utilizar un esquema editor-suscriptor (*ver sección 5.5.3*). Por esta razón aparece el Centro de Mensajes, en el cual los distintos elementos se suscribirán a los mensajes que quieran recibir. De esta forma los productores de información no son conscientes de quiénes son los consumidores, lo cual facilita la sustitución del EV y el simulador e incluso su conexión y desconexión en función de las necesidades del entrenamiento.

E.2. Análisis de resultados

Todos los resultados aparecen descritos en la sección 5.6.

E.3. Suposiciones

Para la realización de esta descomposición se supone que se dispone de una red de banda ancha, de forma que la comunicación a través de la red no supone un cuello de botella debido a la velocidad de la misma. Además, el número y tamaño de los mensajes que se intercambien no serán muy elevados, por lo que se espera que el Centro de Mensajes no sea un embudo que afecte las comunicaciones.

La descomposición también supone que los tres subsistemas que se conectan al centro de mensajes pueden funcionar sin alguno de los otros, o sin todos ellos, aunque la ausencia de las entradas provenientes de ellos suponga un funcionamiento más simple.

Puesto que la comunicación se realiza con un mecanismo de suscripción, todos los subsistemas asumen que existe al menos un productor de la información a la que se suscriben. De no ser así, los subsistemas deben tener alguna forma de funcionamiento en ausencia de esa información. Puesto que, de momento, la seguridad no es un problema, no se requiere suscripción al Centro de Mensajes para poder enviar mensajes, sino sólo para recibirlos.

5.5.1.2. Vista de Descomposición

Entorno Virtual

Presentación de la vista

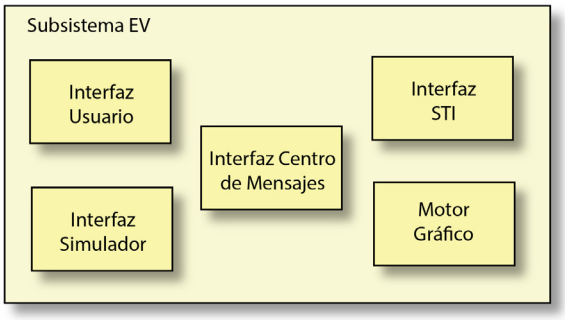


Figura 5.19: Descomposición del EV.

Catálogo de elementos

B.1. Elementos y sus propiedades

Elementos	Propiedades
Motor Gráfico	Es el módulo que se responsabiliza de cargar los escenarios 3D y mostrarlos por pantalla, así como de ir actualizando la escena mostrada en función de las entradas procedentes de cualquier actor que interactúe con el EV.

Interfaz de Usuario	Este módulo se encarga de trasformar las entradas proporcionadas por el usuario en llamadas a métodos del motor gráfico que transformen la escena. Debe soportar las distintas posibilidades de interacción con el usuario a través de los dispositivos que se determinen, como el teclado y el ratón.
Interfaz con el STI	Tiene como responsabilidad la traducción de los mensajes enviados desde el STI a llamadas a métodos del motor gráfico y viceversa.
Interfaz con el Simulador	Se encarga de transformar en llamadas a métodos del motor gráfico los mensajes recibidos desde el simulador y viceversa
Interfaz con el Centro de Mensajes	Se encarga de realizar la traducción de mensajes entre el centro de mensajes y los demás módulos del EV

Tabla 5.3: Elementos y propiedades del Entorno Virtual

B.2. Relaciones y sus propiedades

La relación que dirige la descomposición de esta vista se estructura de la siguiente manera: *forma-parte-de*...

Elementos	Propiedades
Motor Gráfico	Proporciona servicios para crear y destruir objetos, cambiar sus propiedades, como posición, orientación o color, e interactuar con ellos. También proporciona servicios para reproducir animaciones y la posibilidad de consultar los nuevos valores de las propiedades de los elementos. Sólo es visible desde dentro del EV.
Interfaz con el Usuario	Recibe las órdenes enviadas por los dispositivos de interacción y las traduce en llamadas a los servicios proporcionados por el motor gráfico, así como en mensajes hacia los demás subsistemas que forman la arquitectura. Puesto que sirve de comunicación con el usuario, esta interfaz es visible desde el exterior del subsistema.
Interfaz con el STI	Recibe los mensajes enviados por el STI y los traduce en llamadas a los servicios proporcionados por el motor gráfico. Sólo es visible desde dentro del EV.
Interfaz con el Simulador	Recibe los mensajes enviados por el simulador y los traduce en llamadas a los servicios proporcionados por el motor gráfico. Sólo es visible desde dentro del EV.
Interfaz con el Centro de Mensa jes	Proporciona tres interfaces de comunicación. La primera permite al Centro de Mensajes enviarle los mensajes a los que está suscrito el EV, mientras que las otras permiten a los demás módulos del subsistema comunicarle los datos que deben ser enviados a los demás subsistemas de la aplicación. Por tanto, la primera interfaz es visible sólo desde el exterior del subsistema, mientras que las otras sólo lo son desde el interior.

Tabla 5.4: Relaciones y propiedades del Entorno Virtual

B.3. Interfaces

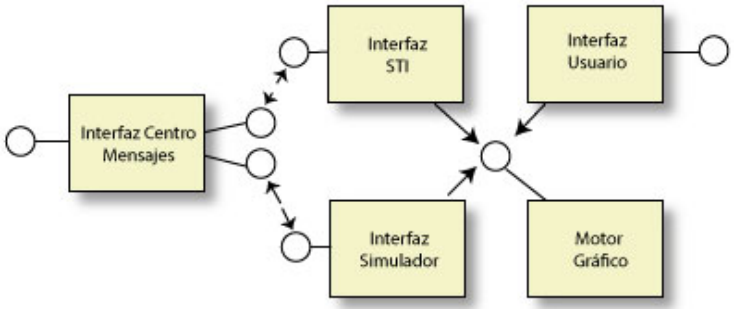


Figura 5.20: Interfaces de los Módulos del EV

B.4. Comportamientos

Para el caso del prototipo propuesto no es aplicable.

Diagrama de contexto

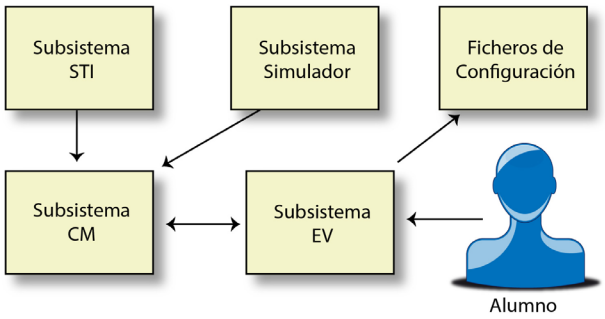


Figura 5.21: Diagrama de contexto del EV.

Guía de variabilidad

La descomposición variaría si el simulador se encuentra integrado con el EV. Dependiendo del simulador utilizado, existe la posibilidad de integrar los módulos de comunicación con el STI y con el simulador en uno solo. La posible variabilidad de los dispositivos de interacción entre el usuario y el EV se debe tener en cuenta al realizar el diseño de la interfaz con el usuario.

E. Información sobre la arquitectura

E.1. Decisiones de diseño

Para realizar la división en módulos se ha utilizado el Principio de Separación de Interfaces (ISP, Interface Segregation Principle) (Martin, 1999). Este principio establece que es preferible utilizar interfaces separadas para distintos tipos de clientes que una gran interfaz genérica que sirva a todos los clientes.

De esta manera, se han identificado como tipos de clientes potencialmente distintos al usuario, al STI y al simulador, por lo que se ha creado un paquete específico para cada uno de ellos que realice las funciones de interfaz con el EV. Estas interfaces serán las que realicen las labores de traducción entre las salidas producidas por cada tipo de cliente y las entradas aceptadas por el motor gráfico.

E.2. Análisis de resultados

Todos los resultados aparecen descritos en la sección 5.6.

Centro de Mensajes

Presentación de la vista

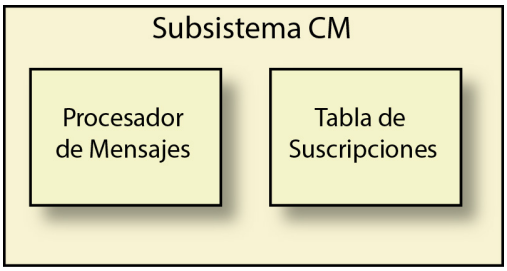


Figura 5.22: Descomposición del Centro de Mensajes

B. Catálogo de elementos

B.1. Elementos y sus propiedades

Elementos	Propiedades

Tabla de Suscripciones	En esta tabla se mantienen las suscripciones que los clientes del Centro de Mensajes han realizado. Estas suscripciones se organizan tanto por tipo de mensaje al que se suscribe como por el remitente de los mismos, de manera que puedan realizarse ambos tipos de suscripciones.
Procesamiento de mensajes	Este módulo implementará la lógica de funcionamiento del centro de mensajes. Por una parte, rellena la información que contiene la Tabla de Suscripciones, y por otra, consulta la tabla para saber a quién reenviar los mensajes recibidos en el Centro de Mensajes.

Tabla 5.5: Elementos y propiedades del Centro de Mensajes

B.2. Relaciones y sus propiedades

La relación que dirige la descomposición de esta vista es forma-parte-de. No hay excepciones ni añadidos a lo que define esta relación.

B.3. Interfaces

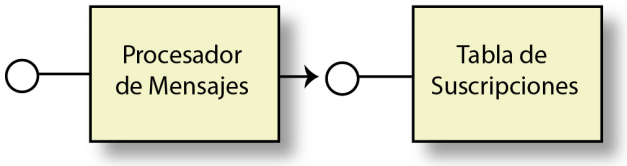


Figura 5.23: Interfaces del Centro de Mensajes

Elementos	Propiedades
Tabla de Suscripciones	Proporciona funciones para registrar nuevas suscripciones, para consultar las suscripciones existentes y para borrar suscripciones. Sólo es visible desde dentro del Centro de Mensajes.
Procesamiento de mensajes	Proporciona servicios para suscribirse a mensajes por tipo y por remitente. Espera que los clientes conectados al Centro de Mensajes proporcionen servicios para enviarles los mensajes a los que se han suscrito.

Tabla 5.6: Elementos y propiedades Centro de Mensajes

B.4. Comportamientos

No aplicable.

C. Diagrama de contexto

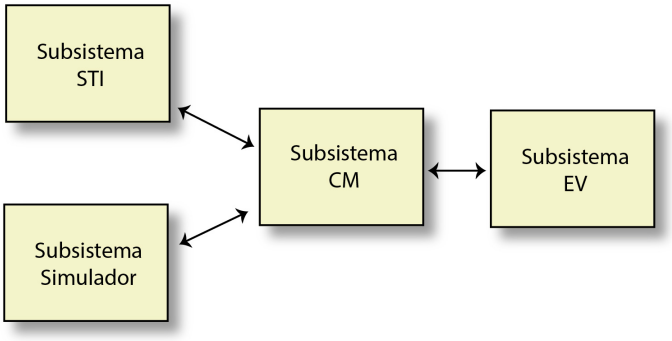


Figura 5.24: Diagrama de contexto del Centro de Mensajes

D. Guía de variabilidad

No aplicable.

E. Información sobre la arquitectura

E.1. Decisiones de diseño

El equipo de trabajo ha decidido hacer uso de la tabla de suscripciones por dos razones: la primera, evitar que las comunicaciones queden fijadas en el código, con lo que también se evita modificarlo para cambiar los detalles relativos a los clientes; la segunda, es que se posibilita la modificación del procesamiento de los mensajes, sin afectar a las suscripciones.

E.2. Análisis de resultados

Todos los resultados aparecen descritos en la sección 5.6.

E.3. Suposiciones

Se ha supuesto que los clientes del Centro de Mensajes no tienen por qué funcionar siempre correctamente. De esta manera, si un cliente se desconecta sin notificarlo y no borrar las suscripciones, el Centro de Mensajes lo realizará de manera automática cuando detecte que el cliente no está activo.

5.5.1.4. Vista de Descomposición Paquete 4: Simulador

Presentación de la vista

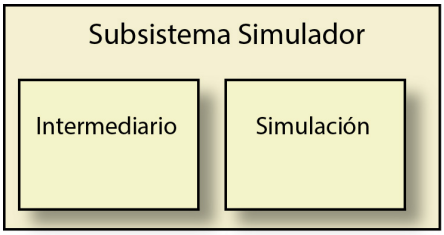


Figura 5.25: Descomposición del Simulador

B. Catálogo de elementos

B.1. Elementos y sus propiedades

Elementos	Propiedades
Simulación	Este módulo es el que realmente ejecuta la simulación. Su responsabilidad varía en función del dominio de enseñanza, pero básicamente consiste en reproducir el comportamiento de objetos o habitantes del entorno que no están bajo control directo de ningún usuario o de otro subsistema de la aplicación.
Intermediario	Puesto que la simulación puede no ser sencilla de modificar en caso de que esté ya implementada, este módulo es responsable de interactuar con la simulación y el resto de subsistemas que se tengan que comunicar con ella. De esta manera, oculta a la simulación los detalles de comunicación con el resto de la aplicación. En caso de ser necesario, también se ocupa de la traducción de mensajes e información entre la simulación y el resto de elementos de la aplicación.

Tabla 5.7: Elementos y propiedades del Simulador

B.2. Relaciones y sus propiedades

La relación que dirige la descomposición de esta vista es forma-parte-de. No hay excepciones ni añadidos a lo que define esta relación.

B.3. Interfaces

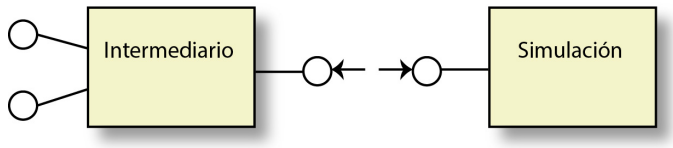


Figura 5.26: Interfaces del Simulador

Elementos	Propiedades
Simulación	Proporciona servicios para realizar acciones sobre elementos de la simulación, así como para consultar el estado o propiedades de los elementos de la misma. Sólo resulta visible desde dentro del subsistema.
Intermediario	Proporciona tres interfaces: una para comunicarse con el STI, otra para la comunicación con el Centro de Mensajes y una tercera para recibir notificaciones del simulador. La interfaz con el centro de mensajes ofrece la posibilidad de recibir mensajes de éste, mientras que la interfaz con el STI recibe directamente las acciones o consultas a realizar sobre la simulación.

Tabla 5.8: Elementos y propiedades del Simulador

C. Diagrama de contexto

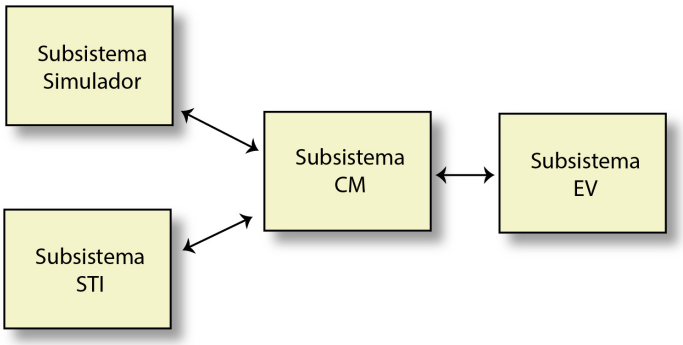


Figura 5.27: Diagrama de contexto del Simulador

D. Guía de variabilidad

El simulador es un componente que presenta una amplia gama de posibilidades a la hora de conectarlo al prototipo, dependiendo de si ya está o no implementado y de cómo lo está. Las situaciones que se han identificado son:

Simulador implementado e integrado en el EV: La comunicación con el STI se realizará a través del Centro de Mensajes, por lo que no tendrá conocimiento de la estructura del simulador ni del EV.

Simulador implementado e independiente: en este caso se presentan varias posibilidades, debidas a la naturaleza del simulador y del dominio en el que se realice el aprendizaje. Si el simulador realiza acciones que afectan tanto al funcionamiento del STI como a la visualización en el EV, lo más adecuado puede resultar que se comunique con el retos de subsistemas a través del Centro de Mensajes. Si el número de eventos que debe notificar el simulador es elevado, será mejor que la comunicación tenga lugar sin hacer uso del Centro de Mensajes para mantener así una buena velocidad de ejecución de la aplicación.

Se puede dar el caso de que el simulador realice muchos cambios que deban reflejarse en el EV, pero que no afecten mucho al STI. En esta Situación, es preferible que se comunique directamente con el EV, posiblemente también sin hacer uso del Centro de Mensajes por la misma razón que en el caso anterior. La comunicación que afecte al STI sí se podrá realizar a través del Centro de Mensajes.

Simulador no implementado: si el simulador es sencillo, se puede implementar de manera que se comunique directamente con el STI, de forma que no sea más complejo el mecanismo de comunicación que el propio simulador. En otro caso, nos podemos remitir a alguna de las situaciones anteriores.

También se puede comunicar el simulador directamente con el STI en caso de que sus eventos no afecten a ningún otro subsistema, sino tan sólo a las decisiones que deba tomar durante la tutoría.

E. Información sobre la arquitectura

E.1. Decisiones de diseño

Las simulaciones manejadas actualmente por el grupo de investigación que desarrolló el proyecto no realizan excesivas actualizaciones en el estado de los objetos que se muestran en el EV. De igual modo, en el EV no se llevan a cabo demasiadas acciones que afecten al estado del simulador. Por esta razón, parece adecuado mantener la integridad conceptual del diseño arquitectónico y realizar la comunicación a través del Centro de Mensajes.

En cuanto al simulador en sí, su funcionamiento debe mantenerse independiente del hecho de que la comunicación se realice o no a través del centro de mensajes, y también de con quién se

realice. Por esta razón se ha decidido incluir un intermediario entre la simulación y el resto de elementos. Este intermediario será quien se ocupe de los detalles de comunicación, de manera que si hay que modificar algún aspecto relativo a ella baste con modificar el intermediario, pero no la simulación.

E.2. Análisis de resultados

Todos los resultados aparecen descritos en la sección 5.6.

E.3. Suposiciones

Para la solución adoptada se ha supuesto que el simulador no está implementado, es de complejidad no muy elevada y no realiza demasiadas actualizaciones en el estado del EV.

F. Otra información

Uno de los motivos para decidir cómo comunicar el simulador con el resto de elementos de la aplicación radica en el hecho de que se propuso utilizar JADE[3] como plataforma para implementar los agentes. JADE implementa el mecanismo de envío de mensajes entre unos agentes y otros de forma tal que todos los agentes se ven afectados por todos los mensajes que se envían (no los reciben, pero se enteran de que se han enviado). Si el simulador envía muchos mensajes, todos los agentes se verían afectados, por lo que el prototipo sufriría una disminución en la velocidad de ejecución. Por eso resulta preferible que el simulador se comunique directamente con el agente de simulación. Por esta misma razón, es preferible que no todos los mensajes pasen a través de dicho agente, ya que si no son mensajes relevantes pará el (ni para otros agentes) se ralentiza el prototipo sin obtener ningún beneficio. Esta es la razón para que la simulación se comunique con el EV directamente a través del Centro de Mensajes en lugar de aprovechar que ya se comunica con el STI.

---

3 **Java Agent DEvelopment Framework**, o **JADE**, es una plataforma software para el desarrollo de agentes, implementada en Java. La plataforma JADE soporta la coordinación de múltiples agentes FIPA y proporciona una implementación estándar del lenguaje de comunicación FIPA-ACL, que facilita la comunicación entre agentes y permite la detección de servicios que se proporcionan en el sistema.

5.5.1.5. Vista de Descomposición

Sistema de Tutoría Inteligente

Presentación de la vista

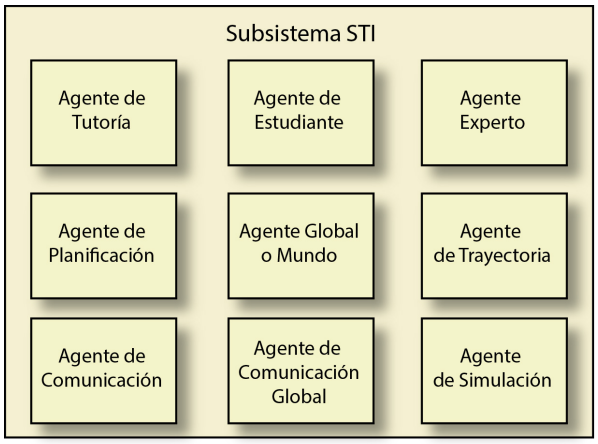


Figura 5.28: Descomposición del STI

Catálogo de elementos

B.1. Elementos y sus propiedades

Elementos	Propiedades
Agente de Comunicación	Se encarga de la comunicación específica del STI con un estudiante concreto. De esta manera, recibe, a través del Centro de Mensajes, todos los mensajes provenientes de un único EV, y envía todos los mensajes del STI que son específicos para ese mismo EV.
Agente de Comunicación Global	Este agente es responsable de una parte de la comunicación del STI con el resto de subsistemas de la aplicación. En concreto, se encarga de las comunicaciones relativas al inicio de sesión de los alumnos, así como de las comunicaciones que no son específicas de estos, como las acciones que el agente de simulación tenga que enviar al EV. También se encarga de la creación de los agentes de comunicación cuando se conectan nuevos estudiantes.
Agente de Simulación	Es el encargado de manejar en el STI el conocimiento relativo a la simulación. De esta manera, si un alumno realiza una acción que contiene alguna precondición relativa al estado de la simulación, el Agente de Simulación es quien maneja la información para saber si la condición se cumple o no. De igual manera, si la acción realizada sobre la simulación tiene consecuencias visibles en el EV, el Agente de Simulación será el encargado de emitir la orden de ejecución de la mencionada acción. Este agente realiza labores de comunicación entre el STI y el Simulador.

Agente Experto	Es el responsable de controlar la ejecución de las acciones realizadas por los estudiantes. Cada acción realizada tiene unas precondiciones, que indican si se puede ejecutar o no independientemente de su corrección, y unas postcondiciones, que indican los efectos de la ejecución de la acción. Una vez ejecutada la acción, es el encargado de enviar las postcondiciones de las acciones para que los agentes interesados actualicen su información al respecto y para que el EV muestre los resultados a los alumnos.
Agente Global o Mundo	Este agente se encarga de mantener actualizada la información del estado del entorno de aprendizaje. De esta manera, todos los demás agentes pueden consultar la información actualizada relativa al estado del mundo sin necesidad de tener que pedirla a los EVs. También es el encargado de responder preguntas acerca de la relación entre distintos objetos, como por ejemplo si están cerca o si uno está encima de otro.
Agente de Planificación	Es el encargado de planificar la ejecución de los ejercicios propuestos a los estudiantes para su aprendizaje. Es capaz de realizar un plan, actualizar el estado del plan a medida que los estudiantes ejecutan acciones y replanificar la ejecución de un ejercicio en caso de que las acciones de los estudiantes se desvíen del plan vigente y no permitan llegar a una solución siguiendo dicho plan.
Agente de Trayectoria	Es el agente responsable de registrar las coordenadas de los movimientos de los estudiantes por el EV y, en caso de requerirlo la actividad objeto de aprendizaje, de comprobar la idoneidad de la trayectoria seguida por los estudiantes en su desplazamiento por el EV y de informar si un estudiante ha llegado a un lugar relevante para la actividad.
Agente de Estudiante	Este agente es el encargado de realizar el modelo del estudiante en función de las acciones que va llevando a cabo durante el aprendizaje. De esta manera, registra las acciones que el estudiante ejecuta dentro del EV junto con su evaluación, las trayectorias seguidas y su evaluación, las preguntas realizadas y cualquier otra información relevante para el modelo utilizado. Asimismo, proporciona la información contenida en este modelo a los agentes que la soliciten.
Agente de Tutoría	El Agente de Tutoría constituye la pieza angular del STI, ya que, se puede decir que todos los demás agentes trabajan para que él pueda realizar su labor. Esta consiste en ayudar al estudiante en el proceso de aprendizaje. Dentro de sus responsabilidades se encuentra plantear el procedimiento en el que se debe enseñar al estudiante y, una vez iniciado el proceso, monitoraer las acciones que realice. El Agente de Tutoría manejará una o varias estrategias de tutoría, las cuales serán utilizadas para decidir qué hacer ante las acciones de los estudiantes. También será el responsable de contestar a las preguntas del estudiante y de ofrecerle pistas cuando lo considere oportuno. Otra de las tareas que se encuentran a su cargo es evaluar las acciones del estudiante, lo que contribuirá a la realización del modelo del estudiante.

Tabla 5.9: Elementos y propiedades del Sistema de Tutoría Inteligente

B.2. Relaciones y sus propiedades

Elementos	Propiedades
Agente de Comunicación	El agente de comunicación es uno de los puntos de unión entre el STI y el resto de los subsistemas de la aplicación. De esta manera, ofrece una interfaz para recibir mensajes provenientes del Centro de Mensajes. Este agente no asume la existencia de ningún otro agente, por lo que no envía la información recibida a ningún agente que no se la haya solicitado previamente. Para ello, proporciona a los demás agentes del STI otra interfaz con servicios para suscribirse a la recepción de mensajes con la información que les interese proveniente del exterior del STI. También proporciona servicios para el envío de información a otros subsistemas de la aplicación.
Agente de Comunicación Global	Al igual que el anterior, es un punto de comunicación entre el STI y los demás subsistemas de la aplicación, por lo que proporciona una interfaz para recibir mensajes del Centro de Mensajes. La información recibida se la envía únicamente a los agentes que se suscriban a ella. Además, proporciona servicios para enviar mensajes a otros subsistemas de la aplicación.
Agente de Simulación	Este agente es otra de las puertas de comunicación del STI con el exterior, en este caso con la simulación. Proporciona servicios para que los agentes del STI puedan consultar aspectos relativos al estado de la simulación y para que ejecuten acciones dentro de la misma. También proporciona una interfaz para que la simulación pueda enviar cambios relevantes para el STI. Asume la existencia de un simulador, pero no la de otros agentes.
Agente Experto	Proporciona una interfaz para que se le solicite la validación de las precondiciones de las acciones realizadas por los estudiantes. Necesita que existan agentes que le proporcionen la información necesaria para efectuar la validación de las precondiciones, así como un agente que actué como intermediario para enviar los resultados de las acciones a los demás subsistemas de la aplicación.
Agente Global o Mundo	Proporciona servicios para consultar y actualizar la información del entorno de aprendizaje. También proporciona servicios para responder preguntas acerca de los objetos de los que mantiene información.
Agente de Planificación	Proporciona servicios para solicitar la realización de un plan, así como de replanificaciones. También permite la aplicación de acciones sobre un plan y la solicitud del plan actual y de la siguiente acción a realizar.
Agente de Trayectoria	Proporciona una interfaz para consultar si un estudiante está siguiendo la trayectoria adecuada o si ha llegado a un lugar relevante. Necesita que algún agente le proporcione las coordenadas con las que debe trabajar.
Agente de Estudiante	Proporciona servicios para consultar y actualizar la información contenida en el modelo del estudiante. Necesita que otros agentes le proporcionen la información con la que elaborar el modelo.

Agente de Tutoría	Proporciona información acerca de las acciones de los estudiantes, así como las correspondientes decisiones de tutoría. Necesita que se le proporcionen las acciones de los estudiantes y las del plan del ejercicio a realizar, así como servicios para validar las acciones de los estudiantes y para consultar datos relativos a los estudiantes.

Tabla 5.10: Relaciones y propiedades del Sistema de Tutoría Inteligente

La relación que dirige la descomposición de esta vista es forma-parte-de. No hay excepciones ni añadidos a lo que define esta relación.

B.3. Interfaces

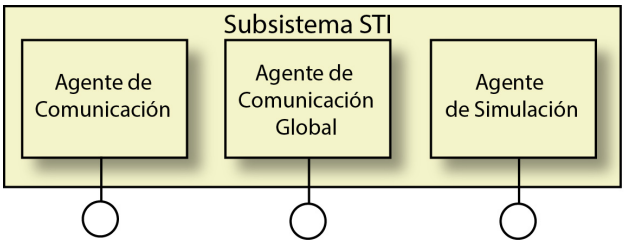


Figura 5.29: Interfaces del STI

En la imagen se muestran únicamente las interfaces que ofrecen los agentes hacia el exterior del STI. Las interfaces entre los agentes se muestran en mayor detalle en las vistas de edición-suscripción 5.5.3.

B.4. Comportamientos

No aplicable.

C. Diagrama de contexto

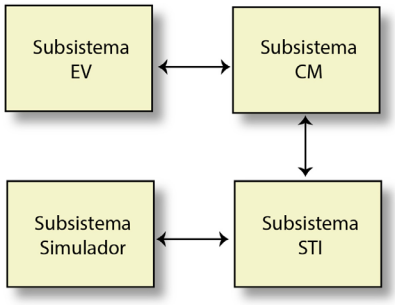


Figura 5.30: Diagrama de contexto del STI

D. Guía de variabilidad

Como propiedades de dinamismo de este subsistema, cabe destacar que existirán tantos Agentes de Comunicación y tantos Agentes de Estudiante como estudiantes se encuentren conectados al sistema. Ambos tipos de agentes se irán creando a medida que se vayan conectando nuevos estudiantes.

Para información relativa a variabilidad, se puede consultar la sección específica de cada uno de los agentes y entender cada caso en particular.

E. Información sobre la arquitectura

E.1. Decisiones de diseño

Se ha decidido crear Agentes de Comunicación y Agentes de Estudiante específicos para cada estudiante por dos motivos:

El primero es que así es posible tratar las acciones de los estudiantes en paralelo, de manera que no sea necesario que las acciones de un estudiante tengan que esperar a ser procesadas si no es imprescindible. Por otro lado, cabe la posibilidad de que, debido a un incremento del número de agentes, sea necesario distribuirlos en varios procesadores. Con agentes específicos para cada estudiante resulta posible distribuir los agentes en función de los estudiantes a los que atienden.

Como ya se ha mencionado al describir el Agente de Tutoría, éste es la pieza central del STI, razón por la cual puede existir la tendencia a asignarle mucha responsabilidad. Como se verá en las próximas secciones, las distintas responsabilidades de los agentes se han separado en distintos comportamientos que deberán mantenerse independientes para, llegado el caso, poder modificar la asignación de responsabilidades e, incluso, introducir agentes nuevos que se hagan cargo de ellas. Este ha sido el caso de agente experto, que ha surgido en parte de la necesidad de reducir la carga de responsabilidades del agente de tutoría, lo que también permite que éste atienda la acción de un alumno mientras se verifica que puede ejecutarse la acción realizada por otro.

Para realizar la descomposición se han tenido en cuenta varios factores. El más influyente ha sido la necesidad de que ciertos componentes deban ser sustituibles (o eliminables), lo que ocasiona que tengan que contemplarse como componentes separados. Este es el caso de los agentes de planificación, estudiante, trayectoria y tutoría.

Para identificar el resto de módulos se ha aplicado el principio de ocultación de información de Parnas (1972) para obtener un diseño más modificable. Existen distintas interpretaciones acerca

de la aplicación de este principio. La utilización de agentes proporciona enormes ventajas a este respecto, ya que la principal forma de comunicación entre ellos, el paso de mensajes, desacopla a los emisores y a los receptores de los mismos, posibilitando el intercambio de unos agentes por otros, sin que el agente que solicita un servicio perciba el cambio. También ayuda a ello el hecho de que un agente no tiene por qué saber qué agente va a prestarle un servicio hasta que el prototipo está en ejecución, a través de un servicio de búsqueda en unas páginas amarillas. Todo esto se debe ver complementado, no obstante, por una manera adecuada de solicitar los servicios, para lo que las interfaces de comunicación entre los agentes deben estar diseñadas adecuadamente.

Se puede encontrar más información en las vistas de descomposición de cada agente, en la vista editor-suscriptor y en las vistas peer-to-peer.

E.2. Análisis de resultados

Todos los resultados aparecen descritos en la sección 5.6.

E.3. Suposiciones

Se ha pensado que la implementación del STI se va a realizar utilizando la plataforma JADE, la cual proporciona una serie de servicios y mecanismos que no se muestran en los diagramas, como pueden ser los servicios de páginas blancas y páginas amarillas. De igual manera, también proporciona la ejecución en paralelo de los agentes y mecanismos de comunicación y sincronización, razón por la cual no es necesario incorporarlos entre los módulos en que se descompone el STI.

5.5.1.6. Vista de Descomposición

Agente de Comunicación Global

Presentación de la vista

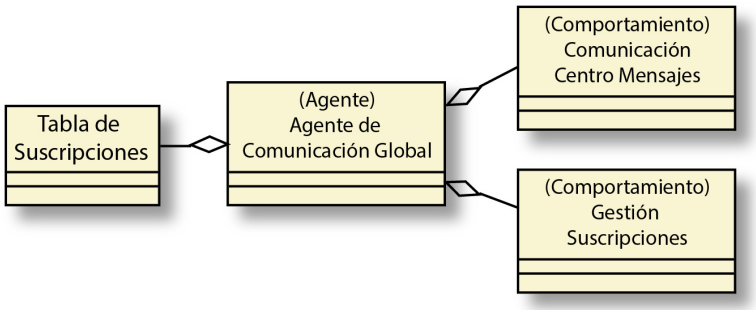


Figura 5.31: Descomposición del Agente de Comunicación Global

Catálogo de elementos

B.1 Elementos y sus propiedades

Elementos	Propiedades
Agente de Comunicación Global	Este elemento, aunque lleva el mismo nombre que el módulo que lo contiene, constituye en sí lo que es el agente, el cual tiene asociados una serie de comportamientos que llevan a cabo la mayor parte de las responsabilidades que le corresponden. Su principal responsabiida consiste en la creación y puesta en marcha inicial de los comportamientos, así como ejercer de representante de los demás elementos ante el resto de los agentes, enviando y recibiendo mensajes.
Comunicación con el Centro de Mensajes	Este elemento es un comportamiento del Agente de Comunicación Global que está encargado de toda la comunicación con el Centro de Mensajes. Cuando el agente recibe un mensaje de otro agente lo envía al Centro de Mensajes, mientras que si el mensaje proviene del Centro de Mensajes, entonces se lo envía a los agentes que estén suscritos a ese tipo de mensaje.
Gestión de Suscripciones	Este comportamiento se encarga de recibir las peticiones de suscripción de otros agentes a los mensajes provenientes del Centro de Mensajes (y, por tanto, de otros subsistemas de la aplicación) y de apuntarlas en la Tabla de Suscripciones.
Tabla de Suscripciones	Este elemento recoge las suscripciones de otros agentes a mensajes provenientes del exterior del STI.

Tabla 5.11: Elementos y propiedades del Agente de Comunicación Global

B.2. Relaciones y sus propiedades

La relación que dirige la descomposición de esta vista es forma-parte-de. No hay excepciones ni añadidos a lo que define esta relación.

B.3. Interfaces

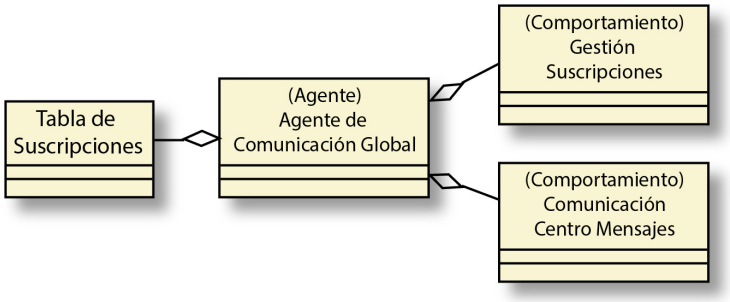


Figura 5.32: Interfaces del Agente de Comunicación Global

Elementos	Propiedades
Agente de Comunicación Global	Proporciona interfaces para enviar y recibir mensajes de otros agentes. No asume la existencia de ningún otro agente.
Comunicación con el Centro de Mensajes	Proporciona servicios para mandar y recibir mensajes del Centro de Mensajes. Asume la existencia del Centro de Mensajes.
Gestión de Suscripciones	Proporciona servicios para suscribirse a mensajes por remitente y por tipo de contenido. También proporciona servicios para borrar suscripciones. Necesita a la existencia de un elemento que almacene las suscripciones.
Tabla de Suscripciones	Proporciona servicios para añadir, borrar y modificar suscripciones.

Tabla 5.12: Relaciones y propiedades del Agente de Comunicación Global

C. Diagrama de contexto

Figura 5.33: Diagrama de contexto del Agente de Comunicación Global

D. Guía de variabilidad

No aplicable.

E. Información sobre la arquitectura

E.1. Decisiones de diseño

Puesto que este agente no necesita entender el contenido de los mensajes, se ha decidido dotarlo únicamente de dos comportamientos. Cada uno de ellos se encarga de la comunicación con un tipo de elemento diferente: agentes y Centro de Mensajes.

E.2. Análisis de resultados

Todos los resultados aparecen descritos en la sección 5.6.

5.5.1.7. Vista de Descomposición

Agente de Comunicación

Presentación de la vista

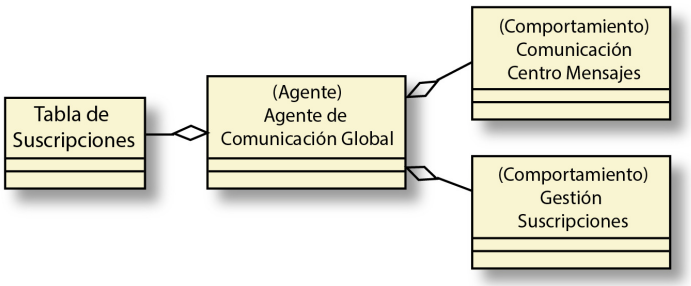


Figura 5.34: Descomposición del Agente de Comunicación.

Catálogo de elementos

B.1. Elementos y sus propiedades

Elementos	Propiedades
Agente de Comunicación	Este elemento, aunque lleva el mismo nombre que el módulo que lo contiene, constituye en sí lo que es el agente, el cual tiene asociados una serie de comportamientos que llevan a cabo la mayor parte de las responsabilidades que le corresponden a este. La responsabilidad principal de este elemento consiste en la creación y puesta en marcha inicial de los comportamientos, así como ejercer de representante de los demás elementos ante el resto de los agentes, enviando y recibiendo mensajes. Además, debe registrarse en el Centro de Mensajes para recibir los mensajes provenientes del estudiante que causó su creación.
Comunicación con el Centro de Mensajes	Este elemento es un comportamiento del Agente de Comunicación que está encargado de toda la comunicación con el Centro de Mensajes. Cuando el agente recibe un mensaje de otro agente lo envía al Centro de Mensajes, mientras que si el mensaje proviene del Centro de Mensajes, entonces se lo envía a los agentes que estén suscritos a ese tipo de mensaje.

Gestión de Suscripciones	Este comportamiento se encarga de recibir las peticiones de suscripción de otros agentes a los mensajes provenientes del Centro de Mensajes (y, por tanto, de un estudiante concreto) y de apuntarlas en la Tabla de Suscripciones.
Tabla de Suscripciones	Este elemento recoge las suscripciones de otros agentes a mensajes provenientes del estudiante al que representa.

Tabla 5.13: Elementos y propiedades del Agente de Comunicación

B.2. Relaciones y sus propiedades

La relación que dirige la descomposición de esta vista es forma-parte-de. No hay excepciones ni añadidos a lo que define esta relación.

B.3. Interfaces

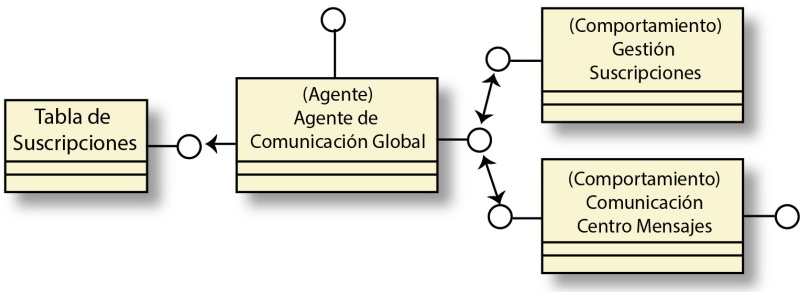


Figura 5.35: Interfaces del Agente de Comunicación

Elementos	Propiedades
Agente de Comunicación	Proporciona interfaces para enviar y recibir mensajes de otros agentes. No asume la existencia de ningún otro agente.
Comunicación con el Centro de Mensajes	Proporciona servicios para mandar y recibir mensajes del Centro de Mensajes. Asume la existencia del Centro de Mensajes.
Gestión de Suscripciones	Proporciona servicios para suscribirse a mensajes por remitente y por tipo de contenido. También proporciona servicios para borrar suscripciones. Necesita a la existencia de un elemento que almacene las suscripciones.
Tabla de Suscripciones	Proporciona servicios para añadir, borrar y modificar suscripciones.

Tabla 5.14: Relaciones y propiedades del Agente de Comunicación.

C. Diagrama de contexto

Figura 5.36: Diagrama de contexto del Agente de Comunicación.

D. Guía de variabilidad

No aplicable.

E. Información sobre la arquitectura

E.1. Decisiones de diseño

Puesto que este agente no necesita entender el contenido de los mensajes, se ha decidido dotarlo únicamente de dos comportamientos, cada uno de los cuales se encarga de la comunicación con un tipo de elemento diferente: agentes y el Centro de Mensajes.

E.2. Análisis de resultados

Todos los resultados aparecen descritos en la sección 5.6.

E.3. Suposiciones

No aplicable.

F. Otra información

No aplicable.

5.5.1.8. Vista de Descomposición

Agente de Simulación

Presentación de la vista

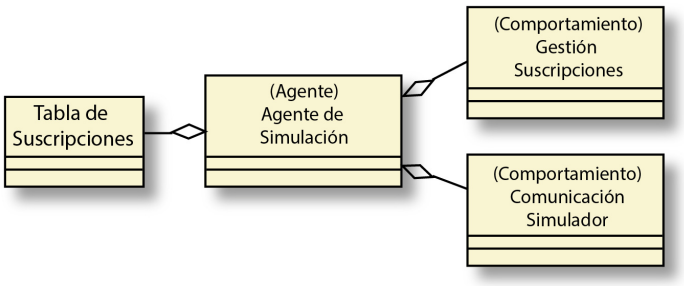


Figura 5.37: Descomposición del Agente de Simulación

B.Catálogo de elementos

B.1. Elementos y sus propiedades

Elementos	Propiedades
Agente de Simulación	Este elemento, aunque lleva el mismo nombre que el módulo que lo contiene, constituye en sí lo que es el agente, el cual tiene asociados una serie de comportamientos que llevan a cabo la mayor parte de las responsabilidades que le corresponden á este. La responsabilidad principal de este elemento consiste en la creación y puesta en marcha inicial de los comportamientos, así como ejercer de representante de los demás elementos ante el resto de los agentes, enviando y recibiendo mensajes.
Comunicación con el Simulador	La responsabilidad de este comportamiento consiste en llevar a cabo la comunicación con el Simulador. De esta manera, cuando el agente reciba un mensaje para consultar el estado del Simulador o para enviarle una acción, este comportamiento será el encargado de ponerse en contacto con el simulador y devolver al agente correspondiente la respuesta obtenida, en caso de ser necesario.
Gestión de Suscripciones	Este comportamiento se encarga de recibir las peticiones de suscripción de otros agentes a los mensajes provenientes del Simulador y de apuntarlas en la Tabla de Suscripciones.
Tabla de Suscripciones	Este elemento recoge las suscripciones de otros agentes a mensajes provenientes del simulador.

Tabla 5.15: Elementos y propiedades del Agente de Simulación

B.2. Relaciones y sus propiedades

La relación que dirige la descomposición de esta vista es forma-parte-de. No hay excepciones ni añadidos a lo que define esta relación.

Elementos	Propiedades
Agente de Simulación	Proporciona interfaces para enviar y recibir mensajes de otros agentes. No asume la existencia de ningún otro agente.
Comunicación con el Simulador	Proporciona servicios para mandar y recibir mensajes del Simulador. Asume la existencia del Simulador.
Gestión de Suscripciones	Proporciona servicios para suscribirse a mensajes por remitente y por tipo de contenido. También proporciona servicios para borrar suscripciones. NeceSTIala existencia de un elemento que almacene las suscripciones.
Tabla de Suscripciones	Proporciona servicios para añadir, borrar y modificar suscripciones.

Tabla 5.16: Relaciones y propiedades del Agente de Simulación

B.3. Interfaces

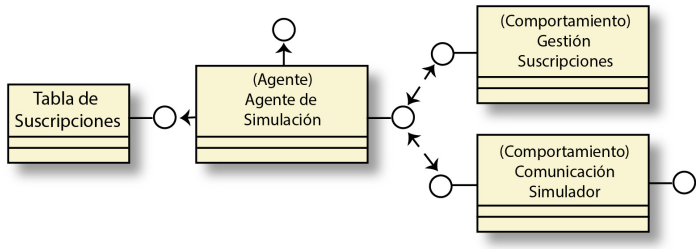


Figura 5.38: Interfaces del Agente de Simulación

C. Diagrama de contexto

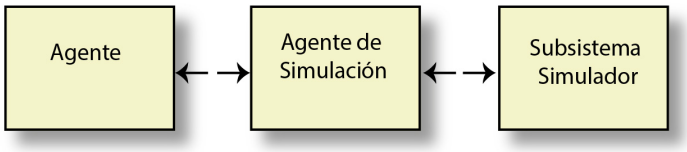


Figura 5.39: Diagrama de contexto del Agente de Simulación

D. Guía de variabilidad

El comportamiento de comunicación con el simulador es dependiente del simulador que se esté utilizando. De esta forma, cuando se desee cambiar el simulador, habrá que sustituir también

este comportamiento por otro equivalente que posibilite la comunicación con el nuevo simulador.

Como se muestra en la vista correspondiente al Simulador(5.5.1.4), éste posee un módulo que realiza las funciones de intermediario entre la simulación y el STI. Llegado el momento de sustituir la simulación por otra, es posible que sólo sea necesario realizar cambios en el intermediario, y no en el comportamiento de comunicación con el simulador del agente de simulación. Sin embargo, como se ha mencionado, este aspecto es dependiente del simulador con el que se trabaje. Al no haber realizado un análisis exhaustivo de todos los tipos de simuladores que pueden utilizarse, no se puede afirmar de manera categórica que no vaya a ser necesario realizar cambios en el comportamiento del agente.

E. Información sobre la arquitectura

E.1. Decisiones de diseño

El Agente de simulación, en su conjunto, es un agente que tiene como responsabilidad comunicar el simulador con el resto de los agentes, llevando a cabo una labor que bien puede identificarse con el patrón Wrapper descrito en Hayden y otros (1999). Es por este motivo que las labores del agente en sí quedan reducidas a la traducción de información y a la comunicación con el Simulador y con los demás agentes. De cara al STI, el Agente de Simulación realiza todas las tareas que lleva a cabo el Simulador, aunque, como se ha dicho, en realidad sólo constituye una forma de comunicar el simulador con el resto de los agentes.

E.2. Análisis de resultados

Todos los resultados aparecen descritos en la sección 5.6.

E.3. Suposiciones

La plataforma de agentes que se va a utilizar para implementar la arquitectura como ya se ha mencionado es JADE, que funciona sobre el lenguaje de programación Java. Se ha supuesto que será posible comunicar la plataforma JADE con el lenguaje de programación utilizado para implementar el simulador, ya sea directamente o bien, por ejemplo, mediante la utilización de sockets.

5.5.1.9. Vista de Descomposición

Agente Experto

Presentación de la vista

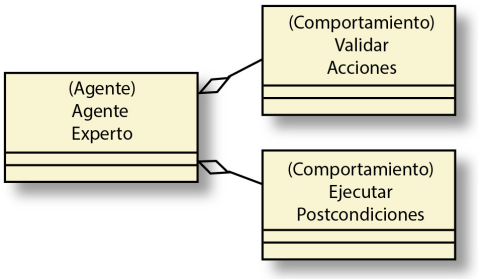


Figura 5.40: Descomposición del Agente Experto.

B. Catálogo de elementos

B.1. Elementos y sus propiedades

Elementos	Propiedades
Agente Experto	Este elemento, aunque lleva el mismo nombre que el módulo que lo contiene, constituye en sí lo que es el agente, el cual tiene asociados una serie de comportamientos que llevan a cabo la mayor parte de las responsabilidades que le corresponden a este. La responsabilidad principal de este elemento consiste en la creación y puesta en marcha inicial de los comportamientos, así como ejercer de representante de los demás elementos ante el resto de los agentes, enviando y recibiendo mensajes.
Validar Acciones	La responsabilidad de este comportamiento consiste en realizar la validación de las precondiciones de una acción. En caso necesario, deberá decidir si necesita información que le pueda proporcionar otro agente para realizar la validación.
Ejecutar Postcondiciones	La responsabilidad de este comportamiento es la de decidir qué postcondiciones de una acción son directamente relevantes para un estudiante o para otros agentes, y enviarle a cada uno las postcondiciones que le puedan interesar.

Tabla 5.17: Elementos y propiedades del Agente Experto

B.2. Relaciones y sus propiedades

La relación que dirige la descomposición de esta vista es forma-parte-de. No hay excepciones ni añadidos a lo que define esta relación.

B.3. Interfaces

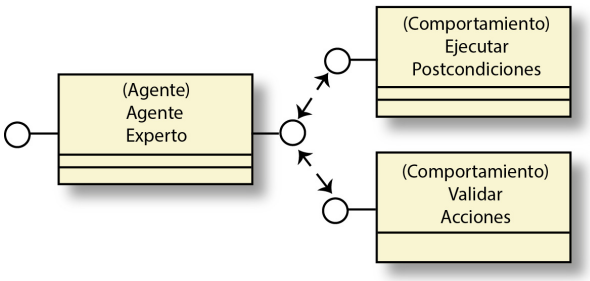


Figura 5.41: Interfaces del Agente Experto

Elementos	Propiedades
Agente Experto	Proporciona interfaces para enviar y recibir mensajes de otros agentes. No asume la existencia de ningún otro agente.
Validar Acciones	Proporciona servicios para que le soliciten la validación de acciones enteras o de precondiciones. No asume la existencia de otros agentes, aunque puede necesitarlos para realizar la validación de alguna precondición.
Ejecutar Postcondiciones	Proporciona servicios para recibir acciones o postcondiciones que tratar. No asume la existencia de ningún agente.

Tabla 5.18: Relaciones y propiedades del Agente Experto

C. Diagrama de contexto

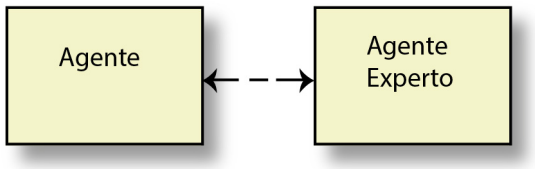


Figura 5.42: Diagrama de contexto del Agente Experto

E. Información sobre la arquitectura

E.1. Decisiones de diseño

Puesto que las dos tareas principales de este agente consisten en la validación de las precondiciones de una acción y la ejecución de las postcondiciones de la misma, se ha tomado la decisión de realizar cada una de estas acciones en un comportamiento distinto.

La validación de las precondiciones depende de la información que manejan el Agente de Simulación y el Agente Mundo, por lo que, aunque por el momento se ha descartado, se contempla la posibilidad de separar la validación en dos comportamientos separados. De esta manera, entre otros aspectos, se posibilita la validación de ambos tipos de precondiciones en paralelo. No se ha hecho uso de esta opción porque, de momento, el número de precondiciones de una acción se prevé que sea pequeño, y la validación de cada precondición es, en principio, una operación rápida.

E.2. Análisis de resultados

Todos los resultados aparecen descritos en la sección 5.6.

5.5.1.10. Vista de Descomposición

Agente Global o Mundo

Presentación de la vista

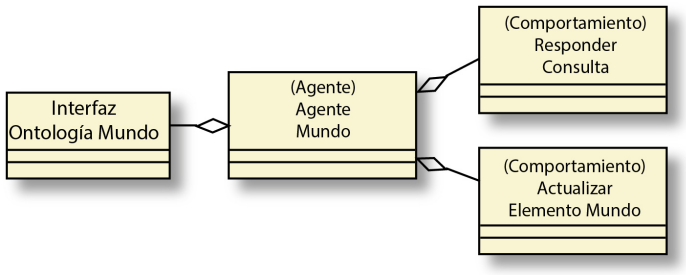


Figura 5.43: Descomposición del Agente Mundo

B. Catálogo de elementos

B.1. Elementos y sus propiedades

Elementos	Propiedades
Agente Mundo	Este elemento, aunque lleva el mismo nombre que el módulo que lo contiene, constituye en sí lo que es el agente, el cual tiene asociados una serie de comportamientos que llevan a cabo la mayor parte de las responsabilidades que le corresponden a éste. La responsabilidad principal de este elemento consiste en la creación y puesta en marcha inicial de los comportamientos, así como ejercer de representante de los demás elementos ante el resto de los agentes, enviando y recibiendo mensajes.

Actualizar Elemento del Mundo	La responsabilidad de este comportamiento consiste en mantener actualizada la información que maneja el STI acerca de los elementos presentes en el mundo virtual. Para ello, se encarga de recibir la información proporcionada por otros agentes y la estructura y almacena en la Ontología del Mundo.
Responder una Consulta	La responsabilidad de este comportamiento es la de responder a las preguntas realizadas por otros agentes acerca de los elementos presentes en el mundo virtual. Estas preguntas pueden ser simples, como la consulta de la posición de un objeto, o más elaboradas, como la relación existente entre dos elementos del mundo. Las respuestas que sea capaz de ofrecer dependerán de la complejidad de la ontología que se maneja, así como del motor de inferencia que opere sobre ella.
Interfaz con la Ontología del Mundo	Este elemento se encarga de facilitar el intercambio de información entre el Agente Mundo y la Ontología del Mundo, así como de realizar las traducciones correspondientes entre uno y otro.

Tabla 5.19: Elementos y propiedades del Agente Mundo

B.2. Relaciones y sus propiedades

La relación que dirige la descomposición de esta vista es forma-parte-de. No hay excepciones ni añadidos a lo que define esta relación.

Elementos	**Propiedades**
Agente Mundo	Proporciona interfaces para enviar y recibir mensajes de otros agentes. No asume la existencia de ningún otro agente.
Actualizar Elemento del Mundo	Proporciona servicios para recibir información de actualización de otros agentes. Asume la existencia de algún elemento que almacene la información recibida.
Responder una Consulta	Proporciona servicios para responder a las preguntas realizadas por otros agentes. Asume la existencia de un repositorio que almacena la información necesaria para responder las preguntas.
Interfaz con la Ontología del Mundo	Proporciona servicios para acceder a la Ontología del Mundo, tanto para realizar consultas sobre ella como para actualizar la información que contiene. Asume la existencia de la ontología.

Tabla 5.20: Relaciones y propiedades del Agente Mundo

B.3. Interfaces

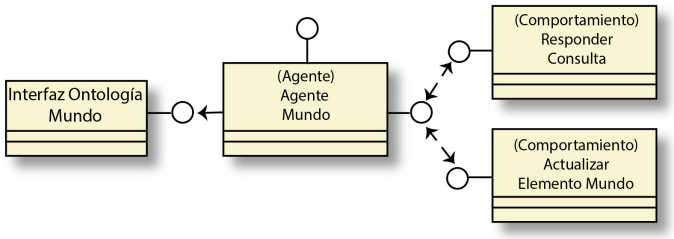


Figura 5.44: Interfaces del Agente Mundo

B.4. Comportamientos

No aplicable.

C. Diagrama de contexto

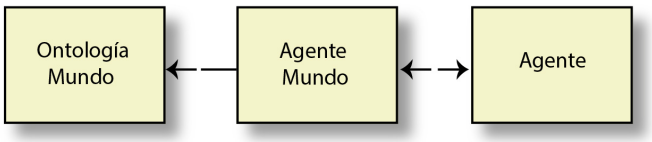


Figura 5.45: Diagrama de contexto del Agente Mundo

E. Información sobre la arquitectura

E.1. Decisiones de diseño

El equipo sugiriró utilizar una ontología para almacenar la información del Mundo en lugar de una base de datos fundamentalmente debido a que con ella resulta más sencillo realizar inferencias y razonamientos a partir de la información que contiene, mediante la utilización de un motor de inferencia. Además, también facilita la creación de nuevas relaciones entre los elementos, lo cual da mayor flexibilidad a la hora de ofrecer a los alumnos respuestas más completas a sus preguntas sobre objetos del entorno.

E.2. Análisis de resultados

Todos los resultados aparecen descritos en la sección 5.6.

E.3. Suposiciones

No aplicable.

F. Otra información

El equipo propuso que para el diseño de la ontología se utilice Protegé[4] y las inferencias se hagan utilizando Jena[5]. El diseño de la ontología y su utilización estarán restringidos a lo que permitan hacer ambas herramientas.

5.5.1.11. Vista de Descomposición

Agente de Planificación

Presentación de la vista

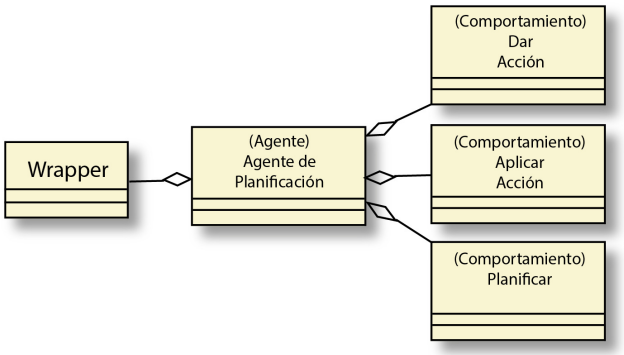


Figura 5.46: Descomposición del Agente de Planificación

4 **Protégé** es un editor libre de código abierto y un sistema de adquisición de conocimiento. es una herramienta integrada de software para desarrollar sistemas basados en el conocimiento.

5 Es un framework de código abierto para desarrollar elementos para la web semántica basado en Java.

B. Catálogo de elementos

B.1. Elementos y sus propiedades

Elementos	Propiedades
Agente de Planificación	Proporciona interfaces para enviar y recibir mensajes de otros agentes. No asume la existencia de ningún otro agente.
Dar Acción	Proporciona servicios para que otros agentes soliciten la siguiente acción del plan. Asume la existencia de un elemento que elabora el mencionado plan.
Aplicar Acción	Proporciona servicios para que otros agentes soliciten la aplicación de una acción sobre el plan en curso. Asume la existencia de un elemento que elabora el mencionado plan.
Planificar	Proporciona servicios para que otros agentes soliciten la elaboración de un nuevo plan. Supone la existencia de un elemento que elabora el mencionado plan.
Wrapper	Proporciona servicios para que el Agente de Planificación se comunique con el planificador. Asume la existencia de su correspondiente planificador.

Tabla 5.21: Elementos y propiedades del Agente de Planificación

B.2. Relaciones y sus propiedades

La relación que dirige la descomposición de esta vista es forma-parte-de. No hay excepciones ni añadidos a lo que define esta relación.

Elementos	Propiedades
Agente de Planificación	Este elemento, aunque lleva el mismo nombre que el módulo que lo contiene, constituye en sí lo que es el agente, el cual tiene asociados una serie de comportamientos que llevan a cabo la mayor parte de las responsabilidades que le corresponden a éste. La responsabilidad principal de este elemento consiste en la creación y puesta en marcha inicial de los comportamientos, así como ejercer de representante de los demás elementos ante el resto de los agentes, enviando y recibiendo mensajes.
Dar Acción	La responsabilidad de este comportamiento consiste en entregar la siguiente acción que se debe ejecutar de acuerdo con el plan actualmente vigente.
Aplicar Acción	Este comportamiento se encarga de aplicar, sobre el estado actual del planificador, la acción que se le envíe, comprobando si es posible realizar la mencionada acción de acuerdo con sus precondiciones y el estado del plan.
Planificar	Este comportamiento se encarga de realizar un nuevo plan cuando se lo solicite algún otro agente, partiendo de una Situación inicial para un problema dado o de la Situación actual del plan en curso.
Wrapper	Este elemento se encarga de comunicar al Agente de Planificación con el planificador, realizando las labores de traducción necesarias entre ambos.

Tabla 5.22: Relaciones y propiedades del Agente de Planificación

B.3. Interfaces

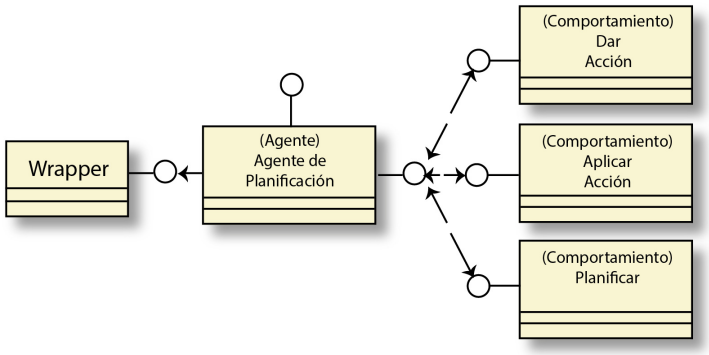


Figura 5.47: Interfaces del Agente de Planificación.

C. Diagrama de contexto

Figura 5.48: Diagrama de contexto del Agente de Planificación

D. Guía de variabilidad

El código del planificador que resuelve un problema debe ser cargado en tiempo de ejecución (ver apartado de “Otra información”).

E. Información sobre la arquitectura

E.1. Decisiones de diseño

Lo más destacable de esta vista es el elemento denominado Wrapper. Como su propio nombre indica, es una envoltura que proporciona un mecanismo de comunicación del agente con el planificador, al tiempo que oculta las peculiaridades del planificador al Agente de Planificación.

El propósito de utilizar el Wrapper consiste en facilitar el cambio del planificador por otro distinto intentando que, en la medida de lo posible, este cambio no afecte al Agente de Planificación y, mucho menos, al resto de los agentes. De esta manera, para cada planificador se desarrollaría un Wrapper específico que implementaría una interfaz común para todos estos envoltorios.

E.2. Análisis de resultados

Todos los resultados aparecen descritos en la sección 5.6.

F. Otra información

Como en los casos anteriores y por recomendación del equipo de trabajo, la implementación del planificador se va a realizar utilizando JShop2[6], que es una implementación de un planificador Shop2 sobre el lenguaje Java. JShop2 tiene la peculiaridad de que, para mejorar la eficiencia del planificador, genera un código específico para cada plan realizado, lo que obliga a cargar este código en tiempo de ejecución.

5.5.1.12. Vista de Descomposición

Agente de Trayectoria

Presentación de la vista

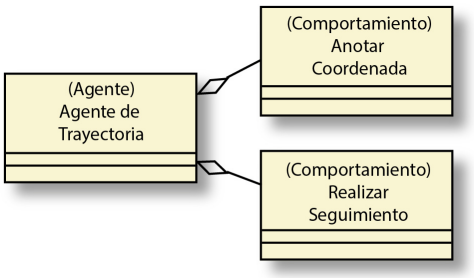


Figura 5.49: Descomposición del Agente de Trayectoria

B. Catálogo de elementos

B.1. Elementos y sus propiedades

Elementos	Propiedades

6 Es un sistema de planificación independiente del dominio basado en HTN (Hierarchical Task Network). Al igual que su predecesor SHOP, SHOP2 genera los pasos de cada uno de las planificaciones en el mismo orden en el que posteriormente serán ejecutadas, por tanto se conoce el estado actual de cada uno de los pasos del proceso de planificación.

Agente de Trayectoria	Este elemento, aunque lleva el mismo nombre que el módulo que lo contiene, constituye en sí lo que es el agente, el cual tiene asociados una serie de comportamientos que llevan a cabo la mayor parte de las responsabilidades que le corresponden a éste. La responsabilidad principal de este elemento consiste en la creación y puesta en marcha inicial de los comportamientos, así como ejercer de representante de los demás elementos ante el resto de los agentes, enviando y recibiendo mensajes.
Anotar Coordenada	La responsabilidad de este comportamiento consiste en recibir las posiciones de un estudiante en el EV e ir guardándolas para que puedan ser usadas para realizar el seguimiento de la trayectoria del alumno.
Realizar Seguimiento	Este comportamiento está encargado de calcular la trayectoria entre dos puntos y compararla con la que realiza el estudiante.

Tabla 5.23: Elenentos y propiedades del Agente de Trayectoria

B.2. Relaciones y sus propiedades

La relación que dirige la descomposición de esta vista es forma-parte-de. No hay excepciones ni añadidos a lo que define esta relación.

Elementos	Propiedades
Agente de Trayectoria	Proporciona interfaces para enviar y recibir mensajes de otros agentes. Asume la existencia de un agente que le proporciona las coordenadas con las que trabaja.
Anotar Coordenada	Proporciona un servicio para recibir las coordenadas de los desplazamientos del estudiante.
Realizar Seguimiento	Proporciona servicios para iniciar y detener el seguimiento de los desplazamientos de un estudiante, así como para facilitar el resultado de dicho seguimiento.

Tabla 5.24: Relaciones y propiedades del Agente de Trayectoria

B.3. Interfaces

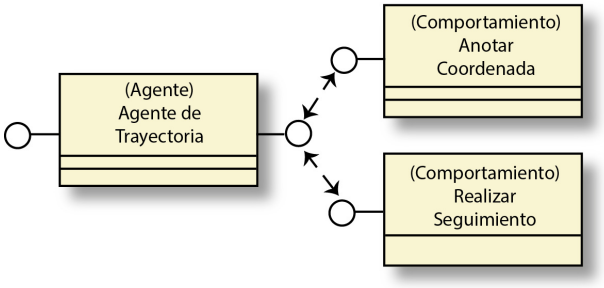


Figura 5.50: Interfaces del Agente de Trayectoria

C. Diagrama de contexto

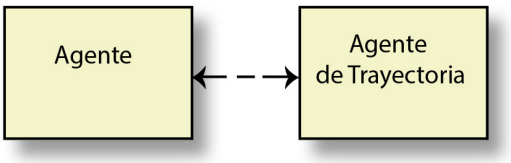


Figura 5.51: Diagrama de contexto del Agente de Trayectoria

D. Guía de variabilidad

El seguimiento de la trayectoria de los estudiantes es algo que no se va a realizar continuamente, sino sólo cuando el estudiante deba moverse. Por este motivo, el comportamiento de seguimiento debe poder activarse y desactivarse a voluntad.

E. Información sobre la arquitectura

E.1. Decisiones de diseño

Se ha decidido separar en dos comportamientos la anotación de coordenadas y el seguimiento por si se desea deshabilitar este último pero manteniendo el registro de los desplazamientos del estudiante.

E.2. Análisis de resultados

Todos los resultados aparecen descritos en la sección 5.6.

E.3. Suposiciones

Como ya se ha mencionado anteriormente, el seguimiento de la trayectoria del estudiante es uno de los elementos que puede hacer que la velocidad de ejecución de la aplicación disminuya. Por lo tanto, se ha supuesto que el algoritmo de seguimiento de la trayectoria del estudiante puede ejecutarse en tiempo real. Si es preciso, deberá disminuirse la precisión del algoritmo, trabajando con menos coordenadas y flexibilizando los criterios utilizados para decidir si la trayectoria del alumno se ajusta a una trayectoria ideal.

5.5.1.13. Vista de Descomposición

Agente de Estudiante

Presentación de la vista

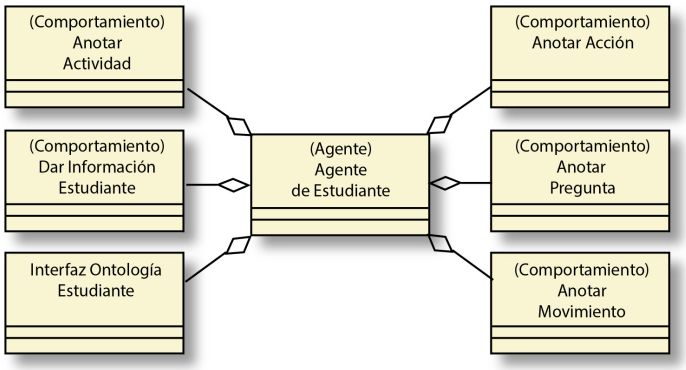


Figura 5.52: Descomposición del Agente de Estudiante

B. Catálogo de elementos

B.1. Elementos y sus propiedades

B.2. Relaciones y sus propiedades

La relación que dirige la descomposición de esta vista es forma-parte-de. No hay excepciones ni añadidos a lo que define esta relación.

Elementos	Propiedades
Agente de Estudiante	La responsabilidad principal de este elemento consiste en la creación y puesta en marcha inicial de los comportamientos, así como ejercer de representante de los demás elementos ante el resto de los agentes, enviando y recibiendo mensajes.
Anotar Acción	Este comportamiento recibe las acciones realizadas por un estudiante durante la realización de una actividad, junto con la evaluación de su corrección. Esta acción es almacenada y utilizada para completar el modelo del estudiante
Anotar Pregunta	La responsabilidad de este comportamiento consiste en almacenar las preguntas realizadas por un estudiante, junto con la evaluación de su pertinencia, así como en usar de esta información para completar el modelo del estudiante.
Anotar Movimiento	Este comportamiento se encarga de recibir información de los desplazamientos del estudiante, junto con su evaluación, para almacenarlo y utilizarlo en la realización del modelo del estudiante.
Anotar Actividad	Este comportamiento tiene como responsabilidad el almacenamiento y procesamiento de la evaluación de una actividad en su conjunto para completar la realización del modelo del estudiante.

Dar Información del Estudiante	La responsabilidad de este comportamiento consiste en proporcionar la información del estudiante contenida hasta el momento en el modelo del estudiante. La información que proporciona está relacionada con las acciones, preguntas y movimientos del estudiante, así como todo lo que se pueda inferir durante la realización del modelo del estudiante, como la capacidad de aprendizaje, la eficiencia en la realización de los ejercicios o el grado de autonomía del estudiante.
Interfaz con la Ontología del Estudiante	Este elemento se encarga de facilitar el intercambio de información entre el Agente de Estudiante y la Ontología del Estudiante, así como de realizar las traducciones correspondientes entre uno y otro.

Tabla 5.25: Elementos y propiedades del Agente de Estudiante

B.3. Interfaces

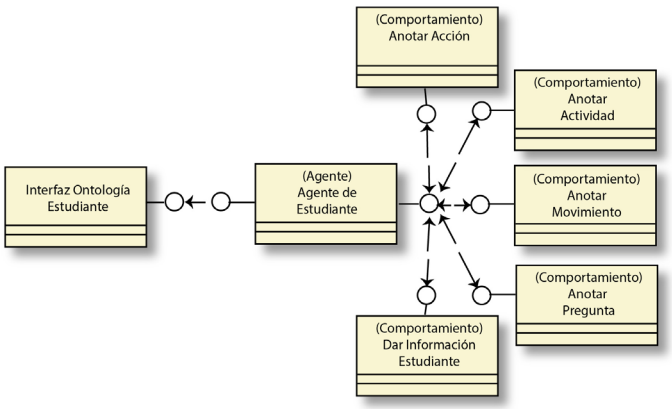


Figura 5.53: Interfaces del Agente de Estudiante

C. Diagrama de contexto

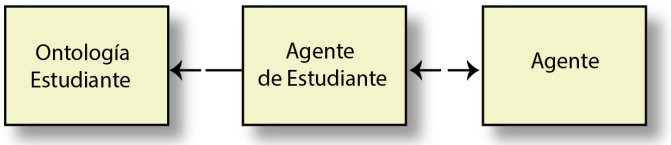


Figura 5.54: Diagrama de contexto del Agente de Estudiante

D. Guía de variabilidad

Existirá un agente de estudiante por cada estudiante conectado al prototipo, de modo que cada agente mantendrá por separado el modelo de cada uno de los estudiantes.

E. Información sobre la arquitectura

E.1. Decisiones de diseño

Aunque el movimiento se puede considerar como una acción dentro del EV, se ha decidido considerarlo de manera separada porque la evaluación que hay que realizar sobre el mismo es diferente a la del resto de las acciones, y los agentes involucrados pueden ser distintos.

Se ha considerado la posibilidad de que el modelado de todos los estudiantes lo realice un único agente, aunque finalmente se ha optado por que cada estudiante tenga su propio agente de estudiante. De esta manera, entre otros beneficios, se espera facilitar la distribución de los agentes en distintas máquinas en caso de que sea necesario tomar esta medida. También se simplifica el diseño del modelo del estudiante y se facilita el traslado del perfil del estudiante entre distintos cursos e, incluso, entre distintas plataformas.

E.2. Análisis de resultados

Todos los resultados aparecen descritos en la sección 5.6.

5.5.1.14. Vista de Descomposición Agente de Tutoría

Presentación de la vista

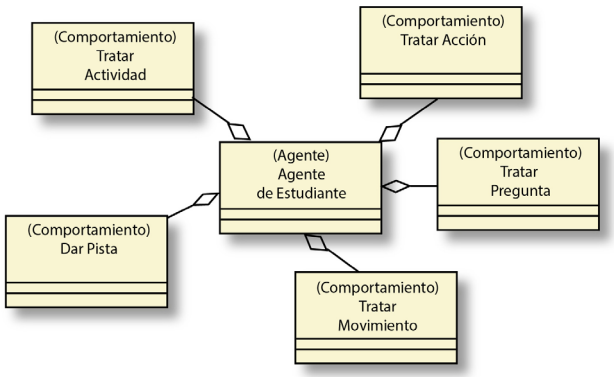


Figura 5.55: Descomposición del Agente de Tutoría

B. Catálogo de elementos

B.1. Elementos y sus propiedades

Elementos	Propiedades
Agente de Tutoría	Este elemento constituye el agente en sí, el cual tiene asociados unos comportamientos que llevan a cabo la mayor parte de las responsabilidades que le corresponden. La responsabilidad principal de este elemento consiste en la creación y puesta en marcha de los comportamientos, así como ejercer de representante de los demás elementos ante el resto de los agentes, enviando y recibiendo mensajes.
Tratar Acción	Este comportamiento se encarga de recibir y evaluar las acciones realizadas por el estudiante para resolver una actividad. Debe comprobar si es una de las acciones que pueden llevarse a cabo para seguir con el ejercicio o, si no, si es posible realizarla. En caso de ser así, debe decidir, en función de la estrategia de tutoría, si permite la realización de la acción o si le indica al alumno que la acción no es posible.
Tratar Actividad	La responsabilidad de este comportamiento consiste en evaluar en su conjunto una actividad realizada por un alumno tras su finalización, decidiendo si se ha completado de manera satisfactoria o no.
Tratar Pregunta	Cuando un alumno realiza una pregunta, este comportamiento es el encargado de evaluar su adecuación y decidir si se responde o no, así como el nivel de detalle de la respuesta.
Tratar Movimiento	Este comportamiento está encargado de evaluar la adecuación de la trayectoria de un estudiante.
Dar Pista	Este comportamiento está encargado de monitorizar el comportamiento del alumno. Si se cumplen las condiciones marcadas en la estrategia de tutoría en uso, se detectará que el alumno no sabe cómo continuar y se le ofrecerán pistas con distinto nivel de detalle para ayudarle a decidir cómo tiene que continuar la ejecución la actividad.

Tabla 5.26: Elementos y propiedades del Agente de Tutoría

B.2. Relaciones y sus propiedades

La relación que dirige la descomposición de esta vista es forma-parte-de. No hay excepciones ni añadidos a lo que define esta relación.

Elementos	Propiedades
Agente de Tutoría	Proporciona interfaces para enviar y recibir mensajes de otros agentes. Asume la existencia de otros agentes que le proporcionan información con la que trabajar.
Tratar Acción	Proporciona servicios para recibir las acciones realizadas por el estudiante. Asume la existencia de agentes que le ayudan a verificar la corrección o aplicabilidad de la acción ejecutada.

Tratar Actividad	Proporciona servicios para recibir notificaciones del comienzo y finalización de las actividades. Asume la existencia de algún agente que almacena información sobre lo acontecido en la realización de una actividad.
Tratar Pregunta	Proporciona servicios para recibir las preguntas realizadas por el estudiante. Asume la existencia de un agente que proporciona la respuesta en función de los parámetros presentes en la estrategia de tutoría.
Tratar Movimiento	Proporciona servicios para recibir los movimientos del estudiante. Asume la existencia de un agente que analiza la trayectoria seguida y proporciona una valoración para poder realizar la evaluación.
Dar Pista	Asume la existencia de algún agente que registra las acciones del estudiante.

Tabla 5.27: Relaciones y propiedades del Agente de Tutoría

B.3. Interfaces

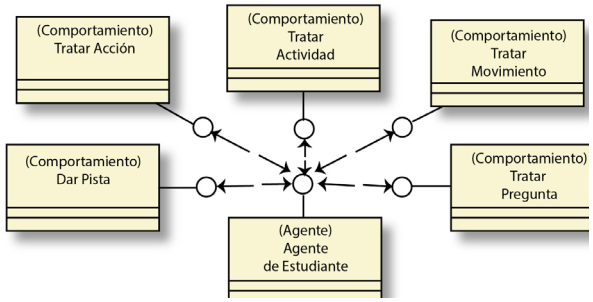


Figura 5.56: Interfaces del Agente de Tutoría

C. Diagrama de contexto

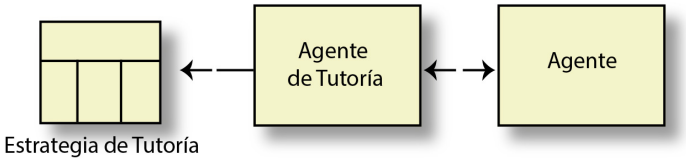


Figura 5.57: Diagrama de contexto del Agente de Tutoría

D. Guía de variabilidad

Algunos de los comportamientos descritos pueden no estar presentes si así lo determinan la estrategia de tutoría, el estudiante o la actividad a ejecutar. Así, el comportamiento de tratar movimientos no se ejecutará en actividades que no requieran un desplazamiento por el EV. El comportamiento de dar pistas no se ejecutará si la estrategia de tutoría así lo determina o si el estudiante solicita que se supriman las pistas. También es posible que se supriman las respuestas ante las preguntas realizadas por el estudiante.

E. Información sobre la arquitectura

E.1. Decisiones de diseño

Existe la posibilidad de activar y desactivar, ya sea de forma dinámica o no, muchas de las funcionalidades proporcionadas por el tutor. Disponer de cada una de esas funcionalidades en un comportamiento separado del agente facilita la labor de poder activarlas y desactivarlas de manera independiente, sin más que activar o desactivar el comportamiento que se encarga de realizarlas. Además de los aspectos mencionados con las pistas o los movimientos, se permite que no se respondan preguntas si, por ejemplo, no se ha definido esa funcionalidad en un dominio o para un problema determinado. De igual forma, se puede deshabilitar el tratamiento de acciones en caso de que exista un modo no supervisado en el que el estudiante pueda explorar el EV.

Por otro lado, dadas las características y funcionalidades de este agente, depende en gran medida de otros agentes presentes en el entorno, que deben ayudarle a validar las acciones del estudiante, almacenar la traza de las acciones que realiza, registrar y analizar sus movimientos o proporcionarle la respuesta a sus preguntas. De esta forma, la estrategia de tutoría puede proporcionar a los comportamientos del agente acciones por omisión ante la inexistencia de alguno de los agentes necesarios para llevar sus responsabilidades a buen fin. Así, por ejemplo, puede no permitirse la ejecución de una acción que no sea estrictamente la que corresponda según el plan o, por el contrario, puede permitirse la ejecución de cualquier acción que sea físicamente posible dentro del EV.

E.2. Análisis de resultados

Todos los resultados se pueden revisar en el apartado 5.6.

5.5.2. Vista de Uso : Sistema de Tutoría Inteligente

Presentación de la vista

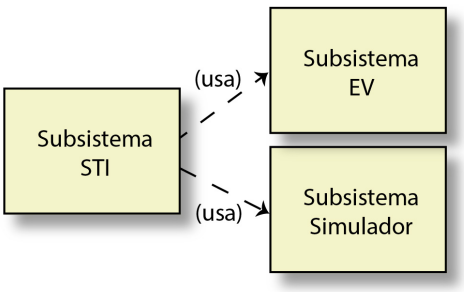


Figura 5.58: Módulos usados por el STI

B. Catálogo de elementos

B.1. Elementos y sus propiedades

Los elementos presentados en esta vista son los mismos que se pueden encontrar en la vista de descomposición del prototipo (sección 5.5.1.1).

B.2. Relaciones y sus propiedades

Un elemento usa a otro si el correcto funcionamiento del primero depende del correcto funcionamiento del segundo. No es suficiente con que el primero realice una llamada al segundo, ni tampoco necesario.

B.3. Interfaces

Las interfaces entre los distintos módulos de esta vista son las descritas en la vista de descomposición del prototipo (sección 5.5.1.1).

C. Diagrama de contexto

El contexto de este paquete es el mostrado en la vista de descomposición del prototipo (sección 5.5.1.1).

E. Información sobre la arquitectura

E.1. Decisiones de diseño

Aunque la comunicación entre los distintos elementos del prototipo se produce a través del Centro de Mensajes,éste no es más que un medio para llevar a cabo dicha comunicación, por lo que la relación de uso efectiva se produce entre el Sistema Inteligente de Tutoría y los dos módulos restantes: Entorno Virtual y Simulador.

E.2. Análisis de resultados

Todos los resultados se pueden revisar en el apartado 5.6.

E.3. Suposiciones

Como se ha mencionado anteriormente, se ha supuesto que tanto el Simulador como el Entorno Virtual tienen al STI como un posible usuario más, si bien no necesitan de él para funcionar, razón por la cual la relación de uso se produce en un solo sentido.

5.5.2.2. Vista de Uso : Agente de Tutoría

Presentación de la vista

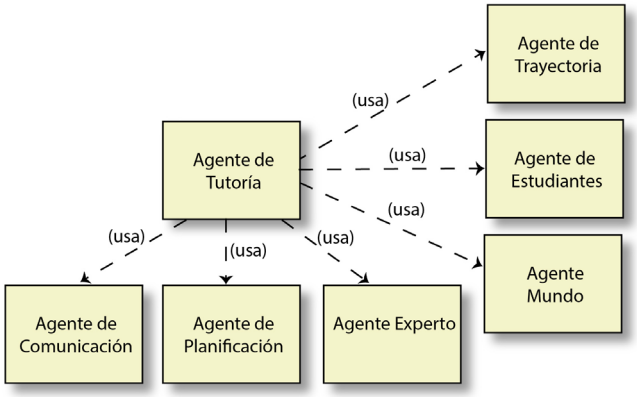


Figura 5.59: Módulos usados por el Agente de Tutoría

B. Catálogo de elementos

B.1. Elementos y sus propiedades

Los elementos presentados en esta vista son los mismos que pueden encontrarse en la vista de descomposición del Sistema Inteligente de Tutoría (sección 5.5.1.5).

B.2. Relaciones y sus propiedades

La relación que se establece entre los módulos de esta vista es usa. Un elemento usa a otro si el correcto funcionamiento del primero depende del correcto funcionamiento del segundo. No es suficiente con que el primero realice una llamada al segundo, ni tampoco necesario.Como decisión de diseño, se ha optado por que los agentes realicen acciones por omisión o ignoren ciertos datos en caso de no existir quien los proporcione, lo que posibilita que la relación de uso se reduzca al mínimo. No obstante, se van a mostrar las relaciones de uso como serían sin las acciones por omisión. De esta manera se da una visión real de las necesidades de los agentes para que el prototipo funcione con todas sus características.

B.3. Interfaces

Las interfaces entre los distintos módulos de esta vista son las descritas en la vista de descomposición correspondiente (sección 5.5.1.5).

C. Diagrama de contexto

El contexto de este paquete es el mostrado en la vista de descomposición del Sistema Inteligente de Tutoría (sección 5.5.1.5).

E. Información sobre la arquitectura

E.1. Decisiones de diseño

En los escenarios de modificabilidad planteados para la construcción del prototipo se establece la necesidad de que ciertas funcionalidades puedan deshabilitarse o incluso no existir. Por este motivo, aunque la relación de uso existe cuando las distintas funcionalidades están activas, el agente de tutoría no debe necesitar la presencia de los demás agentes si no va a precisar de sus servicios. Por ello, la relación entre agentes debe establecerse en tiempo de ejecución y sólo a partir del momento en que el agente de tutoría necesite los servicios de otro. Además, este conocimiento no debe establecerse a priori, sino a través de un mecanismo como las páginas amarillas, que permiten cambiar de manera dinámica a un agente que preste un determinado servicio por otro. Por otro lado, aunque la relación se establece entre agentes, deben ser los comportamientos del agente de tutoría los que, de manera individual, soliciten establecer contacto con los agentes que necesiten.

E.2. Análisis de resultados

Todos los resultados se pueden revisar en el apartado 5.6.

5.5.2.3. Vista de Uso : Agente de Estudiante

Presentación de la vista

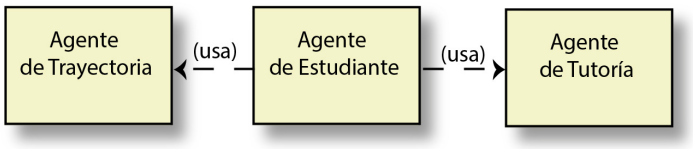


Figura 5.60: Módulos usados por el Agente de Estudiante

B. Catálogo de elementos

B.1. Elementos y sus propiedades

Los elementos presentados en esta vista son los mismos que pueden encontrarse en la vista de descomposición del Sistema Inteligente de Tutoría (sección 5.5.1.5).

B.2. Relaciones y sus propiedades

La relación que se establece entre los módulos de esta vista es *"usa"*. Un elemento usa a otro si el correcto funcionamiento del primero depende del correcto funcionamiento del segundo. No es suficiente con que el primero realice una llamada al segundo, ni tampoco necesario.

Como decisión de diseño, se ha optado por que los agentes realicen acciones por omisión o ignoren ciertos datos en caso de no existir quien los proporcione, lo que posibilita que la relación de uso se reduzca al mínimo. No obstante, se van a mostrar las relaciones de uso como serían sin las acciones por omisión. De esta manera se da una visión real de las necesidades de los agentes para que el prototipo funcione con todas sus características.

B.3. Interfaces

Las interfaces entre los distintos módulos de esta vista son las descritas en la vista de descomposición correspondiente (sección 5.5.1.5).

C. Diagrama de contexto

El contexto de este paquete es el mostrado en la vista de descomposición del Sistema Inteligente de Tutoría (sección 5.5.1.5).

E. Información sobre la arquitectura

E.1. Decisiones de diseño

El agente de estudiante resulta fundamental para el buen funcionamiento de un Sistema Inteligente de Tutoría, ya que es en gran medida quien posibilita la adaptación del la enseñanza las necesidades del estudiante. Por ese motivo, su presencia suele darse por descontada, y es habitual que todos los datos relativos a los alumnos se le envíen directamente. Sin embargo, manteniendo la filosofía de diseño del prototipo, y contemplando la posibilidad de que distintos modelos de estudiante necesiten distinta información para su elaboración, se ha optado por que sea el agente de estudiante, por indicación de sus comportamientos, quien solicite la información necesaria para elaborar el modelo del estudiante que se encuentre a su cargo.

E.2. Análisis de resultados

Ver sección 5.6

5.5.2.4. Vista de Uso : Agente de Trayectoria

Presentación de la vista

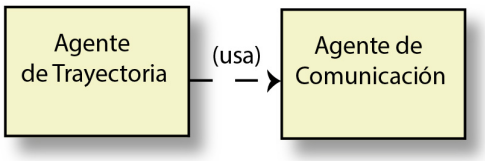


Figura 5.61: Módulos usados por el Agente de Trayectoria

B. Catálogo de elementos

B.1. Elementos y sus propiedades

Los elementos presentados en esta vista son los mismos que pueden encontrarse en la vista de descomposición del Sistema de Tutoría Inteligente (sección 5.5.1.5).

B.2. Relaciones y sus propiedades

La relación que se establece entre los módulos de esta vista es usa. Un elemento usa a otro si el correcto funcionamiento del primero depende del correcto funcionamiento del segundo. No es suficiente con que el primero realice una llamada al segundo, ni tampoco necesario.

Como decisión de diseño, se ha optado por que los agentes realicen acciones por omisión o ignoren ciertos datos en caso de no existir quien los proporcione, lo que posibilita que la relación de uso se reduzca al mínimo. No obstante, se van a mostrar las relaciones de uso como serían sin las acciones por omisión. De esta manera se da una visión real de las necesidades de los agentes para que el prototipo funcione con todas sus características.

B.3. Interfaces

Las interfaces entre los distintos módulos de esta vista son las descritas en la vista de descomposición correspondiente (sección 5.5.1.5).

C. Diagrama de contexto

El contexto de este paquete es el mostrado en la vista de descomposición del Sistema de Tutoría Inteligente (sección 5.5.1.5).

E. Información sobre la arquitectura

E.1. Decisiones de diseño

Al igual que con otros agentes, se ha optado por que sea el agente que necesita una información quien la solicite, en lugar de que sea el agente que la proporciona quien se la ofrezca. Por

este motivo, la relación de uso es del agente de trayectoria hacia el de comunicación, y no al contrario, debido a que es éste quien solicita las posiciones del estudiante para poder realizar el seguimiento de la trayectoria. Para ello utiliza el mecanismo de edición-suscripción que se describe en la sección 5.5.3.

E.2. Análisis de resultados

Ver sección 5.6.

5.5.2.5. Vista de Uso: Agente Global o Mundo

Presentación de la vista

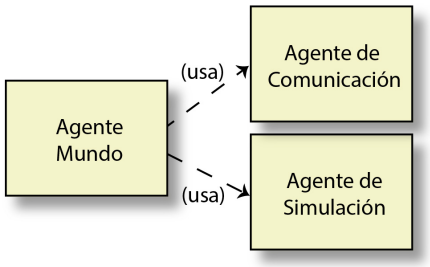


Figura 5.62: Módulos usados por el Agente Mundo

B. Catálogo de elementos

B.1. Elementos y sus propiedades

Los elementos presentados en esta vista son los mismos que pueden encontrarse en la vista de descomposición del Sistema de Tutoría Inteligente (sección 5.5.1.5).

B.2. Relaciones y sus propiedades

Como decisión de diseño, se ha optado por que los agentes realicen acciones por omisión o ignoren ciertos datos en caso de no existir quien los proporcione, lo que posibilita que la relación de uso se reduzca al mínimo. No obstante, se van a mostrar las relaciones de uso como serían sin las acciones por omisión. De esta manera se da una visión real de las necesidades de los agentes para que el prototipo funcione con todas sus características.

B.3. Interfaces

Las interfaces entre los distintos módulos de esta vista son las descritas en la vista de descomposición correspondiente (sección 5.5.1.5).

C. Diagrama de contexto

El contexto de este paquete es el mostrado en la vista de descomposición del Sistema de Tutoría Inteligente(sección 5.5.1.5).

E. Información sobre la arquitectura

E.1. Decisiones de diseño

La principal responsabilidad del agente mundo consiste en mantener un modelo actualizado del estado del entorno de entrenamiento, de forma que el resto de los agentes puedan solicitarle esta información cuando la necesiten. De tal forma, las fuentes de cambio en el estado del mundo son los sucesos que puedan ocurrir dentro del propio entorno de entrenamiento, producidos, por ejemplo, por acciones de los estudiantes, y los cambios en elementos controlados por la simulación. La forma de obtener información sobre estos cambios es, por tanto, a través del agente de simulación y de los de comunicación.

E.2. Análisis de resultados

Todos los resultados se pueden revisar en el apartado 5.6.

5.5.2.6. Vista de Uso : Agente Experto

Presentación de la vista

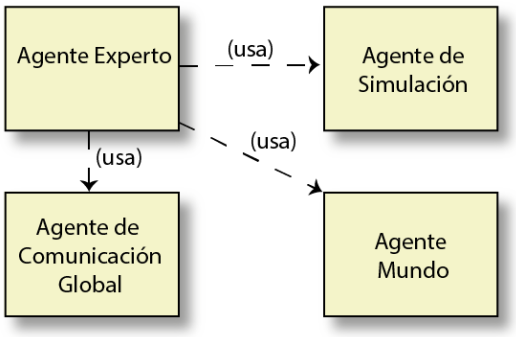


Figura 5.63: Módulos usados por el Agente Experto

B. Catálogo de elementos

B.1. Elementos y sus propiedades

Los elementos presentados en esta vista son los mismos que pueden encontrarse en la vista de descomposición del Sistema de Tutoría Inteligente (sección 5.5.1.5).

B.2. Relaciones y sus propiedades

Como decisión de diseño, se ha optado por que los agentes realicen acciones por omisión o ignoren ciertos datos en caso de no existir quien los proporcione, lo que posibilita que la relación de uso se reduzca al mínimo. No obstante, se van a mostrar las relaciones de uso como serían sin las acciones por omisión. De esta manera se da una visión real de las necesidades de los agentes para que el prototipo funcione con todas sus características.

B.3. Interfaces

Las interfaces entre los distintos módulos de esta vista son las descritas en la vista de descomposición correspondiente (sección 5.5.1.5).

C. Diagrama de contexto

El contexto de este paquete es el mostrado en la vista de descomposición del Sistema Inteligente de Tutoría (sección 5.5.1.5).

E. Información sobre la arquitectura

E.1. Decisiones de diseño

Las responsabilidades del agente experto están relacionadas con las acciones realizadas por los estudiantes. Por una parte, debe comprobar que la acción que el estudiante intenta realizar puede llevarse a cabo. Para ello, debe comprobar que tanto las condiciones del mundo permiten ejecutar la acción, como que el estado de la simulación también lo hace posible. Para realizar cada una de estas verificaciones necesita los servicios proporcionados por el agente mundo y el agente de simulación. Por otra parte, una vez que se ha considerado que la acción puede llevarse a cabo, tiene efectos que se le deban mostrar a los alumnos, será responsabilidad del agente experto enviar las notificaciones pertinentes, lo que hará a través del agente de comunicación global.

E.2. Análisis de resultados

Todos los resultados se pueden revisar en el apartado 5.6.

5.5.2.7. Vista de Uso: Agente de Simulación

Presentación de la vista

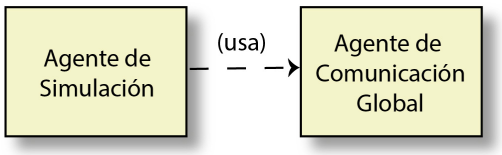


Figura 5.64: Módulos usados por el Agente de Simulación

B. Catálogo de elementos

B.1. Elementos y sus propiedades

Los elementos presentados en esta vista son los mismos que pueden encontrarse en la vista de descomposición del Sistema de Tutoría Inteligente (sección 5.5.1.5).

B.2. Relaciones y sus propiedades

Como decisión de diseño, se ha optado por que los agentes realicen acciones por omisión o ignoren ciertos datos en caso de no existir quien los proporcione, lo que posibilita que la relación de uso se reduzca al mínimo. No obstante, se van a mostrar las relaciones de uso como serían sin las acciones por omisión. De esta manera se da una visión real de las necesidades de los agentes para que el prototipo funcione con todas sus características.

B.3. Interfaces

Las interfaces entre los distintos módulos de esta vista son las descritas en la vista de descomposición correspondiente (sección 5.5.1.5).

C. Diagrama de contexto

El contexto de este paquete es el mostrado en la vista de descomposición del Sistema Inteligente de Tutoría (sección 5.5.1.5).

E. Información sobre la arquitectura

E.1. Decisiones de diseño

La responsabilidad del agente de simulación consiste en notificar los cambios que se producen en la simulación para que, en caso necesario, se muestren en el EV. Para ello, envía la notificación de estos cambios a los alumnos a través del agente de comunicación global.

E.2. Análisis de resultados

Ver sección 5.6

5.5.3. Vista Editor-Suscriptor

Entre las tácticas mencionadas en (Bass y otros, 2003) para aumentar la modificabilidad de un sistema software se encuentra la utilización de un comportamiento del tipo editor-subscriptor. Esto es así principalmente por dos razones. En primer lugar, se consigue que los productores y consumidores de información se conozcan en tiempo de ejecución, pero no antes, lo cual permite una modificación más sencilla de los diseños. Por otro lado, también se desacoplan los productores y consumidores de información, que desconocen su identidad y, por tanto, detalles que puedan comprometer a uno de ellos cuando el otro es modificado - salvo, por supuesto, los relativos al servicio al que se suscriben.

Para documentar esta vista se supone la existencia de un servicio de páginas amarillas, donde los productores van a anunciar los servicios que proporcionan y donde los consumidores buscarán los servicios que necesiten para llevar a cabo las tareas que tienen encomendadas. Esta suposición está soportada por la existencia del mencionado servicio dentro del estándar FIPA para la arquitectura de plataformas multiagente (IEEE-FIPA, 2002a).

Todos los agentes del prototipo van a anunciar en las páginas amarillas los servicios - o información - que pueden proporcionar a otros agentes. Sin embargo, estos anuncios no siempre van a dar lugar a suscripciones. En los casos en los que un agente necesite conocer un cambio siempre que se produzca sí tendrá lugar la mencionada subscripción. Por el contrario, cuando un agente esté interesado en un servicio sólo en determinadas ocasiones, lo que hará en lugar de suscribirse y descartarlo cuando no lo necesite será solicitarlo expresamente cuando así lo desee. En cualquiera de los dos casos, el agente que realiza el anuncio desconoce quiénes son los consumidores ni si, a priori, van a suscribirse al servicio o no. Por su parte, los consumidores

no conocen expresamente al agente que realiza el anuncio, sino que lo hacen a través de lo que aparece en las páginas amarillas, por lo que éste agente podría cambiar.

5.5.3.1. Vista Editor-Suscriptor

Sistema. Presentación de la vista

Una de las necesidades que se busca satisfacer con esta arquitectura es poder conectar y desconectar los diferentes subsistemas que la forman sin que los demás elementos de la arquitectura se vean afectados. Por ello, se ha pensado que una buena solución consiste en la utilización de un mecanismo de edición-suscripción que funcione a través de un centro de mensajes, que constituye el único elemento conocido por los demás.

Su función principal consiste en esperar notificaciones de otras aplicaciones que desean recibir determinado tipo de mensajes y reenviar los mensajes recibidos a los destinatarios que se hayan suscrito a ellos. Así, si alguno de los elementos del prototipo no es necesario para la realización de una actividad, porque ninguno de sus servicios es requerido por los demás subsistemas, se podrá detener su ejecución sin que el resto note ninguna diferencia. Si a esto unimos el hecho de que el Centro de Mensajes ignora todos los detalles relativos al resto de aplicaciones, puesto que lo que hace es enviar y recibir mensajes sin preocuparse de quién los envía, quién los recibe o qué contienen, se puede esperar un grado de acoplamiento bastante bajo entre los distintos subsistemas que conforman el prototipo.

Elemento	Servicio	Propósito
Centro de Mensajes	Suscribirse a mensaje	Proporciona la posibilidad de que cualquier sistema se conecte con el Centro de Mensajes para recibir mensajes enviados por otras aplicaciones conectadas a él. Aunque el uso habitual es suscribirse para recibir uno o varios tipos de mensajes en función del tipo de contenido, también es posible realizar suscripciones en función del emisor del mensaje.
	Borrar suscripción a mensaje	Cuando una aplicación no desea recibir más mensajes de un tipo al que se ha suscrito previamente, puede utilizar este servicio para que dejen de serle enviados. Así, si una aplicación se va a cerrar, el comportamiento esperado es que borre todas las suscripciones que realizó. Si no se procede de esta manera, cuando el Centro de Mensajes detecte errores en los envíos procederá a borrar las suscripciones correspondientes.

Sistema de Tutoría Inteligente	Registrar estudiante	El propósito de este servicio es indicar al sistema de tutoría la existencia de un nuevo estudiante, para que pueda integrarlo dentro de una actividad y monitorizar sus acciones.
	Registrar acción	El propósito de este servicio es monitorizar las acciones del estudiante para poder realizar su seguimiento. Este servicio se presta por el hecho de que el estudiante se haya registrado en el prototipo. La respuesta ante una acción está en función de la estrategia de tutoría en uso.
	Registrar posición	El propósito de este servicio es hacer un seguimiento de los desplazamientos del estudiante por el entorno. Este servicio se presta por el hecho de que el estudiante se haya registrado en el prototipo, aunque puede no estar activo si la actividad objeto de aprendizaje, no requiere desplazamientos. La respuesta variará en función de la estrategia de tutoría.
	Contestar pregunta	El propósito de este servicio es que el estudiante pueda solicitar ayuda cuando lo necesite. Este servicio se presta por el hecho de que el estudiante se haya registrado en el sistema. La respuesta variará en función de la estrategia de tutoría.
Simulador	Realizar acción	El propósito de este servicio es que otras aplicaciones puedan solicitar al simulador que se realice una determinada acción dentro de la simulación.
	Cambiar posición	El propósito de este servicio es que una aplicación externa pueda solicitarle al simulador el cambio en la posición de un elemento de la si- mulación.
	Consultar estado	El propósito de este servicio es que una aplicación pueda consultar el estado de un elemento de la simulación.
	Notificar cambio	Este servicio está pensado para que el simulador notifique cambios en elementos de la simulación cada vez que se produzcan. Este servicio sí necesita suscripción.
Entorno Virtual	Realizar acción	Este servicio permite que una aplicación externa solicite al EV que un elemento del mismo, habitualmente un avatar, realice una determinada acción dentro del entorno.
	Cambiar posición	Este servicio permite que una aplicación solicite al EV el cambio en la posición de un objeto o personaje que se encuentra dentro de éste.
	Mostrar mensaje	Este servicio permite que se solicite al EV que reproduzca un mensaje dentro del mismo, lo que posibilita, por ejemplo, recibir las contestaciones del STI a las preguntas de los alumnos.

Tabla 5.28: Elementos, servicios y propósitos del Prototipo.

B. Catálogo de elementos

B.1. Elementos y sus propiedades

Cada uno de los elementos que conforman el prototipo proporcionan los servicios indicados en la siguiente tabla quemuestra, para cada elemento del prototipo, quién le proporciona los servicios que necesita y para qué los utiliza.

Elemento	Servicio	Propósito
Centro de Mensajes		
		El propósito del centro de mensajes, como su nombre indica, es distribuir los mensajes a los que se suscriben el resto de elementos. El conocimiento que tiene acerca de ellos no va más allá de la capacidad de enviarles los mensajes a los que se han suscrito, por lo que no utiliza servicios de ninguna otra aplicación que pueda formar parte del sistema de enseñanza.
Sistema Inteligente de Tutoría		
Centro de Mensajes	Suscribirse a mensaje	Mediante este servicio, el STI se suscribe a la recepción de los mensajes de los tipos que le interesen y que sean enviados a través del Centro de Mensajes.
	Borrar suscripción a mensaje	Mediante este servicio, el STI notifica que no quiere continuar recibiendo un determinado tipo de mensaje que le llega a través del Centro de Mensajes.
Simulador	Realizar acción	Cuando el STI necesita ejecutar una acción que puede tener efectos sobre la simulación, provoca que se ejecute este servicio de manera que el estado de la simulación se actualice convenientemente.
	Cambiar posición	Cuando el STI necesita modificar la posición de un elemento presente en la simulación, provoca la ejecución de este servicio de manera que el estado de la simulación se actualice en consecuencia.
	Consultar estado	Cuando el STI necesita conocer el estado de un elemento de la simulación, hace uso de este servicio para solicitar la información que precise conocer.
	Notificar cambio	Si el STI necesita conocer de forma continuada los cambios que se van produciendo en determinados elementos de la simulación, solicita que este servicio se los vaya notificando cuando sucedan.
Entorno Virtual	Realizar acción	Cuando el STI necesita ejecutar una acción que puede tener efectos sobre el EV, provoca que se ejecute este servicio de manera que el estado del mismo se actualice convenientemente.

	Cambiar posición	Cuando el STI necesita modificar la posición de un elemento presente en el EV, provoca que se ejecute este servicio de manera que el estado del mismo se actualice en consecuencia.
	Mostrar mensaje	Cuando el STI necediya enviar mensajes a un alumno que se está entrenando a través del uso del EV, provoca la ejecución de este servicio de manera que el mensaje le pueda ser comunicado al alumno de la manera más adecuada.
Simulador		
Centro de Mensajes	Suscribirse a mensaje	Mediante este servicio, el Simulador se suscribe a la recepción de los mensajes de los tipos que le interesen y que sean enviados a través del Centro de Mensajes.
	Borrar suscripción a mensaje	Mediante este servicio, el STI notifica que no quiere continuar recibiendo un determinado tipo de mensaje que le llega a través del Centro de Mensajes.
Sistema de Tutoría Inteligente	Registrar acción	Cuando el simulador realiza una acción que puede tener efecto en el proceso de tutoría, provoca la ejecución de este servicio para que el STI tome en consideración lo que ha sucedido en la simulación.
	Registrar posición	Cuando el simulador cambia la posición de un objeto que puede tener efecto en el proceso de tutoría, provoca la ejecución de este servicio para que el STI tome en consideración lo que ha sucedido en la simulación.
Entorno Virtual		
Centro de Mensajes	Suscribirse a mensaje	Mediante este servicio, el EV se suscribe a la recepción de los mensajes de los tipos que le interesen y que sean enviados a través del Centro de Mensajes.
	Borrar suscripción a mensaje	Mediante este servicio, el STI notifica que no quiere continuar recibiendo un determinado tipo de mensaje que le llega a través del Centro de Mensajes.

Simulador	Realizar acción	Cuando el EV necesita ejecutar una acción que puede tener efectos sobre la simulación, provoca que se ejecute este servicio de manera que el estado de la simulación se actualice convenientemente.
	Cambiar posición	Cuando el EV necesita modificar la posición de un elemento presente en la simulación, provoca que se ejecute este servicio de manera que el estado de la simulación se actualice en consecuencia.
	Notificar cambio	Si el EV necesita conocer de forma continuada los cambios que se van produciendo en determinados elementos de la simulación, solicita que este servicio se los vaya notificando.
Sistema de Tutoría Inteligente	Registrar estudiante	Cuando un estudiante inicia el EV y desea tomar parte en un proceso de aprendizaje, se le comunica al STI para que tome las acciones pertinentes para que el estudiante pueda llevar a cabo el aprendizaje.
	Registrar acción	Cuando el usuario del EV realiza una acción que puede tener efecto en el proceso de tutoría, provoca que se ejecute este servicio para que el STI tome en consideración lo que ha sucedido en la simulación.
	Registrar posición	Cuando el simulador cambia la posición de un objeto que puede tener efecto en el proceso de tutoría, provoca que se ejecute este servicio para que el STI tome en consideración lo que ha sucedido en la simulación.
	Contestar pregunta	Cuando el estudiante que utiliza el EV realiza una pregunta, el EV provoca la ejecución de este servicio del STI para que proporcione al estudiante la respuesta adecuada.

Tabla 5.29: Elementos y propiedades del Sistema.

B.2. Relaciones y sus propiedades

La relación existente entre los elementos del prototipo, característica de esta vista, es de uso del Centro de Mensajes como bus que distribuye mensajes entre el resto de componentes.

B.3. Interfaces

Las interfaces que aparecen en esta vista tienen relación con el uso del Centro de Mensajes como mecanismo de intercambio de información entre el resto de los subsistemas de la arquitectura. De esta manera, el Centro de Mensajes proporciona un servicio para recibir mensajes de los

restantes subsistemas, y espera de ellos que sean capaces de recibir los mensajes que les envíe.

Para que un subsistema pueda recibir mensajes, primero debe suscribirse a los tipos de mensajes que desea recibir. Para borrar una suscripción, primero debe haberse suscrito a los mensajes que desea dejar de recibir. La suscripción y la desuscripción pueden realizarse en cualquier momento.

Para que una aplicación pueda enviar información a otras aplicaciones a través del Centro de Mensajes, lo único que tiene que hacer es enviar el mensaje correspondiente, sin necesidad de ejecutar ningún paso previo.

El formato del mensaje que se intercambia debe incluir el remitente del mismo, el destinatario (sólo en caso necesario), el tipo del mensaje y la información específica que se desea enviar en el mensaje. El tipo del mensaje es la principal información que el Centro de Mensajes utilizará para gestionar las suscripciones.

B.4. Comportamientos

En la figura que se muestra a continuación se puede ver el comportamiento que debe tener el prototipo al arrancar, momento en el que se producirán la mayor parte de las suscripciones en el Centro de Mensajes.

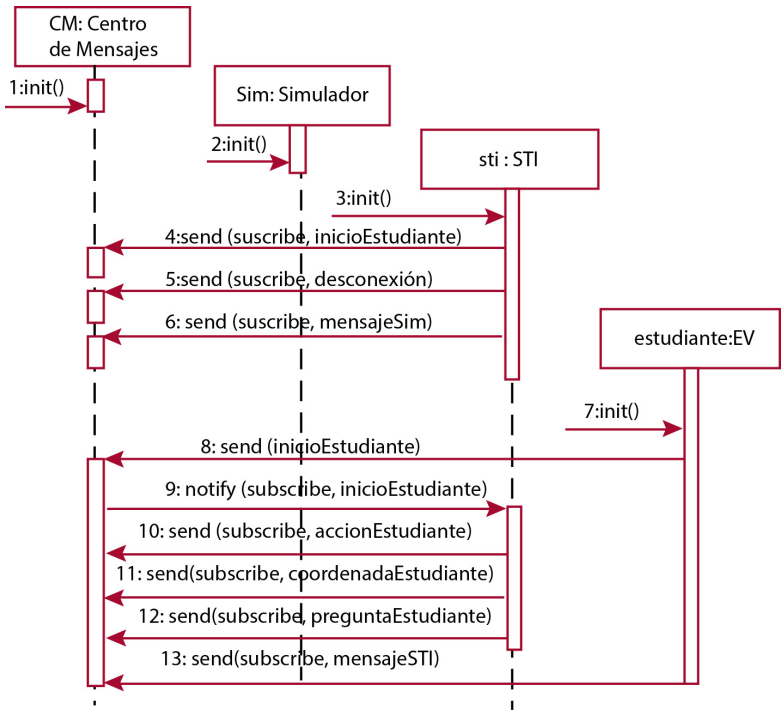


Figura 5.65: Inicio del Prototipo

En primer lugar debe iniciarse el Centro de Mensajes, para que así esté disponible cuando el resto de subsistemas se inicien y comiencen a realizar sus suscripciones. El siguiente en arrancar debe ser el Simulador, siempre y cuando nos encontremos en el caso en que es un simulador independiente que no espera una gran interacción con otras aplicaciones. En caso de que el Simulador esté integrado en el EV o en el STI, su inicio deberá producirse al mismo tiempo. A continuación debe iniciarse el STI, que se suscribe a los mensajes de inicio y finalización de sesión de un estudiante, y quedará a la espera de que éstos se conecten. Finalmente, cada estudiante que se conecte dará lugar a que el STI se suscriba a los mensajes que él envíe, y a su vez el EV del estudiante pedirá recibir los mensajes provenientes del STI. Estas últimas suscripciones pueden realizarse en paralelo, no sucediendo lo mismo con las anteriores.

C. Diagrama de contexto

Ver el diagrama de contexto mostrado en la sección 5.5.1.3.

D. Guía de variabilidad

Como se ha mencionado anteriormente, el propósito al introducir el Centro de Mensajes consiste en desacoplar lo máximo posible el resto de elementos del prototipo. Por este motivo, los subsistemas pueden enviar mensajes al Centro de Mensajes sin saber si alguien va o no a recibirlos, lo que, su su vez, posibilita que cualquier subsistema pueda suscribirse y borrarse en cualquier momento de la recepción de un tipo de mensaje, sin que el resto de subsistemas se vean afectados por ello.

Esto debe, además, ser de utilidad durante el desarrollo, ya que se puede prescindir de alguna de las aplicaciones para realizar pruebas, sustituyéndola, por ejemplo, por una consola. También puede resultar de utilidad para monitorizar todos los mensajes que pasan por el Centro de Mensajes, lo cual puede servir para depurar el funcionamiento de la aplicación.

Por otro lado, como se ha descrito en las vistas relativas al Simulador, la naturaleza de éste puede ser bastante distinta según los casos. La documentación de esta vista se ha realizado pensando que el simulador se comunicará con el resto de las aplicaciones a través del Centro de Mensajes, pero no se ha tenido en cuenta la información que pueda circular por otros cauces, ya sea de forma directa con el STI o con el EV.

E. Información sobre la arquitectura

E.1. Decisiones de diseño

En un principio se pensó en la utilización de un mecanismo directo de suscripción, por el cual cada subsistema recibiría directamente del resto la información que necesitase, sin hacer uso del centro de mensajes. Esta solución tiene la desventaja de que hace necesario saber qué aplicaciones existen para poder solicitarles la información deseada.

En lugar de ello se optó por la utilización del Centro de Mensajes, que hace posible que cada aplicación conozca únicamente la existencia de éste, aunque los mensajes provengan originalmente de una amplia variedad de fuentes. Además, también se consigue que el envío y la recepción de mensajes sea asíncrona, independizando aún más a los productores y consumidores de información.

Los productores (que hacen las veces de editores) no tienen necesidad de anunciar la información que ofrecen, mientras que los consumidores (que hacen las veces de suscriptores) deben informar al Centro de Mensajes del tipo de mensajes que deseen recibir. Es posible hacer esto así por el hecho de encontrarnos en un prototipo cerrado, de manera que siempre podemos controlar los tipos de información que se producen y se consumen.

El funcionamiento resulta entonces similar al de la invocación implícita, ya que un mensaje enviado por una aplicación actúa como un evento que hace que reaccionen todos los suscriptores que se han apuntado a la recepción de ese evento.

E.2. Análisis de resultados

Todos los resultados se pueden revisar en el apartado 5.6.

E.3. Suposiciones

En todo momento se ha evitado asumir que los elementos de la aplicación vayan a estar desarrollados con el mismo lenguaje de programación o con lenguajes que puedan integrarse directamente, como permiten hacer los métodos nativos de Java.

Además, se supone que las aplicaciones pueden ser pesadas y que pueden existir varios usuarios conectados a la vez a la aplicación de entrenamiento, lo cual hace bastante probable que las distintas aplicaciones deban funcionar dentro de una red.

Debido a los factores mencionados, se ha asumido que las aplicaciones estarán desarrolladas en lenguajes que permitan la comunicación a través de una red, preferiblemente a través de sockets, y que la comunicación con el Centro de Mensajes se realizará a través de este mecanismo. Por

este motivo, salvo que los lenguajes de programación permitan hacerlo de otra forma, también se supone que el contenido de los mensajes que se intercambien estará basado en tipos de datos simples, como números y cadenas de caracteres.

F. Otra información

Cuando se ha documentado que un elemento de la arquitectura usa un determinado servicio de otro elemento, debe quedar claro que la llamada a ese servicio no se produce de manera explícita, puesto que al Centro de Mensajes sólo se envían mensajes con información. Lo que se pretende es que el envío de un mensaje provoque la ejecución del servicio que se usa.

5.5.3.2. Vista Editor-Suscriptor Plataforma de Agentes

Presentación de la vista

La mera utilización de agentes hace suponer a muchos autores y desarrolladores que el prototipo tratado va a ser más flexible que otros sistemas construidos mediante otras tecnologías. Pero para que esto sea cierto no basta con utilizar agentes, sino que es necesario hacer uso de determinadas herramientas que éstos nos proporcionan.

Como se ha mencionado anteriormente, el uso de un mecanismo de edición-suscripción es una de las técnicas que contribuyen a obtener un prototipo más modificable. En este sentido es donde un prototipo basado en agentes puede ayudarnos a conseguir la modificabilidad deseada, ya que, si proporciona un servicio de páginas amarillas, facilitará la labor de implementación del citado mecanismo de edición-suscripción.

En este paquete se va a mostrar qué servicios proporciona cada agente. Dichos servicios serán anunciados en las páginas amarillas para que puedan ser utilizados por el resto de agentes. Aunque no todos los servicios requieran una suscripción, el hecho de que sean anunciados en las páginas amarillas permitirá que otros agentes los usen sin necesidad de saber *a priori* qué agente los proporciona. También se muestran los servicios que ejecuta cada agente como respuesta a la información que recibe a través de una suscripción.

B. Catálogo de elementos

B.1. Elementos y sus propiedades

En la siguiente tabla se muestra, para cada agente, qué servicios ofrece cada uno de sus comportamientos, que son los encargados de ofrecer dichos servicios.

Elemento	Servicio	Propósito
Agente de Comunicación Global		
Comunicación con el Centro de Mensajes	Notificar conexión	Este servicio ofrece información acerca de nuevas conexiones de estudiantes al sistema, de manera que puedan ser tenidos en cuenta por el resto de los agentes del STI que estén interesados en ello.
	Notificar desconexión	Al igual que el anterior servicio, se encarga de informar de las desconexiones de los estudiantes para que los demás agentes estén informados y puedan obrar en consecuencia.
	Enviar información	Este servicio ofrece la posibilidad de que un agente envíe información a cualquier elemento que se encuentre fuera del STI, como un mensaje que afecte a todos los alumnos conectados. Este servicio no necesita suscripción para ser utilizado.
	Recibir información	Este servicio se encarga de recibir información enviada desde el Centro de Mensajes al STI.
Gestión de Suscripciones	Solicitar Información	Este servicio ofrece la posibilidad de que un agente solicite recibir información procedente del exterior del STI. Se encarga tanto de registrar la suscripción en la tabla de suscripciones del agente como de registrarla en el Centro de Mensajes, en caso de ser necesario. Este servicio no necesita suscripción para ser utilizado.
Agente de Comunicación		
Comunicación con el Centro de Mensajes	Enviar información	Este servicio permite que un agente envíe información al estudiante cuya comunicación gestiona este agente, como pueden ser los mensajes de respuesta a una pregunta. Este servicio no necedoya suscripción para ser utilizado.
	Recibir información	Este servicio se encarga de recibir la información relativa a un estudiante enviada desde el Centro de Mensajes al STI.
	Solicitar información	Este servicio ofrece la posibilidad de que un agente solicite recibir información que llega desde el estudiante con el que se comunica este agente. Se encarga tanto de registrar la suscripción en la tabla de suscripciones del agente como de registrarla en el Centro de Mensajes, en caso de ser necesario. Este servicio no necesita suscripción para ser utilizado.
Agente de Simulación		

Comunicación con el Simulador	Enviar información	Este servicio ofrece la posibilidad de que un agente envíe información al simulador. Este servicio no necesita suscripción para ser utilizado.
	Ejecutar acción	Este servicio permite que se envíe al simulador una acción que debe ejecutarse enél. También se encarga de gestionar la respuesta a la acción que devuelve el simulador. Este servicio no necesita suscripción para ser usado.
	Recibir información	Este servicio se encarga de recibir información enviada desde el simulador al STI.
Gestión de Suscripciones	Solicitar información	Este servicio ofrece la posibilidad de que un agente solicite recibir directamente información que llega desde el simulador. Se encarga de registrar la suscripción en la tabla de suscripciones del agente. Este servicio no necesita suscripción para ser utilizado.
Agente Experto		
Validar Acciones	Validad acción	El propósito de este servicio es validar una acción realizada por un estudiante, de manera que se compruebe si es posible llevarla a cabo en función del estado actual de la actividad que se está entrenando. No necesita suscripción.
	Validar precondición	Este servicio sirve para comprobar si se cumple una única precondición de una acción para verificar si puede llevarse a cabo. No necesita suscripción.
Ejecutar Post-condiciones	Ejecutar acción	Este servicio se utiliza para ejecutar una acción y actualizar el estado de una actividad. No necesita suscripción.
	Ejecutar postcondición	Este servicio se utiliza para ejecutar una única postcondición de una acción y actualizar así el estado de una actividad. No necesita suscripción.
Agente Mundo		
Actualizar Elemento del Mundo	Actualizar posición	Este servicio se utiliza para actualizar la posición de un elemento dentro del mundo, ya sea un personaje o un objeto. No necesita suscripción.
	Actualizar estado	Este servicio se utiliza para actualizar otros atributos de un objeto o personaje distintos a su posición. No necesita suscripción.
Responder Consulta	Consultar	Este servicio se utiliza para realizar cualquier consulta de los atributos de un objeto o personaje, incluyendo su posición. No necesita suscripción.
	Validar precondición	Este servicio permite validar la precondición de una acción realizada por un alumno, siempre que esté relacionada con su ubicación o estado dentro del EV. No necesita suscripción.
Agente de Planificación		
Dar Acción	Dar acción siguiente	Este servicio devuelve la que, de acuerdo con el plan establecido por el planificador, es la siguiente acción correcta que debería ejecutarse en una actividad. No necesitasuscripción.

Aplicar Acción	Aplicar acción	Este servicio sirve para aplicar una acción sobre un plan para que se actualice el estado actual del mismo
Planificar	Planificar	Este servicio se utiliza para solicitar al agente que se elabore un nuevo plan para una actividad dada. No necesit suscripción.
	Replanificar	Este servicio se utiliza para solicitar que se elabore un nuevo plan para la actividad en curso. No necesita suscripción.
Agente de Trayectoria		
Anotar Coordenada	Anotar coordenada	Este servicio se encarga de registrar las nuevas coordenadas de la posición de un estudiante para que pueda realizarse el seguimiento de sus desplazamientos por el EV. No necesita suscripción.
Realizar Seguimiento	Iniciar seguimiento	Este servicio se encarga de solicitar el envío de las coordenadas de un alumno y de calcular la trayectoria óptima que debería seguir para moverse entre dos puntos del EV. No necesita suscripción.
	Realizar Seguimiento	Este servicio se encarga de comparar la trayectoria que realiza el alumno con la trayectoria que se supone que debería seguir. No necesita suscripción.
	Finalizar seguimiento	Este servicio se encarga de solicitar la finalización del envío de coordenadas del alumno, así como de enviar el resultado de la evaluación de la trayectoria seguida por el mismo. No necesita suscripción.
Agente de Estudiante		
Anotar Acción	Anotar acción	Mediante este servicio, el Agente de Estudiante realiza una traza de las acciones que un estudiante lleva a cabo dentro de una actividad. No se registra ninguna evaluación de la acción. No necesita suscripción.
	Anotar evaluación de acción	El propósito de este servicio es registrar una acción llevada a cabo por el estudiante junto con la evaluación de su corrección. No necesita suscripción.
Anotar Pregunta	Anotar pregunta	Este servicio sirve para que el agente de estudiante esté informado de las preguntas que realiza el estudiante, aunque no se almacena información alguna relativa a su evaluación. No necesita suscripción.
	Anotar evaluación de pregunta	El propósito de este servicio es proporcionar al agente de estudiante información acerca de una pregunta realizada por un estudiante, junto con una evaluación de la misma. No necesitasuscripción.
Anotar Movimiento	Anotar movimiento	Este servicio proporciona al Agente de Estudiante información relativa a los movimientos del estudiante por el EV. No incluye ningún tipo de evaluación de los mismos. No necesita suscripción.
	Anotar evaluación movimiento	Este servicio sirve para que el Agente de Estudiante reciba información acerca de los movimientos del estudiante por el EV, junto con una evaluación de los mismos. No necesita suscripción.

Anotar Actividad	Anotar actividad	El propósito de este servicio es que el Agente de Estudiante reciba información acerca de la actividad sobre la que el estudiante está realizando el entrenamien- to. No incluye una evaluación de la misma. No necesita suscripción.
	Anotar evaluación actividad	Este servicio sirve para que el Agente de Estudiante reciba información acerca de la actividad objeto de entrenamiento junto con una evaluación de la misma. No necesita suscripción.
Dar Información del Estudiante	Dar información de estudiante	Este servicio es el medio por el cual otros agentes pueden consultar al Agente de Estudiante información relativa al estudiante del que está a cargo. La información que se le puede requerir viene dada en función del modelo de estudiante concreto que se esté utilizando. No necesita suscripción.
Agente de Tutoría		
Tratar Acción	Tratar acción	Este servicio recibe una acción realizada por un estudiante y se encarga de comprobar si se puede llevar a cabo y si es correcta. En función de ello, la evaluá y le envía el resultado a los agentes encargados del tratamiento de esta información. Para realizar este envío es necesario que los agentes interesados se suscriban a la recepción de esta información.
Tratar Actividad	Tratar actividad	Este servicio recibe información de inicio y finalización de una actividad y se encarga de realizar una evaluación de la misma para cada estudiante y de enviar la información resultante a los agentes interesados en ella. Es necesario que los agentes interesados se suscriban a la recepción de esta información.
Tratar Movimiento	Tratar movimiento	Este servicio recibe los movimientos realizados por un estudiante y su ajuste a la trayectoria ideal, siésta existe. En función de ello, lo evalúa y le envía el resultado a los agentes encargados del tratamiento de esta información. Para realizar este envío es necesario que los agentes interesados se suscriban a la recepción de esta información.
Tratar Pregunta	Tratar pregunta	Este servicio recibe una pregunta del estudiante y se encarga de elaborar su respuesta y enviarla al estudiante. Además, realiza una evaluación de la pregunta y le envía la información correspondiente a los agentes que se hayan suscrito a la recepción de esta información.

Tabla 5.30: Servicios que ofrece cada agente.

En la siguiente tabla se puede ver, para cada agente, qué otro agente es el encargado de prestar los servicios que demanda. Los agentes que no se muestran no utilizan servicios de ningún otro agente.

Elemento	Servicio	Propósito
Agente de Simulación		
Agente de Comunicación Global	Enviar información	Cuando se ejecuta una acción en el simulador, puede ser necesario actualizar la información que ven los estudiantes en el EV. Para ello, el agente de Simulación utiliza este servicio, que se encarga de enviar la actualización a todos ellos.
Agente Experto		
Agente de Simulación	Ejecutar acción	Cuando el Agente Experto ejecuta una acción o una postcondición, puede encontrar que alguna de las acciones a realizar no la puede llevar a cabo directamente, sino que implica la actualización de algún aspecto de la simulación que debe llevarse a cabo a través de este servicio del Agente de Simulación.
Agente Mundo	Validar precondición	Cuando se le solicita al Agente Experto que valide las precondiciones de una acción, alguna de ellas puede estar relacionada con el estado o la posición de algún elemento dentro del mundo. Si ese ese el caso, le solicita al Agente Mundo que realice la validación a través de este servicio.
Agente de Comunicación Global	Enviar información	Cuando se ejecuta una acción como consecuencia de una acción del estudiante, puede suceder que tenga efectos visibles en uno o varios EVs de los estudiantes, por lo que se utiliza este servicio para notificar los efectos de la acción a todos los estudiantes.
Agente Mundo		
Agente de Comunicación	Solicitar información	La principal labor del Agente Mundo es mantener actualizada la información que maneja el STI respecto al estado del mundo. Para ello, solicita a todos los agentes de comunicación que le envíen la información que reciban de sus respecti- vos EVs, realizando las suscripciones ne- cesarias para recibir toda la información que sea de interés, como posición de estudiantes y objetos.
Agente de Simulación	Solicitar información	Cuando se realiza una acción dentro del simulador, puede ocurrir que se cambie el estado o la ubicación de un objeto del mundo. Para mantener información actualizada del mundo, el Agente Mundo utiliza este servicio para suscribirse a todos los cambios que tengan lugar en la simulación y que puedan afectar a los objetos de los que mantiene información.
Agente de Trayectoria		

Agente de Comunicación	Solicitar información	Para poder realizar el seguimiento de un alumno, el Agente de Trayectoria necesita que se le envíen las coordenadas de sus movimientos por el EV, lo cual hace solicitando dicha información al Agente de Comunicación correspondiente a través de este servicio.
Agente de Estudiante		
Agente de Tutoría	Tratar Acción	El Agente de Estudiante se suscribe a este servicio para recibir información acerca de las acciones realizadas por un estudiante y así poder incorporarlas a su modelo sobre dicho estudiante.
	Tratar actividad	El Agente de Estudiante se suscribe a este servicio para recibir información acerca de las actividades en la que se entrena un estudiante y poder incorporarlas a su modelo sobre dicho estudiante.
	Tratar movimiento	El Agente de Estudiante se suscribe a este servicio para recibir información acerca de las trayectorias seguidas por un estudiante y poder incorporarlas a su modelo sobre dicho estudiante.
	Tratar pregunta	El Agente de Estudiante se suscribe a este servicio para recibir información acerca de las preguntas realizadas por un estudiante y poder incorporarlas a su modelo sobre dicho estudiante.
Agente de Tutoría		
Agente de Comunicación Global	Notificar conexión	El Agente de Tutoría utiliza este servicio para controlar los estudiantes conec- tados a una sesión de entrenamiento, de manera que pueda saber, por ejemplo, si hay suficientes estudiantes para una actividad. También necesita saber si se ha conectado un nuevo estudiante para poder enviarle información acerca de las actividades disponibles, roles dentro de esa actividad y cualquier otra información que considere necesaria.
	Notificar desconexión	Al igual que en el caso anterior, el Agente de Tutoría necesitaconocer si un estudiante se ha desconectado para poder gestionar en consecuencia la actividad en curso o en preparación.
Agente de Comunicación	Solicitar información	El Agente de Tutoría utiliza este servicio para recibir información relativa a las acciones que realiza un estudiante en el EV, sus movimientos, preguntas, etc.
	Enviar información	El Agente de Tutoría utiliza este servicio para dar información al estudiante, como respuestas a sus preguntas, pistas u otro tipo de instrucciones.

Agente Experto	Validar acción	Cuando un alumno realiza una acción, el Agente de Tutoría debe comprobar que esa acción puede llevarse a cabo. Para ello, una de las acciones que debe realizar es comprobar que se cumplen las precondiciones de la mencionada acción, lo que hace a través de este servicio.
	Validar precondición	Cuando un alumno realiza una acción, el Agente de Tutoría debe comprobar que esa acción puede llevarse a cabo. Para ello, una de las acciones que debe realizar es comprobar que se cumplen las precondiciones de la mencionada acción, lo que puede hacer a través de este servicio. La validación precondición a precondición puede hacerse, por ejemplo, para dar información más detallada al alumno acerca de por qué no puede realizar una acción.
	Ejecutar acción	La ejecución de una acción por parte de un alumno puede traer como consecuencia la necesidad de actualizar el estado de la actividad, lo cual se solicita a través de este servicio.
	Ejecutar postcondición	La ejecución de una acción por parte de un alumno puede traer como consecuen- cia la ejecución de otras acciones que ac- tualicen el estado de la actividad, lo cual se solicita a través de este servicio.
Agente de Planificación	Dar acción siguiente	Se usa este servicio para obtener la si- guiente acción del plan y poder compa- rarla con la que ha realizado el estudiante y poder así evaluar su corrección.
	Aplicar acción	Se utiliza este servicio para actualizar el plan con la acción realizada por el estudiante.
	Planificar	Se utiliza este servicio para solicitar que se realice un plan para la actividad que se desea entrenar.
	Replanificar	Se usa este servicio para pedir la ela- boración de un plan alternativo cuando, debido a las acciones realizadas por el alumno, no es posible continuar llevando a cabo el plan actual.
Agente de Trayectoria	Iniciar seguimiento	Se usa este servicio para indicar cuándo se desea comenzar a realizar un seguimiento de los movimientos del estudiante por el EV.
	Finalizar seguimiento	Se utiliza este servicio para indicar cuándo se desea terminar de hacer el seguimiento de los movimientos del alumno por el EV.
Agente de Estudiante	Dar información de estudiante	Dado que uno de los propósitos de un STI es personalizar la tutoría utilizando la información que se posee acerca del estudiante, el Agente de Tutoría utiliza este servicio para consultar información sobre el estudiante y tomar decisiones de tutoría en función de esa información.

Tabla 5.31: Servicios y relaciones entre agentes.

B.2. Relaciones y sus propiedades

La relación existente entre los elementos de esta vista es de uso de las páginas amarillas como medio para anunciar los propios servicios y para buscar los servicios proporcionados por los demás agentes.

B.3. Interfaces

Existen varias interfaces a tener en cuenta en esta vista, aunque no todas serán desarrolladas para esta primera etapa del prototipo.

Como muchos de estos servicios dependen de implementaciones concretas del STI y no son arquitectónicamente relevantes, se ha decidido que los nombres de estos servicios se definan en una etapa más avanzada del diseño. También es recomendable que estos nombres no hagan referencia al agente que los proporciona ni al agente que los utiliza, ya que siéstos variasen también habría que modificar el nombre de los servicios, con el coste asociado que ello pudiera implicar. En cualquier caso, el nombre del agente que facilita los servicios lo proporcionan las páginas amarillas, y el nombre del agente que los utiliza es lo que se quiere evitar conocer.

Por otro lado, hay que considerar la comunicación entre los agentes, que se va a realizar utilizando paso de mensajes que siguen el estándar FIPA-ACL. El contenido de los mensajes dependerá de cada tipo de mensaje concreto, por lo que se pospone su elaboración a una fase más avanzada del diseño, en la que se deberá especificar tanto los distintos mensajes que se pueden enviar como su contenido.

Finalmente, también debe tenerse en cuenta que tanto el Agente de Comunicación Global como los Agentes de Comunicación, encargados de comunicarse con el Centro de Mensajes, deben hacerlo manejando los mensajes definidos en el paquete anterior para comunicar los distintos sistemas entre sí.

B.4. Comportamientos

Cada agente va a buscar en las páginas amarillas los servicios que necesite de otros agentes, y esta búsqueda va a realizarse cuando el agente se inicie.

Algunos agentes, como el de Tutoría y el de Comunicación Global, no dependen ni de la existencia de estudiantes ni del problema concreto a resolver, por lo que deberían iniciarse al arrancar el STI. Otros agentes, como los de Estudiante o los de Comunicación, dependen de la existencia de estudiantes, por lo que deberían iniciarse cuando se conecte un nuevo estudiante. Finalmente, otros agentes, como el de Planificación o el de Trayectoria, dependen de la actividad concreta que se vaya a resolver, por lo que deberían iniciarse una vez la actividad haya sido elegida y se

vaya a comenzar su entrenamiento. Todo esto debe ser tenido en cuenta a la hora de buscar servicios en las páginas amarillas.

De esta manera, al iniciarse un agente lo primero que debe hacer es anunciar sus servicios, para que estén disponibles para todos los demás, y acto seguido comenzar a buscar qué agentes pueden prestarle los servicios que necesita. Según vaya encontrando a estos agentes, deberá ir suscribiéndose a los servicios que necesiten suscripción, y anotando a qué agentes solicitar el resto de servicios. Aunque esta anotación no es estrictamente necesaria, evita tener que buscar a los agentes en las páginas amarillas cada vez que se necesiten sus servicios.

C. Diagrama de contexto

Ver la presentación de la vista de descomposición del STI (sección 5.5.1.5).

D. Guía de variabilidad

En esta parte del prototipo es donde se hace necesaria una mayor flexibilidad, ya que parte de los escenarios de modificabilidad hacen referencia a elementos presentes en esta vista. En concreto, se menciona la posibilidad de deshabilitar funcionalidades tales como el seguimiento de las trayectorias o la propia tutoría.

Puesto que estamos trabajando en un prototipo cerrado, no debería darse la Situación en que un agente no encuentre un servicio que necesita para funcionar. Aún así, debe considerarse la posibilidad de que esto suceda para algunos servicios. Para ello, habría que dotar a los agentes de comportamientos por defecto en caso de que no se encuentren determinados servicios (por ejemplo, el seguimiento de trayectorias, asumiendo que siempre se sigue la mejor trayectoria), aunque si fallan otros, como puede ser la planificación, no debería poder continuar la ejecución.

De esta manera, además de proteger el prototipo frente a errores, se consigue que pueda simplificarse la inhabilitación de algunas funcionalidades. Por ejemplo, eliminar el seguimiento de trayectorias consistiría en no iniciar al Agente de Trayectoria. No es necesario que el Agente de Tutoría decida si necesitasus servicios o no, sino que lo buscará siempre en las páginas amarillas. Cuando no lo encuentre, supondrá que el estudiante realiza bien los desplazamientos por el EV.

E. Información sobre la arquitectura

E.1. Decisiones de diseño

En el apartado B1 de esta vista se han mostrado dos tablas: la primera de ellas contiene los servicios que proporciona cada uno de los agentes del STI, mientras que la segunda contiene los servicios que cada agente usa de entre los que ofrecen los demás.

Aunque el número de servicios que existen no es muy elevado, se ha decidido que, de todos los descritos, sólo se van a anunciar en las páginas amarillas los servicios que son utilizados directamente por otros agentes, que son los que aparecen en la segunda de las tablas mencionadas.

De todos los servicios enumerados, puede considerarse que pertenecen a tres tipos:

El primer tipo son los servicios que proporcionan información de manera periódica y sin control por parte de los agentes, como pueden ser los servicios que proporcionan información sobre las acciones que realizan los estudiantes. Estos servicios requieren suscripción por parte de los otros agentes, y deben ser anunciados en las páginas amarillas.

El segundo tipo son los servicios que se ejecutan bajo demanda y con control directo por parte del agente que los solicita, como la petición de validación de una precondición. Estos también deben ser anunciados en las páginas amarillas.

El tercer tipo son los servicios que se ejecutan como resultado de recibir un mensaje producto de una suscripción. Estos servicios, que se invocan de forma indirecta por parte del agente que envía la información, no neceSTIan ser anunciados en las páginas amarillas, ya que nunca se utilizan de manera directa. Son servicios como el tratamiento de las coordenadas de los desplazamientos del estudiante por el EV.

Análisis de resultados

Todos los resultados se pueden consultar en la sección 5.6

E.3. Suposiciones

En todo momento se ha supuesto que se va a trabajar con una plataforma de desarrollo de sistemas multiagente que proporciona un servicio de páginas amarillas en el que cada agente puede anunciar sus servicios y buscar los servicios publicados por otros agentes.

También se ha supuesto que la plataforma permitirá que los agentes se comuniquen mediante paso de mensajes y que estos mensajes seguirán el estándar FIPA-ACL, según se ha mencionado en el apartado de interfaces.Se ha tomado como prototipo la plataforma JADE.

5.5.4. Vista Peer-to-Peer

La vista peer-to-peer resulta de utilidad para mostrar el comportamiento del prototipo en ejecución. Aunque existen otro tipo de vistas que también sirven a este fin, los sistemas basados en agentes

son esencial y originalmente de tipo peer-to-peer, por lo que se ha considerado más adecuado utilizar esta vista para documentar el comportamiento del prototipo. Por otra parte, si bien es cierto que existen otro tipo de organizaciones para sistemas basados en agentes,éstas se suelen utilizar cuando el número de agentes es elevado, cuando existen protocolos de interacción complejos o cuando existe una clara diferenciación en las funciones de determinados grupos de agentes. Puesto que el prototipo actual no cumple ninguna de estas características, se ha optado por mantener la decisión de utilizar una vista peer-to-peer para documentar los principales flujos de actividad del prototipo.

Debido a que una de las características deseadas del sistema es la posibilidad de modificar con facilidad determinados elementos del mismo, como las estrategias de tutoría o la manera de realizar el modelado del estudiante, en los paquetes que se describen a continuación se ha introducido cierta dosis de ambigu¨edad que permita adaptar el flujo de acciones a las características propias de los elementos para los cuales se desea obtener dicha modificabilidad. Por lo tanto, los aspectos que puedan resultar más ambiguos deberán concretarse en una posterior fase de diseño donde se tomen ya en consideración los detalles concretos de, por ejemplo, una estrategia de tutoría específica.

Para poder reflejar el funcionamiento deseado en cada uno de los paquetes, se ha decidido simplificar los diagramas para mantener su complejidad en un nivel aceptable. De esta manera, sólo se muestra el intercambio de mensajes entre agentes, aunque cada una de las acciones las debe llevar a cabo el correspondiente comportamiento descrito en la sección 5.5.1, a través de los servicios descritos en la sección 5.5.3.2. Estos mensajes intercambiados entre los agentes siguen el estándar FIPA-ACL, aunque su formato concreto deberá ser determinado en una fase más avanzada del diseño.

5.5.4.1. Vista Peer-to-Peer

Paquete 1: Ejecución de una Acción del Estudiante

Presentación de la vista

En este paquete se muestra el intercambio de mensajes que tiene lugar entre los agentes del STI cuando un estudiante realiza una acción en su EV y ésta se envía al STI a través del Centro de Mensajes para que sea tratada.

Como se ha mencionado anteriormente, la secuencia concreta de acciones depende de la estrategia de tutoría que se esté utilizando, y puede no estar definida hasta que el prototipo se encuentre en ejecución, como ocurre si esta estrategia de tutoría es leída desde un fichero de configuración. Puesto queéste es uno de los aspectos en los que se desea que el prototipo sea modificable, las correspondientes acciones se muestran para reflejar cómo tendrían lugar en caso de ser necesarias. En los siguientes apartados de este paquete se pueden encontrar más detalles relativos a esta característica.

B. Catálogo de elementos

B.1. Elementos y sus propiedades

Los elementos pertenecientes a esta vista son tanto los agentes y comportamientos que se pueden ver en la vista de descomposición (ver secciones 5.5.1.5 a 5.5.1.14) como los servicios descritos en la vista editor- suscriptor (ver sección 5.5.3.2).

B.2. Relaciones y sus propiedades

La relación existente entre los elementos de esta vista es invoca, por la cual un elemento invoca la ejecución de un servicio de otro elemento a través del envío de un mensaje.

B.3. Interfaces

Los servicios requeridos y proporcionados por cada elemento pueden verse en el paquete 2 de la vista editor-suscriptor (ver sección 5.5.3.2).

B.4. Comportamientos

La acción comienza cuando un Agente de Comunicación, que es el encargado de comunicarse con un estudiante concreto, recibe a través del Centro de Mensajes una acción realizada por el citado estudiante (mensaje 1). En ese momento envía la acción a los agentes que estén interesados en las acciones del estudiante, que se habrán suscrito previamente a la recepción de las mismas. En este caso concreto son el Agente de Estudiante (mensaje 2), quien mantiene la traza de acciones realizadas por el alumno, y el Agente de Tutoría (mensaje 3), que es el encargado de comprobar su adecuación a la actividad en curso.

Una vez el Agente de Tutoría ha recibido la acción del estudiante, debe comprobar que se dan las condiciones para que el alumno pueda realizar la acción (por ejemplo, que posea la llave que abre una determinada puerta).

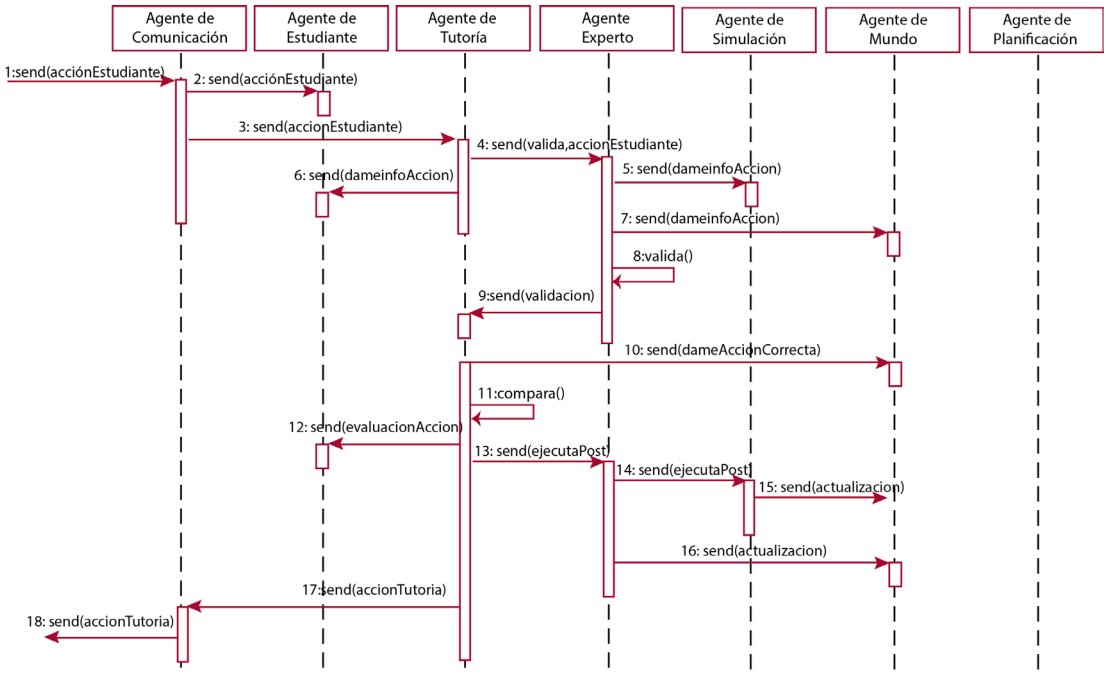


Figura 5.66: Tratamiento de una acción del estudiante

Para ello, le solicita al Agente Experto (mensaje 4) que compruebe si se cumplen todas las precondiciones asociadas a la acción.

Para realizar esta comprobación, el Agente Experto consulta los datos pertinentes a todas las fuentes que pueden tener información relevante para validar las precondiciones. De esta forma, se le puede solicitar información al Agente de Simulación (mensaje 5) acerca del estado de misma (por ejemplo, si una determinada máquina está en marcha) y al Agente Mundo (mensaje 7) relativa a la Situación de algún elemento dentro del EV (por ejemplo, si un objeto está cerca

Con esta información, el Agente Experto valida las precondiciones de la acción (mensaje 8) y le envía el resultado al Agente de Tutoría (mensaje 9).

Mientras el Agente Experto realiza la validación de la acción, el Agente de Tutoría puede ir solicitándole al Agente de Estudiante (mensaje 6) la información relativa aéste que puede hacerle falta para evaluar la acción. Esta información dependerá tanto de las necesidades de la estrategia de tutoría como de las capacidades que posea el Agente de Estudiante de modelar las características del estudiante, sus acciones, etc.

Una vez se ha comprobado que la acción es válida, es decir, que se puede llevar a cabo, también se debe comprobar que es correcta, para lo cual hay que compararla con la que, según el plan realizado por el Agente de Planificación, sería la siguiente acción a realizar (mensajes 10 y 11).

Una vez se conoce la corrección de la acción, y utilizando la información del estudiante de la que se dispone, se puede evaluar la acción y enviarle esta evaluación al Agente de Estudiante (mensaje 12) para que la almacene y la utilice para completar el modelo del estudiante.

Tras comprobar que la ejecución de la acción es posible, es necesario actualizar la actividad con los cambios que produce la ejecución de la acción. Para ello, el Agente de Tutoría le solicita al Agente Experto (mensaje 13) que se encargue de ejecutar las postcondiciones de la acción, lo que implica que se actualice el estado de la simulación (mensaje 14) y el del mundo en función deésta (mensaje 15) y que además se actualice la información del mundo con datos que no dependen de la simulación (mensaje 16). Aunque no aparece en el diagrama, también podría ser necesario ejecutar alguna acción en el EV de alguno de los estudiantes, lo que se haría a través de un mensaje del Agente Experto al Agente de Comunicación Global.

Finalmente, también puede ser necesario proporcionarle al estudiante algún tipo de realimentación acerca de su acción, lo cual se realiza a través de un mensaje del Agente de Tutoría al Agente de Comunicación del estudiante (mensaje 17) que se le envía a través del Centro de Mensajes (mensaje 18).

C. Diagrama de contexto

Ver la presentación de la vista de descomposición del STI (sección 5.5.1.5).

D. Guía de variabilidad

Las variaciones sobre lo expuesto pueden llegar a ser bastante notables. Para que la estrategia de tutoría sea lo más modificable posible, una de las opciones más atractivas, aunque también más complejas de implementar, consiste en que se lea de un fichero de configuración, donde se ajustarían distintos parámetros relativos a la misma, como si se permite realizar acciones erróneas, si se le dan pistas al alumno y cuándo, si se le responden preguntas, si se dan distintas respuestas en función de sus conocimientos y aptitudes, y tantos otros aspectos como se quieran considerar. De esta manera, se puede variar la estrategia de tutoría sin necesidad de modificar el diseño o la implementación de la aplicación.

Por otra parte, también puede hacerse que la estrategia de tutoría se apoye en mayor o menor grado en el conocimiento que se tiene sobre el alumno, y a su vez, este conocimiento puede variar enormemente su contenido y su grado de detalle. Por tanto, lo que se ha modelado como un único mensaje para solicitar información sobre el estudiante puede transformarse en un intercambio de mensajes más numeroso. Lo mismo sucede con el intercambio de mensajes para validar las precondiciones de las acciones realizadas por el estudiante.

Lo que sí queda patente es que la sofisticación de la tutoría - y por tanto las variaciones sobre el modelo presentado - dependerá en gran medida de las capacidades de los agentes que dan apoyo al Agente de Tutoría.

E. Información sobre la arquitectura

E.1. Decisiones de diseño

Siguiendo el curso de acciones de tratamiento de una actividad, a lo largo del diseño de la arquitectura se han tenido en cuenta otras posibilidades, que se describen a continuación.

En primer lugar, hay que tener en cuenta que la estrategia de tutoría puede no permitir la realización de acciones que no sean válidas. En ese caso, todas las acciones que realice un alumno pueden ser correctas o incorrectas, pero siempre serán válidas. En este caso, no sería necesario comprobar la validez de la acción, pues ya se sabría de antemano que la acción sí es válida.

Para realizar la validación de las acciones, inicialmente se pensó que podría ser suficiente con preguntar al planificador, ya que es labor del planificador validar precondiciones para elaborar el plan. Sin embargo, pronto se hizo patente la necesidad de realizar la validación por otros medios, pues el planificador muchas veces carece de la información necesaria para efectuar la validación, y en otras ocasiones no es capaz de operar con la información mínima en la validación.

Una vez validada la acción, se le solicita al Agente de Planificación que nos diga si la acción es correcta o no. Si la acción no era válida tampoco será correcta, pero si era válida puede darse el caso de que no sea correcta o que no sea relevante. El hecho de que se permita la ejecución de acciones incorrectas o irrelevantes depende de que la estrategia de tutoría sea más o menos restrictiva, pero lo que no debería permitir en ningún caso es la realización de acciones no válidas, pues el estado del mundo podría quedar inconsistente. En caso de que se permita la realización de acciones incorrectas, será de utilidad que el planificador sea capaz de devolvernos a la Situación anterior por si se desea reconducir las acciones del alumno. También resulta de utilidad que sea capaz de realizar replanificaciones, para tratar de resolver la actividad en curso por un camino que, sin ser el mejor desde el punto de partida, es el que ha decidido seguir el estudiante, ya sea de forma consciente o no. Todas estas características permitirán que la tutoría se adapte en mayor medida a las acciones del alumno.

Aunque en el diagrama anterior se han mostrado todas las acciones de manera secuencial, algunas de ellas pueden realizarse en paralelo, ya que no interfieren unas con otras y, si la plataforma utilizada lo permite, la ejecución de los agentes se realiza en paralelo. Así, la información que se solicita en los mensajes 5, 6 y 7 se puede pedir a la vez, sin necesidad de esperar a recibir

una para solicitar la siguiente. Sin embargo, si habrá que disponer de la información de los mensajes 5 y 7 para poder realizar la acción 8. De igual manera, la acción que desencadena el mensaje 6 puede realizarse en paralelo con las acciones 5 a 11, pero será necesario disponer de la información para poder realizar la evaluación de la acción y enviar el mensaje 12. Se sugiere que la acción 6 se realice en ese momento porque la obtención de la información del estudiante puede requerir un procesamiento que tome algo de tiempo, lo que, si se realiza en paralelo con las otras acciones, repercutirá en una mayor velocidad de ejecución.

Finalmente, debe tenerse en cuenta el hecho de que, dado que la validación de precondiciones y la ejecución de postcondiciones se hacen de manera separada a la planificación, será necesario definir, por un lado, los dominios y operadores que maneje el planificador, y por otro, los operadores que vaya a manejar el Agente Experto, junto con las precondiciones y postcondiciones correctamente especificadas para que este agente sepa a quién debe pedirle la información para hacer la validación.

E.2. Análisis de resultados

Ver sección 5.6

E.3. Suposiciones

En todo momento se ha supuesto que los agentes tienen a su disposición los servicios que necesitan para llevar a cabo sus responsabilidades. Se ha hecho esta suposición porque, como ya se ha mencionado anteriormente, se trabaja en un entorno cerrado en el que todos los agentes se encuentran bajo nuestro control. De esta forma, siempre podemos asegurarnos, antes de poner en marcha el prototipo, de que contamos con todos los agentes necesarios y queéstos prestan todos los servicios que se requieren de ellos.

Aún así, es necesario tener en cuenta que, para que se puedan añadir y quitar agentes en función de las necesidades de la actividad que se entrena, se debe diseñar un mecanismo para que, o bien no se busquen los servicios que no se van a neceSTIar, o bien se busquen todos los servicios conocidos pero se pueda iniciar la actividad si no se encuentran los servicios que luego no se van a utilizar.

F. Otra información

El hecho de que se necesite que el Agente Mundo realice ciertas operaciones ha influido notablemente en la decisión de que la información con la que trabaja se almacene en una ontología en lugar de en una base de datos. La ontología permite la definición de relaciones entre elementos y la utilización de razonadores que trabajen sobre ella, lo que supone una herramienta muy potente y elimina la necesidad de desarrollar esas funcionalidades.

Lo mismo sucede en el caso del Agente de Estudiante, que en esta propuesta almacena la información del modelo del estudiante en una ontología sobre la que poder realizar razonamientos y deducciones.

Por otra parte, dado que varios de los agentes manejan información relativa a las acciones de los estudiantes, se puede inferir que el prototipo es sensible al formato que adopte esta representación. Por este motivo, de cara a poder sustituir el planificador utilizado, se debe intentar que la representación de las acciones sea independiente de la representación de los operadores que maneja el planificador, y deberá realizarse la traducción pertinente con algún elemento situado entre el Agente de Planificación y el planificador. Resulta probable que el Agente de Planificación se vea afectado por un cambio de planificador, pero el hecho de utilizar un envoltorio (lo que en la vista de descomposición se ha denominado Wrapper) posibilita que el impacto de este cambio pueda verse atenuado incluso en el propio agente. De esta manera, con cambios menores podría sería posible sustituir el planificador solamente diseñando un nuevo envoltorio que realice la comunicación con el planificador nuevo.

5.5.4.2. Vista Peer-to-Peer.
Seguimiento de Movimientos del Estudiante / Presentación de la vista

En este paquete se muestra el intercambio de mensajes entre los agentes cuando tiene lugar un desplazamiento del estudiante por el EV. De manera similar a lo que sucedía en el paquete anterior, el intercambio de mensajes está sujeto a lo que estipule la estrategia de tutoría que se esté utilizando, por lo que, igual que en el caso anterior, lo aquí expuesto está sujeto a variaciones en función de la estrategia de tutoría.

B. Catálogo de elementos

B.1. Elementos y sus propiedades

Los elementos pertenecientes a esta vista son tanto los agentes y comportamientos que se pueden ver en la vista de descomposición (ver secciones 5.5.1.5 a 5.5.1.14) como los servicios descritos en la vista editor- suscriptor (ver sección 5.5.3.2).

B.2. Relaciones y sus propiedades

La relación existente entre los elementos de esta vista es invoca, por la cual un elemento invoca la ejecución de un servicio de otro elemento a través del envío de un mensaje.

B.3. Interfaces

Los servicios requeridos y proporcionados por cada elemento pueden verse en el paquete 2 de la vista editor-suscriptor (ver sección 5.5.3.2).

B.4. Comportamientos

La ejecución comienza cuando el Agente de Tutoría indica al de Trayectoria que comience la realización del seguimiento de los movimientos del estudiante (mensaje 1) y éste calcula las trayectorias más adecuadas.

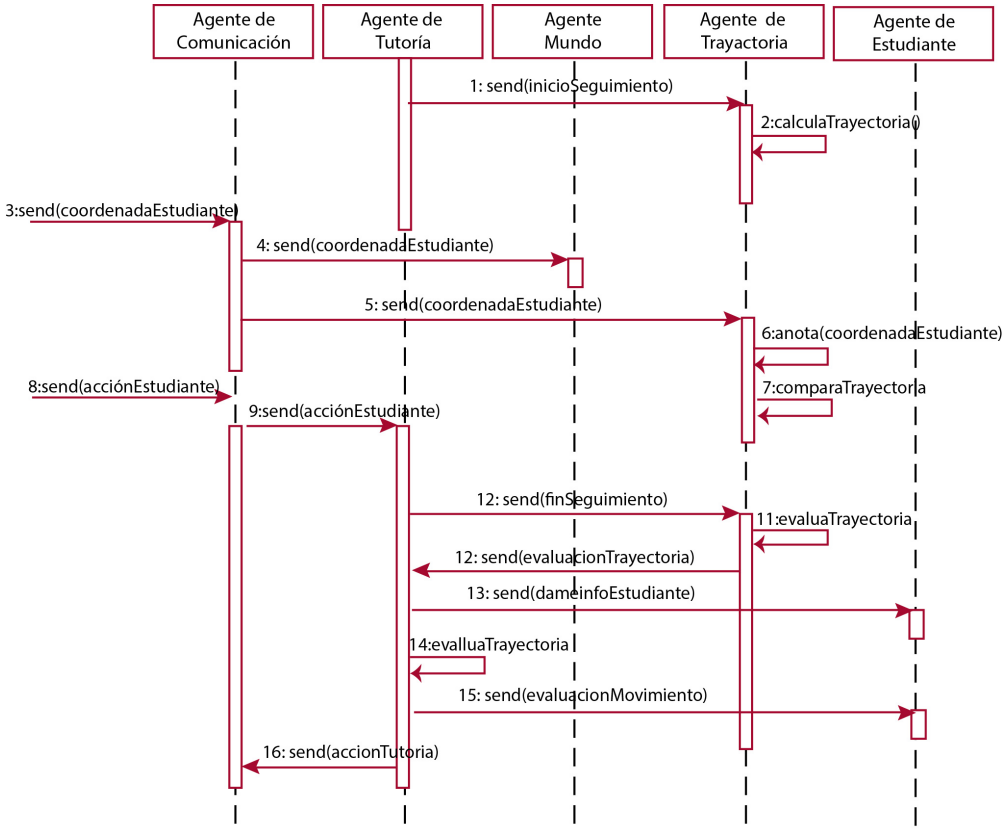


Figura 5.67: Tratamiento de un movimiento del estudiante

Posteriormente, el Agente de Comunicación, que es el encargado de comunicarse con un estudiante concreto, comienza a recibir mensajes desde el Centro de Mensajes (mensaje 3) con los que se le indican las nuevas coordenadas de Situación del estudiante dentro de su EV. Este agente reenvía las coordenadas a los agentes que se hayan suscrito a su recepción, en este caso al Agente Mundo (mensaje 4), que mantiene actualizada la situación del estudiante en su representación del EV, y el Agente de Trayectoria (mensaje 5), que realiza el seguimiento de los movimientos. Para ello, registra la coordenada (mensaje 6) y va construyendo una trayectoria que compara con la trayectoria calculada inicialmente (mensaje 7).

Este proceso se repite hasta que el estudiante realiza una acción (mensaje 8). Cuando el Agente de Tutoría recibe notificación de este suceso (mensaje 9), le indica al de Trayectoria que finalice el seguimiento del tramo de trayectoria actual (mensaje 10). Es entonces cuando el Agente de Trayectoria calcula cuánto se ajusta la trayectoria del estudiante a la ideal calculada por él (mensaje 11) y le envía el resultado al Agente de Tutoría (mensaje 12).

Al recibir esta evaluación es cuando, junto con la información que recibe sobre el estudiante (mensaje 13), realiza su propia evaluación de la trayectoria seguida por el estudiante (mensaje 14) y le envía dicha evaluación al Agente de Estudiante (mensaje 15) para que la utilice para completar el modelo que realiza sobre el estudiante.

En caso necesario, se envía un mensaje al EV del estudiante (mensaje 16) con las acciones de tutoría necesarias para indicarle al estudiante la idoneidad de sus movimientos.

C. Diagrama de contexto

Ver la presentación de la vista de descomposición del STI (sección 5.5.1.5).

D. Guía de variabilidad

Una de las variaciones que se contempla sobre el esquema propuesto es que no se realice ningún seguimiento de los movimientos del estudiante, bien porque la estrategia de tutoría no lo considere como un factor a tener en cuenta, bien porque la actividad objeto de entrenamiento no implique desplazamientos por el EV.

Otra de las posibles variaciones consiste en que el Agente de Trayectoria, a medida que va comparando la trayectoria del estudiante con la calculada, le envíe pistas si detecta que se desvía en exceso del camino que debería seguir.

E. Información sobre la arquitectura

E.1. Decisiones de diseño

Aunque se ha considerado la posibilidad de enviar desde el EV mensajes que indiquen el comienzo y la finalización de un desplazamiento, lo que facilitaría la labor del Agente de Tutoría y del de Trayectoria, se ha descartado principalmente por dos motivos.El primero es que hay dispositivos de interacción que funcionan directamente transmitiendo coordenadas, como los sensores de posición, lo cual dificulta identificar cuándo se inicia y cuándo finaliza un movimiento. El segundo motivo es que, aunque podría realizarse en el EV un análisis de las coordenadas que indique cuando comienza y finaliza el desplazamiento, esto conlleva trasladar al EV una lógica que no le

corresponde. Puesto que uno de los objetivos que se persiguen es poder intercambiar los EVs utilizados, hay que pensar que este tipo de acción no la realizan habitualmente los Entornos Virtuales, por lo que habría que realizar modificaciones en los mismos siempre que se desee sustituir uno por otro.

Dependiendo del número de estudiantes y de los algoritmos de cálculo y seguimiento de trayectorias que se utilicen, puede suceder que el Agente de Trayectoria sea incapaz de procesar los desplazamientos de todos los estudiantes con la rapidez necesaria. Se postpone, por tanto, la decisión de utilizar tantos Agentes de Trayectoria como estudiantes para poder realizar el procesamiento con mayor velocidad, en caso de que sea necesario.

E.2. Análisis de resultados

Todos los resultados se pueden consultar en la sección 5.6.

E.3. Suposiciones

Se parte de la premisa de que una trayectoria ideal no viene dada únicamente por criterios geométricos, sino que puede haber otros factores que influyan en la adecuación de la misma. Así, por ejemplo, en una central nuclear la trayectoria más corta puede pasar junto a una fuente de radiación, mientras que otra más larga puede mantenernos alejados de dicha fuente, convirtiéndola así en una trayectoria más adecuada. Por tanto, se debe contar con algoritmos de cálculo de trayectorias que permitan incorporar criterios heurísticos adicionales a los meramente geométricos.

5.5.4.3. Vista Peer-to-Peer.

Respuesta a una Pregunta del Estudiante / Presentación de la vista

En este paquete se presenta el funcionamiento del STI cuando un estudiante realiza una pregunta que debe ser respondida por éste. La variedad de preguntas que puede realizar un estudiante es bastante amplia, y las formas de realizar una misma pregunta son también numerosas, por lo que en la presente vista no se considera la problemática de analizar la pregunta para que la entienda el STI, sino que se supone que ya está en un formato manejable por éste.

Por el mismo motivo, muchos de los agentes pueden verse involucrados en la elaboración de las respuestas, por lo que de momento se ha restringido el prototipo a la elaboración de respuestas no excesivamente complejas a preguntas relativamente sencillas, que en general se pueden responder con la información de la que dispone uno sólo de los agentes.

B. Catálogo de elementos

B.1. Elementos y sus propiedades

Los elementos pertenecientes a esta vista son tanto los agentes y comportamientos que se pueden ver en la vista de descomposición (ver secciones 5.5.1.5 a 5.5.1.14) como los servicios descritos en la vista editor- suscriptor (ver sección 5.5.3.2).

B.2. Relaciones y sus propiedades

La relación existente entre los elementos de esta vista es invoca, por la cual un elemento invoca la ejecución de un servicio de otro elemento a través del envío de un mensaje.

B.3. Interfaces

Los servicios requeridos y proporcionados por cada elemento pueden verse en el paquete 2 de la vista editor-suscriptor (ver sección 5.5.3.2).

B.4. Comportamientos

La acción comienza cuando el Agente de Comunicación, encargado de la comunicación con un estudiante determinado, recibe desde el Centro de Mensajes un mensaje con la pregunta realizada por el estudiante (mensaje 1). Este agente le envía la pregunta a los agentes que se hayan suscrito a su recepción, que en este caso son el Agente de Estudiante (mensaje 2) para elaborar el modelo de estudiante, y el Agente de Tutoría (mensaje 3), que es el encargado de responder. Tras analizar el tipo de pregunta realizada, el Agente de Tutoría solicita la información necesaria para elaborar la respuesta al agente apropiado (alternativamente, mensajes 4 a 6). Tras recibir información sobre la respuesta, solicita también al Agente de Estudiante información relativa aéste (mensaje 7) para, por un lado, decidir el nivel de detalle de la respuesta (mensaje 8), y por otro, para evaluar la adecuación de la misma (mensaje 9). Finalmente, se le envía la respuesta al Agente de Comunicación (mensaje 10) para queéste se la envíe al estudiante que la realizó.

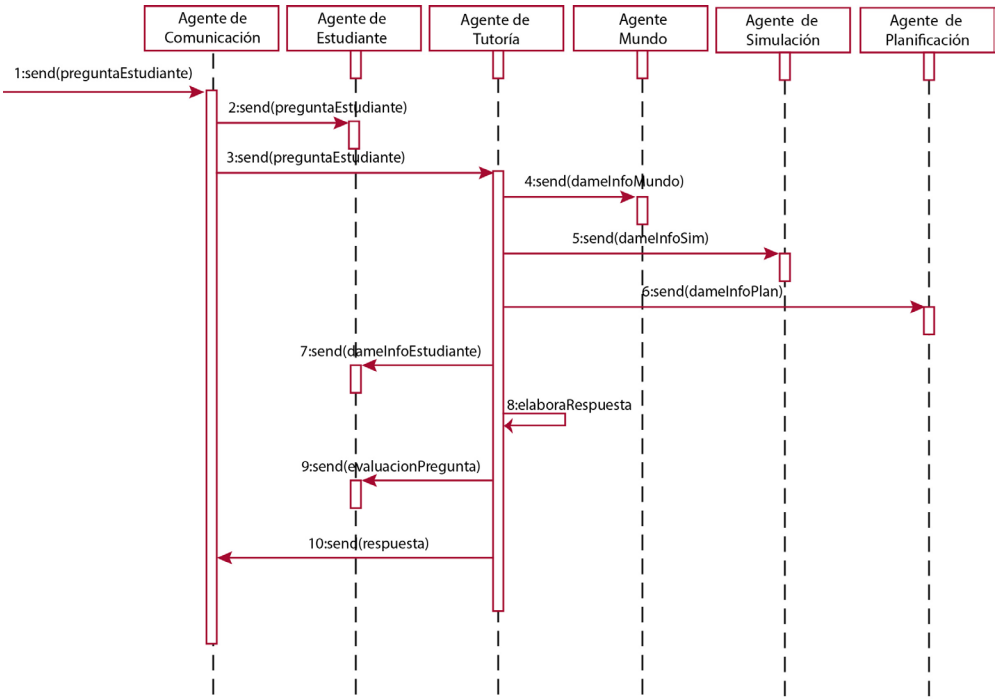


Figura 5.68: Tratamiento de una pregunta del estudiante

C. Diagrama de contexto

Ver la presentación de la vista de descomposición del STI (sección 5.5.1.5).

D. Guía de variabilidad

Para diseñar el mecanismo de respuesta a una pregunta hay que tener en cuenta el grado de complejidad que se quiere incluir tanto en el análisis de las preguntas como en el tipo y la elaboración de las respuestas.

La variación más obvia respecto a lo planteado es la inclusión de nuevos tipos de preguntas simples para las que puede ser necesario solicitar información a otros agentes. Para ello, sería necesario incluir nuevos agentes en el diagrama mostrado, pero el funcionamiento básico sería el mismo.

E. Información sobre la arquitectura

E.1. Decisiones de diseño

La probabilidad de encontrar un EV que permita el envío de preguntas, salvo que se haya diseñado expresamente para ello, es bastante baja. Esto, que a priori puede parecer un inconveniente, puede acabar convirtiéndose en una ventaja, ya que posibilita que pueda disponerse de una

aplicación en el cliente, construida más a medida del STI, que se ejecute junto con el EV pero que funcione de manera separada y, sobre todo, que procese las preguntas y las envíe al STI en un formato fácilmente manejable poréste.

La preguntas que se ha pensado resolver, en principio, son relativas a los objetos del entorno (qué son o para qué sirven), cuya respuesta se puede almacenar como una descripción relativa a los objetos en el mismo soporte, fichero o base de datos, en el que se guarde otra información relativa a ellos, y que puede ser manejada y proporcionada por el Agente Mundo.

También se pueden responder preguntas relativas a las acciones a realizar (por ejemplo, qué hay que hacer o qué se debería haber hecho), cuya respuesta la puede proporcionar el Agente de Planificación.

Otros posibles tipos de preguntas a considerar son por qué no se puede hacer algo, cuya respuesta la proporcionaría el Agente Experto a partir de las precondiciones de la acción que se quiere realizar, o por dónde llegar a algún punto del EV, que sería respondida a partir de información proporcionada por el Agente de Trayectoria, posiblemente completada con información proporcionada por el Agente Mundo, ya que el primero sólo maneja información relativa a coordenadas, y esto no es especialmente informativo a la hora de construir una respuesta para un estudiante.

E.2. Análisis de resultados

Ver sección 5.6

E.3. Suposiciones

Se ha supuesto que el estudiante realiza sus preguntas utilizando un reconocedor de voz o una aplicación de texto similar a un chat. No se han contemplado otras posibilidades ni, por tanto, sus posibles implicaciones en la arquitectura del prototipo.

5.5.5. Vista de Despliegue

Dentro del conjunto de vistas de asignación, la vista de despliegue permite especificar en qué elementos hardware se van a ejecutar los elementos software que se han descrito en las vistas anteriores.

Como se verá a continuación, algunos de los elementos hardware pueden variar significativamente en función del dominio de entrenamiento, pero en líneas generales no se requiere un hardware con unas exigencias mayores que las prestaciones que puede ofrecer un PC de gama alta.

5.5.5.1. Vista de Despliegue

Despliegue de la Aplicación / Presentación de la vista

Como se puede ver en la imagen que acompaña este apartado, la idea general es que cada uno de los subsistemas que conforman la aplicación de entrenamiento, Sistema Inteligente de Tutoría, Centro de Mensajes, Simulador y Entorno Virtual, se ejecuten en máquinas distintas, y que cada estudiante que se conecte a la aplicación a través de un EV lo haga desde una máquina distinta a la de otros estudiantes.

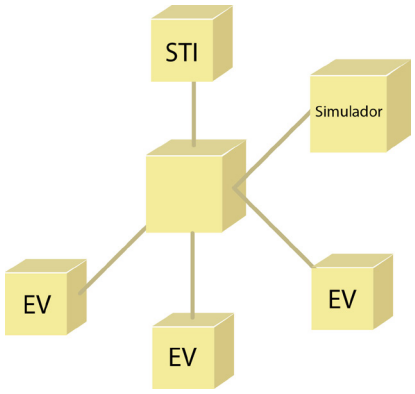


Figura 5.69: Diagrama de despliegue de la aplicación

B. Catálogo de elementos

B.1. Elementos y sus propiedades

Elementos	Propiedades
Entorno Virtual	Las máquinas donde se ejecuten los EVs, al ser aplicaciones tridimensionales cuyos requisitos a nivel gráfico suelen ser relativamente altos, neceSTIan ser PCs de gama alta provistos de tarjetas gráficas con aceleración 3D, de manera que sean capaces de realizar una ejecución suave, de al menos 30 fps (frames por segundo), para dar una sensación de movimiento fluido durante la navegación a través del EV.
Sistema de Tutoría Inteligente	La máquina donde se ejecute el STI debe ser un PC de gama alta con capacidad de procesamiento múltiple, que pueda dar cabida a la ejecución de los agentes en paralelo y que soporte en momentos determinados la ejecución de procesos con un consumo alto de procesador y memoria, como puede ser el cálculo de trayectorias o, en mayor medida, el proceso de planificación.

Centro de Mensajes	En principio, el Centro de Mensajes es una aplicación que consume pocos recursos y que no realiza un procesamiento exigente, por lo que será suficiente con que se ejecute en un PC de gama baja con una conexión a la red de velocidad media (100Mbps) que le permita redirigir todos los mensajes que reciba sin retardos apreciables.
Simulador	Las características de la máquina donde se ejecute el simulador deben ser definidas según los requisitos de cada simulador concreto.

Tabla 5.32:Elementos y Despliegue de la Aplicación

B.2. Relaciones y sus propiedades

La relación que dirige esta vista es asignado a, que relaciona los elementos software con los elementos hardware en los que se ejecutan. No existe ninguna variación respecto a lo definido por esta relación.

B.3. Interfaces

Las interfaces entre los distintos nodos, al producirse la comunicación en una red a través del Centro de Mensajes, están constituidas por unas tarjetas de red de velocidad media o alta (≥100Mbps) que realizan la comunicación a través del protocolo TCP/IP.

D. Guía de variabilidad

Existen dos puntos importantes de variabilidad respecto a lo mostrado anteriormente. El primero de ellos lo constituye el simulador, que en vez de ejecutarse en un máquina separada, que puede ser el caso más habitual, puede presentar otras situaciones. La primera de ellas es que se ejecute en una misma máquina junto con el EV, ya que en muchos casos ambas aplicaciones se encuentran implementadas de manera conjunta. También puede darse el caso de un simulador sencillo que se implemente como parte del agente de simulación, en cuyo caso se ejecutaría en la misma máquina que el STI.

Si se ejecuta en una máquina separada, dependiendo de los requisitos de la simulación se pueden dar la Situación de que se ejecute en un PC de gama media o en una máquina de altas prestaciones diseñada exclusivamente para ejecutar esa simulación, como podrían ser cabinas simuladoras de vuelo.

Por otra parte, en función de las características del dominio de entrenamiento, puede aparecer la necesidad de ejecutar el STI en distintas máquinas para dar soporte a unas necesidades de procesamiento exigentes o para poder albergar un número alto de estudiantes que trabajan de manera conjunta. En este caso, plataformas de agentes como JADE proporcionan de serie

mecanismos para distribuir la plataforma y sus agentes en varias máquinas sin modificar la forma en que los agentes están programados.

Lo que no se contempla en ningún caso, por el momento, es la migración dinámica de partes de la aplicación a otras máquinas.

E. Información sobre la arquitectura

E.1. Decisiones de diseño

Dentro del tratamiento de una acción del estudiante, las operaciones que pueden llevar más tiempo de ejecución y más carga de trabajo son las replanificaciones de la actividad, si el alumno realiza muchas acciones incorrectas, la validación de precondiciones en el mundo, si las precondiciones son grandes o complejas, y el seguimiento de las trayectorias, especialmente si el alumno se desvía mucho de las trayectorias ideales.

Puesto que la comunicación de los agentes encargados de estas labores con otros está bastante restringida, son candidatos bastante claros a ejecutarse en una máquina distinta a la del resto de los agentes. Lo mismo sucede con los Agentes de Estudiante, que además pueden proliferar mucho, dependiendo del número de estudiantes involucrados en una actividad. En este último caso la distribución en otras máquinas podría ser algo más compleja, ya que darían lugar a una mayor comunicación entre distintas máquinas.

La distribución de agentes en distintas máquinas también puede resultar problemática por imponer la restricción de que la plataforma utilizada para implementar los agentes debe permitir realizar esta distribución.

E.2. Análisis de resultados

Todos los resultados se pueden consultar en la sección 5.6

E.3. Suposiciones

Para realizar el despliegue de la aplicación se ha supuesto la utilización de una plataforma con características similares a JADE, que permite la ejecución de una plataforma de agentes en varias máquinas de manera que el diseño de los agentes no se vea afectado, sino que es la propia plataforma la que se encarga de gestionar el envío y recepción de mensajes de manera idéntica, se encuentren los agentes en la misma máquina o en máquinas diferentes.

5.6. Análisis de la Arquitectura

El propósito de esta sección es analizar la arquitectura descrita en las secciones anteriores para comprobar que cumple con las condiciones establecidas al comienzo del capítulo con los escenarios de calidad. El objetivo es, pues, ver si lo que se ha diseñado cumple los requisitos de modificabilidad y eficiencia que se desea que tenga el sistema.

Para ello, se analizará la arquitectura escenario a escenario, razonando, para cada uno de ellos, de qué manera cumple el diseño de la arquitectura con las necesidades descritas en ellos.

5.6.1. Escenarios y propiedades de Modificabilidad

La mayoría de los escenarios de modificabilidad que aparecen a continuación están relacionados con la sustitución de un elemento del prototipo por otro que cumple funciones similares, pero de distinta manera. A nivel teórico, siempre es posible realizar esta sustitución, independientemente de cómo esté implementado el nuevo elemento, a través del uso de lo que en (Hayden y otros, 1999) se denomina patrón Wrapper, lo que informalmente se denomina agentificar un elemento para que a los ojos del resto de los agentes aparezca como un agente más.

Las mayores dificultades para realizar las sustituciones van a estar originadas principalmente por dos situaciones. La primera de ellas es la incompatibilidad del nuevo elemento con los existentes, por el hecho de que no proporcione la funcionalidad que el resto requiere. Y la segunda es la posibilidad de que esté implementado de tal manera que no se pueda adaptar al modelo utilizado basado en la prestación de servicios.

5.6.1.1. Sustitución del Entorno Virtual

La arquitectura del prototipo se ha diseñado de forma tal que todos los subsistemas se comunican a través del Centro de Mensajes. Además, cada subsistema se suscribe a la recepción de los tipos de mensajes que desea recibir, pero no en función de quién realiza el envío de los citados mensajes. De esta manera, no existe una comunicación directa entre el STI y el EV, y la que tiene lugar, a través del Centro de Mensajes, se hace de manera anónima, utilizando un mecanismo de paso de mensajes. Por lo tanto, siempre y cuando el EV que se incorpore sea capaz de manejar la comunicación por paso de mensajes a través del Centro de Mensajes, y siempre que esos mensajes tengan el mismo formato y sigan los mismos protocolos que utilizaba el EV que se sustituye, el intercambio de un EV por otro no debería afectar al resto de subsistemas.

Puesto que lo más probable es que el nuevo EV no se haya diseñado para funcionar junto con el STI, lo más habitual será que aparezca la necesidad de modificar el EV para poder incorporarlo al prototipo. En concreto, lo más común es que surja la necesidad de realizar dos tipos de

modificaciones. En primer lugar, hará falta crear un módulo de comunicación con el centro de mensajes que se encargue de realizar la comunicación con el resto de subsistemas que conforman la aplicación de entrenamiento. Aunque la comunicación se realice sólo a través del centro de mensajes, habrá que dotarlo con la capacidad para manejar todos los tipos de mensajes que manejaba el EV anterior.

En segundo lugar, es poco probable que el EV esté preparado para enviar a otras aplicaciones las acciones que tienen lugar dentro del mismo, razón por la cual se deberá modificar para que las acciones que realice el usuario se envíen al nuevo módulo de comunicación, además de tratarlas en el propio EV. También se deberán enviar las posiciones y las orientaciones de todos los elementos móviles que experimenten algún cambio en alguna de ellas.

De hecho, y dependiendo de la estrategia de tutoría utilizada, existe la posibilidad de que sea necesario que las acciones del estudiante cuenten con el permiso del STI para que se puedan ejecutar, por lo que llegado el caso haría falta también integrar este mecanismo de validación previa en la interacción del estudiante con el EV. De no hacerlo así no sería posible hacer uso de este tipo de tutoría.

Esta necesidad puede surgir tanto si se sustituye el EV como si se cambia la estrategia de tutoría, por lo que constituye un punto sensible en cuanto a la modificabilidad del prototipo. Si el EV se diseña con cuidado, el cambio no pasará de afectar a la secuencia de acciones que tienen lugar cuando el estudiante interactuá con el sistema. En caso contrario, probablemente habrá que modificar sustancialmente su diseño.

5.6.1.2. Sustitución de la Estrategia de Tutoría

La estrategia de tutoría, a través del Agente de Tutoría, constituye la pieza fundamental de un sistema cuyo principal objetivo es justamente la tutoría

Así como se ha planteado la arquitectura, ningún agente depende de la presencia del Agente de Tutoría, lo cual, en apariencia, hace pensar que la sustitución de este elemento puede resultar sencilla. Sin embargo, aunque algunos de los mecanismos que se han introducido al diseñar la arquitectura buscan facilitar la modificabilidad de la misma, no se debe caer en la tentación de dar por hecho que cualquier cambio será fácil y rápido.

El STI no deja de ofrecer un conjunto de agentes que giran alrededor del Agente de Tutoría para satisfacer sus necesidades a la hora de ayudar al estudiante durante el entrenamiento. Así, la sustitución de la estrategia de tutoría puede suponer la necesidad de añadir nuevos agentes con

capacidades que presten servicios que no se han considerado hasta el momento.

Como se verá en el siguiente apartado, cualquier cambio de estrategia de tutoría es susceptible de provocar modificaciones, a veces en la sustitución del modelo del estudiante, ya que resulta fundamental para la personalización de la tutoría y, por tanto, existe una dependencia notable entre ambos elementos. Aunque no se produzcan diferencias en la manera en que ambos elementos se comunican, sí pueden producirse variaciones en la información que solicita el Agente de Tutoría y que el modelo del estudiante debe mantener.

En cuanto al resto de los agentes, puede suceder que no se requieran sus servicios, como es el caso del Agente de Trayectoria si la estrategia de tutoría no valora la adecuación de los desplazamientos del estudiante. También puede resultar sensible la relación con el Agente de Planificación. Si no se permite que el estudiante se desvíe del plan establecido, puede darse el caso de que el Agente de Tutoría suponga que se le va a entregar el plan completo de una vez, y así no tener que estar comunicándose continuamente con el Agente de Planificación. En ese caso, la modificación más sencilla será añadir un servicio a ese agente para que proporcione el plan completo, lo que también requerirá añadir un elemento que traduzca el plan a un formato manejable por el Agente de Tutoría.

Como se puede apreciar, la sustitución de la estrategia de tutoría no está exenta de producir ciertas modificaciones en otros elementos, justo por ser uno de los elementos fundamentales del prototipo. Al sustituir la esencia del sistema, lo esperable es que otras partes del mismo se vean afectadas. Sin embargo, el impacto de estos cambios, que son susceptibles de producirse en cualquier situación, puede verse notablemente reducido a través de la utilización de servicios, siempre que el nuevo elemento introducido admita el funcionamiento utilizando este enfoque.

5.6.1.3. Sustitución del Modelo del Estudiante

El modelo del estudiante es uno de los componentes del prototipo que es susceptible de sufrir una mayor evolución. Poco a poco se van incorporando elementos que hacen un modelado más complejo de los estudiantes, ampliando la traza de sus acciones e incorporando elementos cada vez más relacionados con el mundo de la psicología, lo que incluye modelos de personalidad, de atención o de aprendizaje.

La evolución de estos modelos va de la mano a la evolución de la estrategia de tutoría, que se irá adaptando al estudiante de una forma más compleja a medida que el modelo del estudiante vaya realizando un perfil más completo del mismo. Por lo tanto, como se mencionó en el apartado anterior, un cambio en la estrategia de tutoría es susceptible de producir cambios en la información

que se necesita que proporcione el modelo del estudiante.

Las modificaciones en este modelo, y en concreto la sustitución del mismo, pueden generar cambios en la estrategia de tutoría, debido a que el nuevo modelo del estudiante no incorpore elementos que el anterior sí incluía, o incorpore elementos diferentes o bien que sean tratados de otra manera. Independientemente de estas situaciones, una de las principales razones para sustituir un modelo de estudiante por otro es precisamente que se deseen realizar modificaciones en la estrategia de tutoría para incorporar las funcionalidades proporcionadas por el nuevo modelo del estudiante, razón por la cual muchas de las modificaciones a realizar no serán consecuencia del cambio, sino una de sus causas.

En los casos mencionados, las modificaciones son inherentes a la propia estrategia de tutoría o a la funcionalidad proporcionada por el modelo del estudiante. Siempre que la estrategia de tutoría y el modelo de estudiante que se utilizan sean compatibles, la arquitectura permite que cada uno de ellos publique y utilice los servicios que necesite. En caso de que no sean compatibles, el problema que se produce es más a nivel teórico que arquitectónico, por lo que la solución que pueda proporcionarse en lo referente a la arquitectura estará supeditada a la existencia de una solución conceptual aceptable. Lo más importante en este punto es dejar manifiesta la estrecha relación existente entre la estrategia de tutoría y el modelo del estudiante.

5.6.1.4. Sustitución del Planificador

Una de las principales razones por las que se decidió rediseñar la arquitectura descrita se debió a la dificultad para sustituir el planificador utilizado por otro con unas características diferentes, más sofisticadas, desarrollado por otro grupo de investigación.

En la arquitectura descrita en el presente capítulo, el planificador es un elemento controlado por el Agente de Planificación, y ningún otro agente tiene relación con él salvo el Agente de Tutoría, que le envía al de Planificación las acciones realizadas por el estudiante.

La labor del Agente de Planificación consiste en atender las peticiones del Agente de Tutoría y en realizar labores de traducción entre éste y el planificador, de manera que los operadores manejados por los agentes y los EVs no dependan, en la medida de lo posible, del lenguaje utilizado por el planificador para definir el dominio del problema y las acciones que se pueden realizar en el mismo. La posible necesidad de adaptar los operadores manejados por los agentes al planificador utilizado puede afectar a varios de los agentes del prototipo.

Por tanto, una de las principales labores de diseño del prototipo consistirá en cuidar el formato de los operadores que representan las acciones de los estudiantes, de manera que la traducción que debe realizar el Agente de Planificación cuente con todos los elementos necesarios para convertirlos en operadores del planificador. De forma paralela se debe evitar propagar datos innecesarios o inconvenientes al resto de los agentes o, incluso, a los EVs, que son los que, en definitiva, generan los operadores relativos a las acciones de los estudiantes.

5.6.1.5. Sustitución del Planificador de Rutas

El planificador de rutas es un elemento que incluyen algunos de los sistemas que requieren que el alumno o el tutor virtual se desplacen por el mundo virtual. En los casos más sencillos, como sucede con Steve (Rickel y Johnson, 1999), el cálculo de rutas se realiza a través de un grafo que indica qué puntos significativos del mundo son accesibles desde otros. En casos algo más complejos, el uso de los grafos evoluciona hacia el uso de algoritmos como el de Dijkstra (Russell y Norvig, 1995).

En otros casos, como en (Sud y otros, 2007; Sud y otros, 2008), en lugar de trabajar con lugares significativos del mundo para definir nodos de un grafo, se trabaja con diagramas de Voronoi[7] (Aurenhammer, 1991), lo que hace posible que se puedan utilizar algoritmos más precisos e, incluso, trabajar con entornos cambiantes en los que no siempre se puede seguir la misma ruta para recorrer el camino entre dos puntos.

En cualquiera de los dos casos, si el planificador de rutas no recibe directamente del Agente de Comunicación la información que necesita, existe la posibilidad de que el Agente Mundo, que es el encargado de mantener el modelo del EV que manejan los agentes, proporcione la coordenada correspondiente a un lugar significativo o viceversa. Esto implica que, de no ofrecer un servicio similar, el cambio del planificador de rutas puede suponer la necesidad de actualizar el Agente Mundo para que pueda realizar la mencionada traducción entre lugares y coordenadas.

5.6.1.6. Sustitución del Simulador

La sustitución del simulador es una tarea que habrá que realizar de manera regular cada vez que se cambie el dominio en el que se realiza el aprendizaje. Un simulador, en general, es específico de un dominio determinado, incluso de una parte concreta de un dominio específico, razón por la que

7 Los diagramas de Voronoi son una de las estructuras fundamentales dentro de la Geometría Computacional, de alguna forma ellos almacenan toda la información referente a la proximidad entre puntos. Son numerosísimas sus aplicaciones.

puede constituir uno de los elementos que presenten una mayor variabilidad dentro del prototipo. En función de cómo esté implementado y de lo que suceda dentro de él habrá que integrarlo en el prototipo de muy distintas maneras. Por este motivo, dentro del alcance del presente trabajo resulta difícil plantear una solución lo suficientemente abierta como para poder aventurarse a decir que se podrá integrar con el resto de subsistemas que conforman la arquitectura sin hacer grandes modificaciones.

Como opciones para esta propuesta se han considerado dos situaciones distintas. En primer lugar, que el simulador sea de una complejidad relativamente pequeña, en cuyo caso puede formar parte del Agente de Simulación, comunicándose directamente con él. En este caso, el Agente de Simulación se comportaría como un Wrapper (Hayden y otros, 1999), transformando el simulador en un agente a la vista de los demás agentes. Una segunda opción consiste en que el simulador sea una aplicación autónoma de complejidad mayor, en cuyo caso puede comunicarse con el STI a través del centro de mensajes.

Entre las situaciones que se han contemplado aparece la posibilidad de que el simulador sea de pequeño tamaño, pero realice frecuentes actualizaciones que deban verse reflejadas en el EV, en cuyo caso no interesa que se comunique directamente a través del Agente de Simulación, pues puede llegar a inundar el STI con sus mensajes. Otra posible situación consiste en que el simulador esté integrado junto con un EV, en cuyo caso se debe tener también en cuenta si el EV satisface el resto de necesidades del prototipo y, si es posible, hacer que los mensajes salgan del prototipo para que puedan ser utilizados por el resto de elementos del sistema.

Como puede apreciarse, la cuestión es lo suficientemente abierta como para encontrar una solución que satisfaga todas las posibles situaciones, así que será fácil realizar las modificaciones que sean necesarias en cada uno de los casos. Por ello, la integración de un simulador queda como cuestión parcialmente abierta para ser estudiada como una posible creación de una línea de producto (Clements y Northrop, 2001).

5.6.1.7. Deshabilitación de la Tutoría

Deshabilitar la tutoría es una funcionalidad de la aplicación que además está relacionada con la capacidad de modificación de la misma en tiempo de ejecución. La forma más inmediata de hacerlo consiste en no iniciar el Agente de Tutoría cuando comience a ejercitarse la actividad. De esta manera, al no estar presente este agente, las labores de tutoría quedan suspendidas, y no solicitará a ninguno de los agentes restantes que realicen las tareas asociadas a ellas. Por este

motivo, de deshabilitarse así la tutoría, será posible deshabilitar también otros agentes, como el de planificación.

Sin embargo, al desactivar los agentes completos también se desactivan otras acciones que podemos querer que se realicen, como la respuesta a preguntas del estudiante. Por ese motivo, se considera más conveniente la desactivación de comportamientos de manera selectiva.

Una tercera opción consiste en hacer que el agente de tutoría no se suscriba a la recepción de acciones del estudiante. De esta manera, el resto de funciones seguirán activas, pero no se realizará el seguimiento de las acciones del estudiante.

Cualquiera de las tres opciones es sencilla de realizar con la arquitectura propuesta, aunque la primera de ellas es la que en su momento puede acarrear más riesgos, ya que puede provocar problemas inesperados al desactivar funciones necesarias de forma inadvertida.

5.6.1.8. Deshabilitación del Cálculo de Trayectorias

Al igual que sucedía en el caso anterior, existe más de una forma en la que se puede desactivar el cálculo de trayectorias dentro del aprendizaje de una actividad.

Puesto que el cálculo de trayectorias lo activa y desactiva el Agente de Tutoría cuando solicita este servicio del Agente de Trayectoria, el hecho de que la actividad no lleve aparejado el desplazamiento por el entorno es una de las formas de hacer que el Agente de Tutoría no solicite este servicio.

Aunque, de manera análoga al caso anterior, se podría pensar en desactivar el Agente de Trayectoria completo, en este caso existe la posibilidad de que haya otros agentes que utilicen los servicios que presta, por lo que la opción más factible en este caso es que simplemente no se solicite este servicio.

5.6.2. Escenarios de Eficiencia

La velocidad en la ejecución de un EVEATI es un aspecto clave para conseguir el éxito en la experiencia del estudiante, ya que una respuesta pobre ante sus acciones puede desembocar en frustración o aburrimiento y, por ende, en el abandono del entorno.

Entre las causas de una ejecución pobre se encuentran, por un lado, el hecho de que el EV en sí

funcione despacio o a saltos, por lo que es precisa una programación cuidadosa del mismo para que la ejecución se realice de manera contínua, sin interrupciones y suave. Es por ello que uno de los requisitos de modificabilidad tiene que ver con la posibilidad de incorporar EVs desarrollados por terceras partes que estén especializadas en la implementación de este tipo de sistema.

La otra causa se encuentra en respuestas con excesivo retardo ante las acciones del estudiante, lo cual puede llegar a provocar que el usuario realice varias acciones antes de recibir la respuesta a la primera de ellas. Esto, a su vez, puede originar confusiones, ya que el estudiante puede no identificar a cuál de las acciones corresponde la respuesta que acaba de recibir.

A estas dos situaciones que pueden resultar complejas se refieren los dos escenarios de eficiencia que se exponen a continuación.

5.6.2.1. Respuesta ante Acciones

Parte del éxito del prototipo radica en la buena velocidad de ejecución . Adicionalmente, el hecho de que los agentes trabajen de manera concurrente posibilita que la velocidad de ejecución sea más fluida, especialmente si la máquina donde se ejecutan permite el procesamiento en paralelo o si la plataforma utilizada facilita la distribución del conjunto de agentes en varias máquinas.

Además, el orden de ejecución de las acciones también determina que, cuando se reciba una acción del estudiante, esté todo preparado para evaluarla, y cuando se haya hecho esto, se terminará de trabajar con ella y se preparará la llegada de la siguiente.

De esta forma, antes de que llegue una acción, el agente de tutoría debe haber solicitado al agente de planificación la siguiente acción correcta. Cuando llegue la acción se deberá validar y enviar la respuesta al estudiante, tras lo cual se evaluará la corrección de la misma para dársela al Agente de Estudiante y se solicitará la siguiente acción al Agente de Planificación para iniciar nuevamente el ciclo.

Llegado el caso en que se perciba que la velocidad de respuesta es algo lenta, se ha pensado en incorporar una pequeña mejora, que consiste en evaluar de antemano la acción que debería realizar el estudiante, de forma que la respuesta será inmediata.

Si el estudiante se equivoca continuamente, en muchos casos se realizarán el doble de evaluaciones, por lo que el STI estará más cargado. En ese caso, se puede complementar esta técnica con consultas al modelo del estudiante, de manera que si el estudiante no se suele

equivocar se realice la validación por adelantado y, en caso contrario, se espere a que llegue la acción.

Para hacer esto último, bastaría con reutilizar uno de los servicios que ya presta el Agente de Estudiante, aunque con unos fines distintos a los previstos originalmente. Esto da una muestra de una posible forma de reutilizar funcionalidades proporcionadas por los agentes.

Otra opción consistiría en la desactivación de la validación de las acciones, lo cual condicionaría la estrategia de tutoría a utilizar. Además, esta alternativa impone restricciones en las opciones de interacción disponibles en el EV, que deben ser siempre válidas, aunque no sean correctas.

5.6.2.2. Respuesta en Navegación

Cuando el estudiante se desplaza por el EV, sus coordenadas se envían continuamente al STI para que pueda supervisarse su trayectoria. Aunque el envío de coordenadas es un paquete de tamaño reducido, la ejecución continuada puede provocar el STI reciba demasiada información, en ocasiones innecesaria, y que se sobrecargue ligeramente al cliente, lo cual puede constituir un factor de retardo en la ejecución.

El EV debe permitir decidir cada cuánto tiempo se quieren enviar las coordenadas al STI, de manera que pueda mantenerse un equilibrio entre el número de coordenadas que se envían para reproducir la trayectoria fielmente y la velocidad de ejecución del mismo.

Si fuera necesario, puede implementarse un sencillo mecanismo que regule cada cuánto tiempo se envían coordenadas en función de la velocidad de ejecución del EV. También puede suceder que el EV envíe más coordenadas de las que el Agente de Trayectoria puede o debe procesar, por lo que éste debe disponer de un mecanismo que le permita descartar coordenadas.

5.7 Síntesis de la Metodología y criterios de diseño

5.7.1 Recomendaciones para la Arquitectura

Aunque el método ADD (Bass y otros, 2003) constituye un buen punto de partida, en poco tiempo quedó patente que aportaba una visión sobre el diseño de arquitecturas que no resolvía las necesidades de la presente investigación. De hecho, en (Haythorn, 1994) ya se apunta a los

inconvenientes que genera esta forma de diseñar, que si bien puede ser adecuada como técnica de programación, no tiene por qué dar el mismo resultado al definir una arquitectura.

Por ello, basándose en ese método, se ensayó una manera distinta de abordar el problema que, en vista de los resultados obtenidos, constituye una aproximación más adecuada para una tarea como la abordada en este documento. De esta forma y considerando la utilidad para la definición de la arquitectura descrita a lo largo de todo el capítulo, a continuación se presentan unas recomendaciones que servirán como guía de apoyo para extender la arquitectura que constituye la solución a uno de los problemas planteados en este trabajo.

En primer lugar se describe la manera de decidir qué elementos de la arquitectura utilizar para construir un nuevo prototipo. Inmediatamente despúes, se describe la forma de introducir nuevos elementos en la arquitectura general del prototipo, y a continuación se muestra la manera de llevar a cabo la misma tarea dentro de la arquitectura del prototipo.

5.7.1.1. Selección de Componentes de la Arquitectura

A la hora de desarrollar un EVEATI a partir de esta arquitectura, resulta complejo asimilar toda la información presente en la documentación generada, sobetodo cuando prácticamente no existen experiencias previas y el equipo empieza desde cero. Por ese motivo, si no se necesita utilizar todos los componentes descritos, cabe la posibilidad de que se cometan errores a la hora de decidir qué componentes resultan necesarios y cuáles no.
Con objeto de facilitar la labor de quien desee llevar a cabo esta tarea, a continuación se propone la utilización de una lista de comprobación que ayude a decidir qué elementos de la arquitectura utilizar y cuáles se pueden descartar. Mediante la selección de determinadas opciones y la respuesta a una serie de preguntas que tienen que responder los responsables de definir la arquitectura sobre cuáles de los elementos descritos le resultarán de utilidad en el nuevo sistema a construir.

Para poder identificar qué elementos será necesario incluir en la arquitectura se ha recurrido a la documentación generada para describir la arquitectura, en concreto a la relativa a dos de las vistas: la vista de uso y la vista de edición-suscripción.

Según Clements y otros, (2002a), la vista de uso documenta la relación de uso existente entre los diferentes elementos de la arquitectura. Entre otras utilidades, la vista de uso puede utilizarse para dirigir el proceso de desarrollo, ya que se pueden identificar los elementos que deben estar presentes para que otro elemento dado pueda funciona

Con este objetivo, se ha elaborado un diagrama que muestra una vista conjunta de las relaciones de uso que se dan entre los agentes que forman el STI (Figura 5.70), lo que a su vez servirá para identificar, dadas unas necesidades, qué elementos deben incluirse para satisfacerlas.

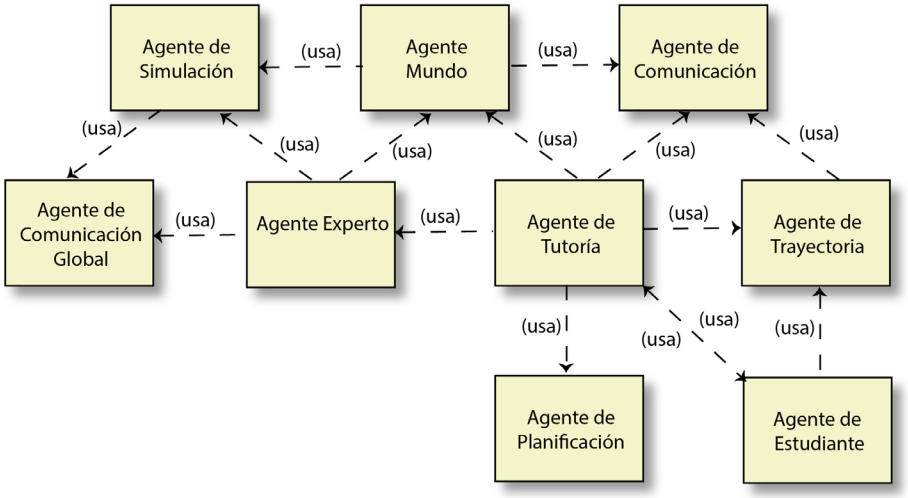


Figura 5.70: Vista de Uso de los Agentes del STI.

Una vez identificados los agentes que deben estar presentes, se han utilizado las tablas de la sección del Editor-suscriptor para seleccionar los servicios que deben ofrecer cada uno de ellos. Ahí se muestran los servicios de los que hace uso cada uno de los elementos que forman la arquitectura propuesta, junto con los elementos que los prestan. De esta forma, una vez identificados los elementos que deben estar presentes a través de las vistas de uso, se puede hacer uso de estas tablas para decidir qué servicios debe implementar cada uno de ellos.

Considerando la utilidad de ambas vistas, se propone la utilización de las listas de comprobación que se muestran a continuación para seleccionar los elementos que deben incluirse en la arquitectura de la aplicación.

Los servicios que no solicita directamente ningún agente se ejecutan como respuesta a un mensaje recibido a través de una suscripción. Así, cuando un agente se suscriba a un servicio habrá que marcar los servicios de ese agente que se ejecutan cuando llega el mensaje correspondiente. De esta manera, se puede obtener una primera visión de los elementos de la arquitectura que serán necesarios para la implementación de un sistema concreto. En cualquier caso, se recomienda la consulta de la documentación de la arquitectura para llevar a cabo una selección final de los elementos de la arquitectura a utilizar.

A continuación se muestran las listas de comprobación que se han elaborado para ayudar a realizar la selección de elementos de la arquitectura.

1. Selección de Subsistemas

2. Si se necesita disponer de tutoría inteligente, utilizar el STI.

3. Si se va a interactuar a través de un EV, añadir este elemento a la arquitectura.

4. Si existe alguna aplicación que simule el comportamiento de elementos del EV, añadir el Simulador.

5. Si se han seleccionado por lo menos dos de las anteriores, utilizar el Centro de Mensajes. Opcionalmente, utilizarlo también si sólo se seleccionó una.

Selección de Agentes, comportamientos y servicios

1. Incluir los Agentes de Comunicación y Comunicación Global o Mundo, junto con sus comportamientos y servicios. Si no se van a enviar respuestas a los estudiantes no incluir los servicios de Enviar Información.

2. Si se va a hacer uso de un simulador, seleccionar el Agente de Simulación, junto con sus comportamientos y servicios.

3. Si se va a realizar el seguimiento de las acciones de los estudiantes, añadir al Agente de Tutoría y al Agente de Estudiante. Para éste, añadir los comportamientos de Anotar Acción y Anotar Actividad, así como los servicios del mismo nombre.
4. Si se va a realizar tutoría sobre las acciones de los estudiantes, añadir los comportamientos de Tratar Acción y Tratar Actividad del Agente de Tutoría, así como los servicios del mismo nombre. Para el Agente de Estudiante, añadir también los servicios de Anotar Evaluación de Acción y Anotar Evaluación de Actividad.

5. Si se permite que el alumno pueda resolver una actividad por distintos caminos, incluir el Agente de Planificación junto con sus servicios. Añadir también los Agentes Global o Mundo y Experto, junto con sus comportamientos y servicios para validar precondiciones y ejecutar postcondiciones y acciones.

6. Si se va a permitir la realización de preguntas, incluir el comportamiento de Tratar Pregunta del Agente de Tutoría, el de Anotar Pregunta del Agente de Estudiante y el servicio de Consultar del Agente Mundo. En caso de haber incluido un Simulador, si dispone de la capacidad de proporcionar información para responder preguntas, añadir el Agente de Simulación, junto con el servicio Solicitar Información.

7. Si se va a realizar el seguimiento de las trayectorias del estudiante, añadir el Agente de Trayectoria, junto con sus comportamientos y servicios. Añadir también el comportamiento de Tratar Movimiento del Agente de Tutoría y el de Anotar Movimiento del Agente de

Estudiante.

8. Si se va a personalizar la tutoría según las características del estudiante, añadir el comportamiento de Dar Información del Estudiante del Agente de Estudiante.

5.7.1.2. Realización de Modificaciones en el Prototipo

La introducción en la arquitectura de un nuevo subsistema, como podría ser un Entorno Virtual diferente u otra aplicación que deba comunicarse con los Entornos Virtuales, el Simulador o el Sistema de Tutoría Inteligente, requiere la realización de una serie de tareas que se detallan a continuación: En primer lugar, se debe verificar que el nuevo subsistema permite realizar una comunicación basada en paso de mensajes a través del Centro de Mensajes, tal y como aparece ya referido y que se reproduce a continuación para mayor comodidad. En caso de no ser así, se deberá introducir un elemento de nueva creación que sea capaz de realizar la comunicación entre el nuevo subsistema y el Centro de Mensajes.

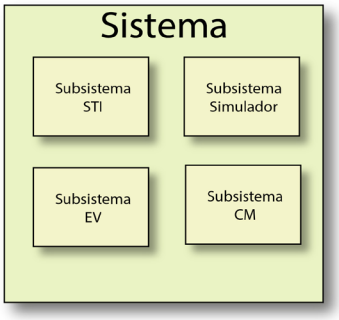


Figura 5.71 Descomposición del prototipo

Después deberán revisarse los escenarios de calidad existentes para comprobar si la introducción del nuevo subsistema interfiere con alguno de ellos. Además, se deberán elaborar nuevos escenarios de calidad en caso de que la introducción del nuevo subsistema lo requiera.

A continuación se deberán listar los tipos de mensajes que los subsistemas existentes envían al Centro de Mensajes, y extraer de entre ellos los que el nuevo subsistema debe recibir, de manera que pueda realizarse la suscripción a los mismos en el Centro de Mensajes. Posteriormente, se deben analizar las necesidades de los subsistemas existentes que puede satisfacer el nuevo subsistema, e identificar en función de las mismas los mensajes queéste debe enviar al Centro de Mensajes.

Posteriormente, será necesario actualizar los subsistemas existentes, para que se suscriban a los nuevos tipos de mensajes, si los hay, y añadir las modificaciones pertinentes en los mismos para

que estos subsistemas puedan realizar el tratamiento de los nuevos mensajes.

Finalmente, deberán comprobarse nuevamente los escenarios de calidad existentes y verificar que la introducción del nuevo subsistema no ha provocado el incumplimiento de ninguno de ellos.

El procedimiento descrito se ha elaborado bajo la suposición de que la introducción del nuevo subsistema no implica descartar los atributos y escenarios de calidad manejados hasta el momento. En caso de que las necesidades hayan variado notablemente, será necesario someter a la arquitectura a un estudio más minucioso que excede los objetivos de estas recomendaciones metodológicas.

5.7.13 Realización de Modificaciones en el STI

Para introducir modificaciones en el Sistema de Tutoría Inteligente se tiene que realizar un proceso similar al que se sigue para introducir cambios en el sistema. En este caso, sin embargo, deben tenerse en cuenta los servicios que prestan los agentes en lugar de los mensajes que envía cada uno de los subsistemas.

En primer lugar se debe especificar la funcionalidad del elemento que se desea introducir en el STI, para así poder comprobar si parte o la totalidad de esa funcionalidad la presta alguno de los agentes presentes. En caso de que se desee sustituir un agente existente en el STI, se deberá comprobar que el nuevo agente proporciona, al menos, toda la funcionalidad proporcionada por el agente que se va a eliminar, o que la funcionalidad que no se proporciona no será requerida por los agentes restantes.

A continuación se deberán revisar los escenarios de calidad existentes, para comprobar si son suficientes para la funcionalidad que se desea introducir. En caso contrario, deberán elaborarse los escenarios de calidad correspondientes a la funcionalidad que se va a introducir en el STI.

Posteriormente debe decidirse si la funcionalidad se introduce en forma de un nuevo agente dentro del STI, como un nuevo comportamiento de un agente existente o como parte de un comportamiento de alguno de los agentes del STI. Posteriormente, se deberán identificar los servicios que el nuevo agente o comportamiento necesita de los que hay presentes en el STI. Si alguno de los servicios no lo presta ninguno de los agentes, se deberá decidir si se incluirá en alguno de los agentes existentes, como parte de un nuevo agente o incluso si se puede prescindir de él.

De manera análoga, se deben definir los servicios que los agentes existentes necesitan del elemento nuevo, y se crearán los comportamientos necesarios para prestar los servicios demandados. También, de manera similar al paso anterior, si alguno de los servicios necesarios no se prestan, se deberá decidir si incluirlos en el nuevo agente, como parte de un agente existente o si se puede prescindir de ellos.

Finalmente, debe comprobarse que tras la introducción de la nueva funcionalidad en el STI se siguen satisfaciendo los escenarios de calidad que afecten tanto al STI como al prototipo en su totalidad.

Al igual que sucedía en la sección anterior, este proceso es válido en casos en los que las modificaciones a realizar se ajusten a los criterios de diseño utilizados para diseñar esta arquitectura, los cuales se muestran en la siguiente sección, y siempre que la nueva funcionalidad no rompa con la funcionalidad ofrecida por el STI.

5.8 Criterios para Realizar el Diseño de la Arquitectura

Si se diseña una aplicación para que sea modificable, lo será en aquellos aspectos que se hayan considerado para generar cambios con facilidad, pero habrá otros cambios que serán difícilmente realizables porque rompen del todo la filosofía del diseño o, simplemente, porque nunca se pensó el prototipo para que esos cambios se pudiesen introducir, fuese o no de manera intencionada.

Como se dice tanto en (Haythorn, 1994) como en (Bass y otros, 2003), la manera de saber si un sistema es modificable consiste en hacer una lista de posibles modificaciones para contrastar el diseño con los cambios propuestos. De esta forma, sabremos si el sistema admite o no los cambios listados y cuál es la implicación de dichos cambios.

Lo que se propuso fue comenzar desarrollando una parte pequeña y central del sistema, y contemplar como posibles cambios las características que sabemos que hay que añadir a la arquitectura. En la medida en que esas características vayan añadiéndose, se irá comprobando que las demás modificaciones se pueden seguir realizando y se irán resolviendo las dificultades que puedan ir apareciendo. Esta manera de proceder es similar a lo que se propone en el desarrollo iterativo e incremental (Pressman, 2004; Sommerville, 2007).

Con esto, podremos estar razonablemente seguros de que se podrán incluir las características requeridas. Para ello, es necesario que la arquitectura se haya diseñado utilizando algún mecanismo lo suficientemente flexible como para incorporar cambios no contemplados *a priori*, aunque siempre teniendo en cuenta que será difícil que se pueda incorporar de forma sencilla,

cualquier tipo de cambio. En el caso concreto de esta propuesta, los mecanismos planteados para ello son la utilización de servicios y el esquema editor-suscriptor.

5.8.1 Organización Peer-To-Peer

Así, lo que nos ha llevado hacia una organización de tipo peer-to-peer no es sino la facilidad que proporciona para introducir nuevos tipos de agentes en la medida en que se vayan añadiendo nuevas funcionalidades que requieran su inclusión. Esta facilidad viene dada, sobre todo, por la ausencia de unas reglas que puedan llegar a dificultar la introducción de nuevos agentes.

Esto no quiere decir que la arquitectura vaya a ser más fácilmente modificable sólo por el hecho de tener una estructura peer-to-peer; para ello habrá que complementarla con otros elementos. Lo que sí es cierto es que no va a suponer una dificultad adicional a la hora de modificar el prototipo, lo que sí parece suceder, por el contrario, con sistemas de tipo jerárquico.

Por tanto, la propuesta realizada en el presente trabajo para conseguir un sistema que resulte fácilmente modificable se basa en la utilización de una organización de tipo peer-to-peer, complementada por otros mecanismos que se describen en el presente capítulo.

5.8.2 Edición-Suscripción

El esquema de edición-suscripción (publish-subscribe), como solución de diseño orientado a objetos, aparece publicada en (Gamma y otros, 1993; Gamma y otros, 1995,), y posteriormente en (Buschmann y otros, 1996). En ambos casos se identifica también como patrón observador. Como elemento de diseño con agentes, aparece referenciado de forma similar a la utilizada en este trabajo en (Silva y otros, 2005; Méndez, 2008).

En la descripción que se hace en ambos trabajos, el patrón aparece muy relacionado con el modo de funcionamiento de las interfaces gráficas de usuario, donde un mismo evento puede ser notificado a varios elementos, y donde un cambio puede mostrarse por varias salidas. A este comportamiento es al que en (Clements y otros, 2002a) se denomina invocación implícita. Esta invocación implícita consiste en que los consumidores de información se suscriben a la recepción de determinados eventos a través de la implementación de unos métodos concretos, mientras que los productores anuncian los eventos escribiéndolos en el lugar que la infraestructura de comunicación determine. Una vez escritos, la infraestructura de comunicación los reparte entre los suscriptores en un orden preestablecido, que puede ser el orden de suscripción, mediante la

invocación de los métodos anteriormente mencionados.

Al contrario de lo que sucede con los objetos, la utilización de agentes nos facilita la introducción de una pequeña variante, que es la que se propone utilizar, para conseguir también desacoplar a los consumidores de información del productor.

Esto se consigue a través de la introducción de un mecanismo de indirección, las páginas amarillas, que permiten a los productores anunciar los tipos de eventos o información que pueden producir, en lugar de anunciar sólo los eventos.

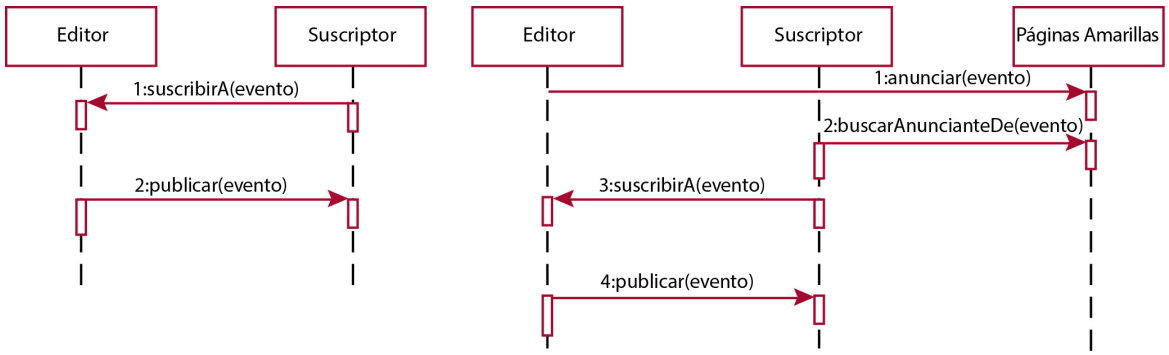


Figura 5.72: Variación del patrón Editor-Suscriptor. A la izquierda, el esquema original, y a la derecha, el modificado con el uso de las páginas amarillas

De este modo, los usuarios tienen un conocimiento de la información que les interesa, algo que ya sucedía antes, si bien con las páginas amarillas se consigue que no necesiten un conocimiento directo y explícito del productor de esa información. Ahora pueden preguntar quién produce lo que les hace falta, evitando conocer directamente a los productores. Lo único que llega a saber un agente es que se comunica con el agente que proporciona X, que puede ser cualquiera de los agentes que se encuentran activos en ese momento.

Empleando este mecanismo se consigue, sobre el papel, que los editores y los suscriptores actuén de manera completamente independiente de para quién publiquen o de quién consuman. De esta manera, siempre que se mantengan los servicios proporcionados por un agente, se podrá sustituir éste por uno o varios agentes diferentes que presten los mismos servicios sin que los consumidores se vean afectados, lo cual amplía el grado de modificabilidad proporcionado por el patrón editor- suscriptor tradicional.

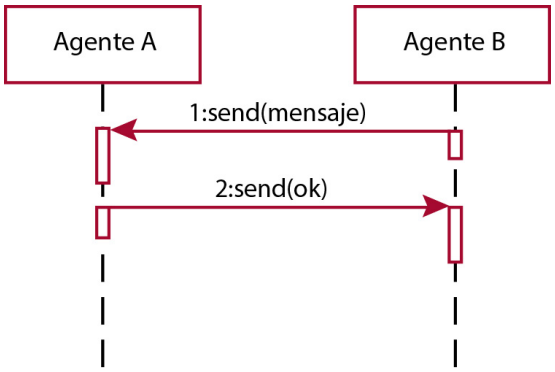


Figura 5.73: Confirmación de recepción de mensajes

5.8.3. Relaciones de Uso

A lo largo del presente trabajo se ha mencionado repetidamente que uno de los principales objetivos que se persiguen es el diseño de una arquitectura fácilmente modificable para construir EVEATI. Un indicador utilizado para medir la modificabilidad de un diseño es el número de relaciones existentes entre los distintos módulos del mismo (Bass y otros, 2003). Sin embargo, en este trabajo no se ha hecho uso de él por varias razones.

En primer lugar, aunque un diseño en el que existan muchas relaciones entre módulos puede ser indicativo de un acoplamiento alto y, por tanto, de una modificabilidad compleja, lo contrario no tiene por qué ser cierto, es decir, pocas relaciones entre módulos no son necesariamente una indicación de un diseño modificable. Una relación entre dos módulos puede comprometer enormemente un diseño si esa relación no se realiza de la manera adecuada. Esto sucede si, por ejemplo, a través de una única llamada a una función se obtienen datos que deberían ser conocidos únicamente por quien los ha proporcionado. Además, la existencia de pocas relaciones entre módulos también puede ser un síntoma de un criterio pobre a la hora de decidir cómo realizar una división del prototipo en módulos. Así, un sistema monolítico que se integra de un único módulo sería el ideal de prototipo modificable, cuando en general esto nunca es así.

En segundo lugar, una relación de un módulo con otro puede esconder una relación, directa o indirecta, con un tercero, y esa relación también afecta a la modificabilidad del prototipo (Clements y otros, 2002a). Por esa razón, se estima que, de cara a llevar a cabo un estudio de la modificabilidad del prototipo, es interesante disponer de la documentación de una vista de uso que muestre las relaciones de uso entre los distintos módulos del sistema.

Tal y como describen los autores de (Clements y otros, 2002a), las relaciones de uso se dan no sólo a través de llamadas directas de un módulo a otro, sino también, por ejemplo, a través de información que se comparte por medio de un elemento intermedio, del estado en que se deja a un módulo de uso común o mediante una llamada a un tercer módulo que se realiza para satisfacer una petición.

Esta relación dará una visión algo más certera de las dependencias entre módulos que la mera existencia de llamadas directas entre ellos.

5.8.4 Complementos finales de Arquitectura

La propuesta que se realiza en este trabajo consistió en definir una arquitectura para EVEATI que sirva como base para desarrollarlos, dejando libertad para que quien utilice la arquitectura pueda concretar, en una fase posterior del diseño, la manera en que se llevan a cabo las funciones de cada uno de los módulos que componen el prototipo.Además, se ha tratado la problemática de que los distintos módulos se puedan agregar y eliminar en función de los requisitos que plantee la situación de aprendizaje, lo cual hace que el prototipo se pueda adaptar a las necesidades que surjan en cada momento.

Esta perspectiva proporciona un marco en el que encuadrar el uso de, por ejemplo, distintos métodos de planificación o de cálculo de trayectorias en un Entorno Virtual, además de poder activarlos o desactivarlos cuando sea requerido.Para ello, ha sido necesario abordar el diseño de la arquitectura software del sistema de forma que se considere explícitamente qué modificaciones debe soportar la arquitectura, de forma que se puedan incluir en el diseño mecanismos que faciliten esta labor.

El estudio de los trabajos que se retomaron y aplicaron en este capítulo permite, por un lado, visualizar las características que es frecuente encontrar en este tipo de sistemas, lo que ha servido para poder tenerlas en cuenta a la hora de decidir qué funcionalidades debía soportar la arquitectura y qué bloques de funcionalidad deseábamos que se pudiesen sustituir. Por otro lado, también ha resultado útil para identificar la forma en que se suele abordar la construcción de estos sistemas y para revelar las carencias que suelen existir en su desarrollo.

En este punto se tomó la decisión de diseñar una arquitectura lo suficientemente abierta como para que permita distintas soluciones presentes en la literatura revisada, además de completarla

con una serie de recomendaciones metodológicas que ayuden a mantener las propiedades de las que se dotaría a la arquitectura.

Finalmente, con el apoyo tanto en la arquitectura como en las recomendaciones metodológicas, se realizó la implementación de un framework[8] que sirvió como base para la integración de los elementos presentes en la arquitectura, que pudiera ser utilizado para desarrollar EVEATI sin necesidad de realizar todo el esfuerzo de implementación desde el principio, y que facilite la adopción de la arquitectura propuesta.

Del análisis de todos los aspectos tanto teóricos como metodológicos, se recopiló una lista de los elementos presentes en distintos sistemas que se consideraron más importantes para ser incluidos en la propuesta de solución:

De este análisis surge la necesidad de considerar como componentes básicos de la arquitectura elementos que permitan incluir tanto la estrategia de tutoría como el modelado del estudiante, así como el conocimiento experto sobre el dominio del problema, que son los elementos que conforman la estructura básica de un STI. También es un elemento básico el EV, sin el cual el prototipo pierde gran parte de su utilidad como entorno de aprendizaje.

Además, por el hecho de estar realizando un proceso educativo dentro de un EV, se hace necesario modelar cierta información semántica que represente las acciones que el estudiante puede realizar en él, así como los objetos con los que puede interactuar y su estado. Para ello se considera necesaria la inclusión de un elemento que maneje la información del mundo virtual y que se concluye que fundamentalmente se encuentra en el modelado de las interfaces.

La necesidad de realizar el aprendizaje en entornos complejos en los que se interactuá con elementos que no están controlados directamente por los usuarios, sino por algún tipo de simulación, marca la necesidad de incluir otro elemento que se haga cargo de la interacción con dicha simulación.

Para que la tutoría sea flexible y que dé libertad a los estudiantes, algunos sistemas plantean la posibilidad de no dirigir al estudiante por un único rumbo, sino que se le da la posibilidad de seguir el camino que considere más adecuado y el prototipo adapta la solución a la situación creada por el estudiante cuando es necesario. Con el objetivo de llevar ésta posibilidad hasta los extremos más amplios, se propone la utilización de un planificador que trace la solución a un ejercicio en la

8 La palabra inglesa "**framework**" (marco de trabajo) define, en términos generales, un conjunto estandarizado de conceptos, prácticas y criterios para enfocar un tipo de problemática particular que sirve como referencia, para enfrentar y resolver nuevos problemas de índole similar.

medida en que un estudiante vaya avanzando por el camino que elija para su resolución.

Finalmente, también se identifican las interacciones que típicamente tienen lugar en estos sistemas para darles soporte dentro de la arquitectura, lo que nos lleva a identificar la realización de acciones y de preguntas y la navegación por el Entorno Virtual.

Una vez identificadas estas características, se valoran las distintas opciones disponibles para diseñar e implementar el prototipo. Se llega a la conclusión de que la utilización de agentes software constituye una opción prometedora que puede facilitar la tarea que se aborda y que puede incluso aportar propiedades de interés a la arquitectura.

Al tomar esta decisión, se opta también por acudir al área de la ingeniería de software orientada a agentes para desarrollar la arquitectura, eligiendo una organización jerárquica para el diseño de la arquitectura. A medida que la arquitectura va siendo diseñada e implementada se hace patente que hay algo que no se ha considerado, ya que la arquitectura y el prototipo implementados no presentan las propiedades que se esperaba obtener de ellos.

Se decide realizar un análisis de la arquitectura, que muestra que el enfoque jerárquico y la manera de llevarlo a la práctica no constituyen una buena solución para el problema planteado.

Tras un largo periodo de análisis, confrontación con la teoría y revisión de otros casos exitosos, se acude al área de las arquitecturas software como camino para resolver los problemas encontrados en la solución inicial. Se identifican métodos de diseño y evaluación de arquitecturas, y se opta por utilizar los desarrollados en el SEI por cubrir todo el espectro de diseño, documentación y evaluación de las arquitecturas.

Se recopila un conjunto de escenarios de calidad que pone sobre la mesa los atributos de calidad que se le quieren dar a la arquitectura y que, aunque conocidos de forma implícita, nunca se han enunciado de forma explícita. Es en este momento cuando cobran forma y permiten dirigir el diseño con unos objetivos claros.

Después de considerar la posibilidad de no usar agentes en el nuevo diseño de la arquitectura, pues los expertos en el área dicen que la arquitectura es independiente de estas cuestiones, se llega a la conclusión de que su inclusión facilita la utilización de determinados mecanismos presentes en el paradigma de la orientación a agentes. Esta decisión, conduce a la solución descrita en este capítulo, cuya evaluación confirma que posee las propiedades de las que se le pretendía dotar en la fase de diseño de la arquitectura.

El diseño de esta arquitectura se ha visto acompañado del desarrollo de una segunda implementación, que actualmente se encuentra en desarrollo y que en los próximos meses se pretende evaluar, para que pueda utilizarse en el desarrollo de EV y que es una de líneas principales de investigación futura que tomará esta tesis.

5.8.5 Resultados Relativos a la Arquitectura

El resultado de la investigación es una arquitectura software para EVEATIs modificable y flexible. En esta arquitectura se han definido los elementos que deben formar parte del EV, así como los que deben conformar el STI, que es uno de los elementos integrantes del sistema de aprendizaje.

Como primera aportación relevante del diseño de la arquitectura se encuentra el hecho de que incluye elementos que no contemplan otras arquitecturas de manera global, como son el planificador, el planificador de rutas, el modelo sintáctico-semántico del mundo virtual y el simulador, además de los componentes básicos de un STI, como son el módulo de tutoría, el experto y el de modelado de los estudiantes.

Con esto se consigue que la arquitectura admita la utilización de un mayor número de funciones que otras arquitecturas propuestas anteriormente, y por lo tanto permita el desarrollo de sistemas más complejos y completos que los analizados en el capítulo 2.

Además, se ha preparado la arquitectura para que admita la inclusión, de manera sencilla, de otros elementos de importancia, como un agente tutor virtual que controle la presencia física del tutor en el Entorno Virtual. También se han definido unos comportamientos básicos para cada uno de los agentes, junto con los servicios que presta cada uno de ellos, de manera que sirvan como base para desarrollar la funcionalidad que debe llevar a cabo cada uno de los agentes que forman el prototipo.

Otra aportación importante de esta tesis se encuentra en la aplicación de métodos de diseño y evaluación de arquitecturas software a la elaboración de una arquitectura basada en agentes, que en pocos proyectos se incluye como parte de la documentación general y que para el equipo de trabajo resultó básica para lograr definir si en efecto se trataba de una propuesta innovadora y con todos los atributos que debería de tener.

Actualmente, la comunidad que investiga sobre sistemas multiagente ha prestado poca atención al trabajo realizado en el área de las arquitecturas software, obviando los resultados obtenidos

en un área que también debería ser relevante para el desarrollo de sistemas multiagente y que cuenta con mayor experiencia en el desarrollo de software.

En la elaboración del estado de la cuestión se han encontrado muy pocos ejemplos de trabajos que hayan realizado incursiones en ambas áreas. Destacan Boucké y otros (2006), Boucké y Holvoet (2008), Woods y Barbacci (1999) y Méndez (2008), los cuales se han centrado más en el proceso de evaluación de la arquitectura que en su diseño.

Sin embargo, es precisamente al diseño a lo que hay que prestarle mayor atención inicialmente, ya que la evaluación nos puede decir si una arquitectura presenta unos determinados atributos de calidad o no, pero la manera de hacer que los presente es precisamente a través del diseño. Este trabajo constituye, por tanto, una aportación novedosa en lo que a la aplicación de métodos de diseño de arquitecturas software a los sistemas multiagente se refiere.

Esta experiencia ha proporcionado un resultado inesperado y para la autora, altamente sorprendente, ya que el equipo de desarrollo se ha aventurado a la utilización de agentes en el diseño de la arquitectura. Esta decisión permite incluir mecanismos de diseño arquitectónico que posiblemente no se utilizarían de no haberse considerado explícitamente el uso de agentes en el diseño de la arquitectura.

Otra aportación original de este trabajo consiste en la aplicación al desarrollo de Sistemas de Tutoría Inteligentes basados en agentes de los mecanismos proporcionados por estos últimos para la consecución de una mayor modificabilidad de la arquitectura. De esta manera, la combinación de tres elementos como son las páginas amarillas, un esquema editor-suscriptor y la prestación de servicios por parte de los agentes, ha conducido a una solución que, de forma contrastada, proporciona al prototipo la modificabilidad buscada.

En resumen las siguientes serían algunas propiedades con las que se ha dotado a la arquitectura:

- No se trata de una arquitectura genérica que pueda ser utilizada para cualquier tipo de Entorno Virtual. Es válida para Entornos Virtuales de aprendizaje conceptual y procedimental, independientemente de su naturaleza, siempre que admitan la definición de las actividades como una serie de acciones que deben realizarse de manera secuencial.

- Permite enseñarle a de varios estudiantes de forma simultánea realizando tareas complementarias en una misma actividad.

- Permite la incorporación de aplicaciones distintas a las consideradas durante el diseño de la arquitectura a través del Centro de Mensajes. Estas aplicaciones podrán enviar mensajes, que serán recibidos por el resto de las aplicaciones que forman el prototipo de enseñanza si son de los tipos a los que se han suscrito. De igual manera, esta aplicaciones se podrán suscribir a los tipos de mensajes que deseen recibir.
- Permite deshabilitar los elementos que no sean necesarios en una determinada situación, sin afectar al resto de elementos del prototipo.
- Dentro del STI, permite la inclusión de nuevos agentes que hagan uso de los servicios proporcionados por los existentes, sin necesidad de realizar modificaciones en el prototipo.
- Permite deshabilitar comportamientos de los agentes que no realicen tareas de utilidad en una actividad determinada.
- Permite el intercambio de agentes por otros que proporcionen servicios análogos, sin necesidad de modificar los agentes que hagan uso de estos servicios.

Además, se ha definido un marco que permite determinar en qué situación y a través de la utilización de qué mecanismos puede afirmarse que un sistema basado en agentes es flexible.

Así pues, tras la realización de este trabajo, se dispone de argumentos sólidos para afirmar que se ha desarrollado una arquitectura que es modificable en aspectos determinados porque se ha diseñado para que así sea.

Este trabajo también constituye una experiencia novedosa, si bien no es la primera vez que se lleva a cabo, en lo que se refiere a la evaluación de una arquitectura software basada en agentes utilizando ATAM. Esta evaluación no ha puesto de manifiesto problemas relevantes relacionados con la arquitectura, puesto que el proceso de diseño ha permitido controlar bastante ese aspecto.

Sin embargo, lo que sí ha permitido esta evaluación de la arquitectura es la elaboración de una lista de características relevantes. Esta lista puede servir para tratar de minimizar los efectos de alguno de los inconvenientes del diseño, así como para dirigir las modificaciones que a buen seguro sufrirá este diseño en un futuro.

5.8.6 Resultados Relativos a las Sugerencia y el Diseño

Aunque una de las principales aportaciones de este trabajo lo constituye la arquitectura software para EVEATIs, la relativa complejidad de la misma junto con la cantidad de documentación que la

acompaña pueden causar que su utilización para algunos casos resulte muy complicada de llevar paso a paso. Por este motivo, la aportación introducida por las recomendaciones metodológicas y de diseño, en cuanto a que proporcionan una ayuda para comprender los fundamentos que han guiado el diseño de la arquitectura, por un lado, y guían a los usuarios de la misma en el proceso de modificación de sus distintas partes para adaptarla a sus necesidades.

De esta manera, las recomendaciones constituyen una aportación relevante en cuanto que guían al usuario de la arquitectura en la utilización y modificación de la misma. Todas estas recomendaciones facilitan la adopción de la arquitectura por parte de otros grupos, lo cual puede ser una posibilidad para enriquecer o difundir esta propuesta. Esto se ha realizado a través de la elaboración de una lista de comprobación que sugiere los elementos de la arquitectura a utilizar en función de los requisitos de la aplicación a construir.

Por otra parte, al indicar cómo se deben llevar a cabo las modificaciones, se proporciona un medio para que la arquitectura no se degrade por causa de las modificaciones realizadas.

Adicionalmente, las recomendaciones de diseño aportan la visión de conjunto de los criterios utilizados en la elaboración de la arquitectura, los cuales constituyen un elemento que permite mantener la integridad conceptual de la misma y confrontarla de cara con las recomendaciones que se presentan en el siguiente capítulo. Esto representa uno de los principales factores que permiten mantener los atributos de calidad con los que se ha dotado a la arquitectura descrita.

Por lo tanto, la principal aportación realizada con la elaboración de estas recomendaciones es su contribución a mantener la integridad conceptual de la arquitectura, lo que retrasa el inevitable proceso de deterioro causado por la sucesivas modificaciones que es de esperar que tengan lugar.

5.8.7 Resultados Relativos a la Implementación de la Arquitectura

La definición de la arquitectura objeto del trabajo se ha utilizado para diseñar e implementar Entornos Virtuales que utilizan la estructura y los mecanismos definidos en ella.

El resultado de esta implementación es un framework que ya se ha probado con éxito en un entorno de aprendizaje sencillo de complejidad creciente. Gracias a estos resultados, el equipo trabaja actualmente en un ejemplo de aplicación real, de complejidad más elevada, para la formación de competencias docentes.

Actualmente, la mayor complejidad de la aplicación de la implementación realizada a distintos entornos radica principalmente en lo laborioso que resulta el proceso de definición del entorno de aprendizaje. Por una parte, se debe definir el dominio del problema y los operadores que debe utilizar el planificador. Por otra, se debe elaborar toda la información semántica relativa al Entorno Virtual que debe utilizar el Agente Mundo. Ambas labores son las que, una vez implementado el framework, requieren la mayor parte del esfuerzo de desarrollo de nuevas tareas de entrenamiento.

Por ello, una de los trabajos más fuertes que surgieron una vez revisada esta propuesta actualmente se concentra en la elaboración de herramientas de autor que faciliten la creación de nuevas actividades (ver Figura 5.78).

En lo relativo a la implementación del prototipo en sí,ésta se ha llevado a cabo utilizando distintos lenguajes de programación para cada uno de los subsistemas que integran la aplicación. Todos ellos han podido comunicarse entre sí a través del Centro de Mensajes, lo que constituye una primera prueba de la utilidad de este medio de comunicación y de la flexibilidad que introduce en el prototipo.

Durante el desarrollo de la aplicación se hace necesario realizar numerosas pruebas para comprobar su correcto funcionamiento. Para probar las distintas funciones del STI, resulta excesivamente complejo tener que arrancar el sistema completo y realizar todo el procedimiento de entrenamiento para verificar una funcionalidad que quizás no aparece hasta pasado un determinado tiempo.

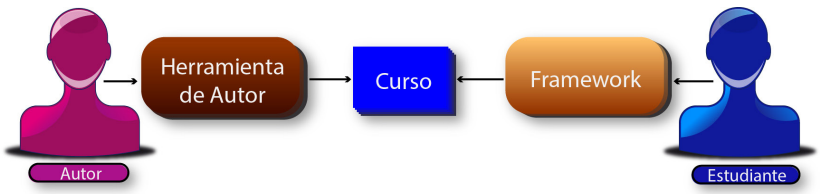


Figura 5.74 Herramienta de autor

Para facilitar esta tarea, se han desarrollado dos pequeños clientes programables que sustituyen al Entorno Virtual y que funcionan de manera autónoma, lo que ha permitido agilizar la realización de dichas pruebas (ver Figura 5.79).

Figura 5.75 Clientes para pruebas

Adicionalmente, la utilización de estos sustitutos ha supuesto una evidencia más de la flexibilidad que proporciona el Centro de Mensajes al prototipo, ya que han podido utilizarse los dos clientes de manera indistinta sin necesidad de realizar ninguna modificación en el resto de aplicaciones.

En cuanto a la implementación del STI, se ha realizado en pequeños incrementos que se han iniciado con el Agente de Comunicación Global y el de Tutoría. Este método de desarrollo ha servido también para comprobar el funcionamiento del sistema de suscripciones, la utilización de servicios y, sobre todo, la manera de incluir nuevos agentes en el prototipo, bien sea proporcionando nuevos servicios que han utilizado otros agentes, como ha sido el caso del agente de planificación, bien utilizando servicios existentes para llevar a cabo sus propias responsabilidades, como ha sido el caso de la inclusión de un agente de modelado de estudiante sencillo.

Así pues, la implementación de este prototipo ha servido para poner a prueba los mecanismos de modificabilidad incluidos en la arquitectura, con resultados, por el momento, bastante satisfactorios.

Conclusiones del diseño de la arquitectura

El diseño de la arquitectura mostrada en el presente capítulo ha supuesto un trabajo dirigido a conseguir las propiedades de modificabilidad y eficiencia que se buscan para la arquitectura del sistema.

Para cada decisión de diseño que se ha tomado se ha prestado especial atención a tener en cuenta si servía a los fines descritos en los escenarios de calidad definidos al principio del capítulo.

Siendo muy específicos, este trabajo debió considerar desde el inicio un número manejable de elementos. De otro modo, resulta complicado administrar, asignar la responsabilidad adecuada a todos los elementos resultantes, definir sus interfaces y las relaciones entre ellos. Además, se hace necesario disponer de mucha información, para no tener que rehacer la descomposición demasiadas veces a medida que se realiza el diseño de la arquitectura, situación que se presentó, motivo creo yo de la inexperiencia y que resultó en mucho tiempo de más para llegar a la solución.

Adicionalmente, y una vez que el equipo comenzó a trabajar, nos dimos cuenta que realizar un diseño a través de una descomposición no resulta natural, o al menos no es la forma en la que los expertos y muchos otros diseñadores trabajan. La costumbre de diseñar de una manera iterativa e incremental hace que parezca más adecuado proceder de la misma manera en lo tocante al diseño de la arquitectura. De esta forma, en lugar de ir descomponiendo, la arquitectura se puede obtener por el proceso inverso, es decir, componiendo. Para ello, se puede comenzar diseñando una arquitectura muy pequeña, de dos o tres módulos, a la que poco a poco se le vayan añadiendo más módulos hasta que se hayan tenido en cuenta los mismos factores que en el caso de utilizar ADD. Pero esta consideración honestamente llegó muy tarde, de haberla sabido antes, nos hubiera ahorrado muchas horas de trabajo.

La conclusión respecto a este punto es que ADD nos lleva ligeramente atrás en el tiempo, a una época en la que era más habitual realizar descomposiciones. Sin embargo, con los sistemas que se desarrollan actualmente, es necesario incorporar otra visión al diseño de los sistemas, y en concreto a su arquitectura.

Aún queda otro punto a tener en cuenta respecto a los atributos de calidad para dirigir el diseño. Como se ha mencionado al principio del capítulo, no se ha tenido en cuenta para el diseño de la arquitectura ningún atributo de calidad relacionado con la usabilidad.

Esto es debido a que, para un STI, algunos de los que se consideran atributos de usabilidad constituyen funcionalidades propias de la aplicación y particularmente de la interacción con la interfaz. Algunas de las funciones más asociadas a este principio son, por ejemplo, la adaptación al usuario o la posibilidad de deshacer acciones. Estas funcionalidades pueden estar o no presentes en el STI, e incluso pueden activarse y desactivarse a voluntad, bajo el control de la estrategia de tutoría. Por eso, por el hecho de ser características que no se desea que estén siempre presentes, se ha preferido considerarlas sólo desde el punto de vista de la funcionalidad, y no como atributos de calidad del prototipo.

Sin embargo, queda abierta la posibilidad de utilizar atributos de usabilidad que dirijan también el diseño de la arquitectura, para comprobar si esto tiene un efecto beneficioso sobre el resultado.

Otra reflexión que se hizo patente muchas veces cuando los expertos evaluban o durante el diseño y documentación de la arquitectura está relacionada con la necesidad o la utilidad de introducir agentes en el diseño de la arquitectura. Por un lado, en (Bass y otros, 2003) se dice que el diseño de la arquitectura es independiente de las tecnologías utilizadas para implementar el prototipo. Por otro lado, en algunas de las revisiones realizadas sobre las publicaciones relacionadas con este trabajo se realizaba el mismo comentario, comentario que se hizo muchas veces patente en las reuniones de trabajo. Después de analizarlo desde varios puntos de vista, surgen distintos motivos que apoyan la inclusión de los agentes ya en esta parte del diseño.

El análisis de los sistemas que aparecen en el capítulo de estado de la cuestión, así como el contacto que se ha tenido con otros sistemas basados en agentes, ha puesto de manifiesto que muchos de estos sistemas se desarrollan de manera artesanal y sin aplicar principios de diseño adecuados, tanto en lo relativo a la arquitectura del sistema como al posterior diseño de bajo nivel.

Adicionalmente, en el proceso de revisión de artículos y materiales publicados recicientemente con la técnica de cocitación de autor, se deja entrever que algunas de estas experiencias sobretodo en el diseño en general, y el arquitectónico en particular, no reciben la suficiente atención por parte de la comunidad científica y por ende son proceso inacabados y experimentos fallidos, que sólo documentan una parte muy general del proceso.Desde el área de las arquitecturas software, esto se le atribuye al hecho de que rara vez se desarrollan sistemas que lleguen a un estado de producción, sino que simplemente se construye lo necesario para validar los resultados de una investigación, por lo que los sistemas desarrollados rara vez se convierten en prototipos.

Aunque los criterios de diseño proporcionados son puntos genéricos, constituyen la base que ha permitido dotar a la arquitectura presentada de los atributos de calidad deseados. Además, y sobre todo, se ha justificado de manera razonada por qué la utilización de estos aspectos, beneficia el diseño de la arquitectura propuesta, lo cual los convierte en herramientas útiles para desarrollar estos sistemas con la garantía, al menos, de que los atributos de calidad han sido considerados, y no simplemente dados por supuesto.

Finalmente, queda por hacer una última reflexión respecto al diseño obtenido. Como resultado de la búsqueda de modificabilidad en el prototipo se ha llegado a una solución en la que los agentes del STI interactuán a través de la solicitud de realización de unos determinados servicios. El resultado es similar a una arquitectura orientada a servicios obtenida mediante la aplicación de técnicas que intentan conseguir modificabilidad, y no como solución elegida *a priori*.

CAPÍTULO 6:

MODELO DE INTERFAZ GRÁFICA DE USUARIO

CAPÍTULO 6. Modelo de Interfaz Gráfica de Usuario. Metodología y Criterios para su desarrollo.

Como se ha mencionado la interfaz para un EV debe de facilitar todas las ayudas a los parcipantes para lograr la interacción y el acceso a las herramientas necesarias para la realización de las actividades de aprendizaje. A lo largo de esta investigación se ha hecho énfasis en que la interfaz debe cumplir con los niveles de usabilidad requeridos, para tal efecto, el diseñador de estos entornos necesita de una metodología, técnicas y procedimientos para ese propósito.

La representación de la información en un entorno de aprendizaje virtual puede ser muy variada. Desde un espacio basado fundamentalmente en texto hasta la incorporación de componentes de interacción multimedia o simuladores en tres dimensiones o con realidad virtual. Lo realmente importante es que el prototipo de organización de la información explicite el propósito instruccional del entorno e incorpore elementos hipermediales (mediante sonido, animaciones, vídeo) lo que otorga un papel más interactivo a los estudiantes.

Una parte importante de la metodología de esta investigación se enfoca, particularmente en este capitulo, hacia la identificación de los requerimientos característicos de un proceso de aprendizaje que se organiza en relación a un modelo y un propósito instruccional y que para lograr sus objetivos requiere de un sistema computacional que facilite sus tareas. Otra parte de este capítulo está orientada a obtener los requerimientos que soporten esta interacción es decir los componentes de interfaz, que ayudarán a realizar la tarea actividad en concreto.

El diseño de la interfaz de usuario es un proceso complejo que requiere comprender las tareas que el usuario debe realizar sobre el prototipo y las características de los diferentes usuarios que lo manejan. Esto implica que la actividad de diseño de interfaces de usuario es crucial en el proceso de desarrollo de las aplicaciones educativas.

Adicionamente para complementar este capítulo, se revisaron distintos modelos para el desarrollo de la interfaz (ANEXO: MODELOS DE INTERFAZ) con el objetivo de fundamentar el criterio de diseño propuesto para el diseño de la interfaz de nuestro prototipo para el aprendizaje. Partimos de algunas consideraciones referentes a esto modelos y de la idea básica que la interfaz de usuario: *es la parte del prototipo computacional que permite al usuario acceder a las facilidades que le otorga una computadora así como lo afirman* Dix, A., Finlay, J., Abowd & G., Beale, R.(1998).

6.1 Adaptación en las interfases de usuario

Existen distintas taxonomías que intentan clasificar la amplia variedad de posibles sistemas con algún grado de adaptación. Tradicionalmente se han considerado dos tipos de adaptación de la interfaz de usuario (Ben, 1993):

Adaptabilidad: en este tipo de adaptaciones el usuario realiza la adaptación. Por lo tanto es el usuario el que explícitamente adapta la interfaz de usuario para que se ajuste a sus gustos y características. Un ejemplo típico de este tipo de adaptación es la configuración del aspecto del escritorio en gestores de ventanas como el de Microsoft Windows, o KDE y GNOME en el sistema operativo Linux. Estos gestores de ventanas permiten al usuario cambiar los colores, fuentes, el fondo del escritorio o el comportamiento de algunos de sus componentes.

Adaptividad: cuando se da este tipo de adaptación, el prototipo es el actor responsable de realizar las acciones necesarias para realizar la adaptación. Un ejemplo de este tipo de adaptación es cuando durante la escritura de un documento en un procesador de texto, como por ejemplo Microsoft Word, la aplicación detecta un error gramatical y automáticamente lo marca o incluso lo corrige.

Sin embargo, dentro del concepto de adaptividad existe un amplio rango de combinaciones en las que los actores inmersos en la interacción (normalmente el sistema y el usuario) pueden tomar la iniciativa en las distintas etapas necesarias para la realización de una adaptación. De esta forma podríamos encontrarnos con que una adaptación no se realiza de forma automática sino semiautomática. Las etapas en las que los distintos actores pueden tomar la iniciativa son:

Iniciativa: uno de los actores involucrados en la interacción sugiere su intención de realizar una adaptación. Los actores principales en este caso suelen ser el usuario o el sistema.

Propuestas: si se detecta la necesidad de adaptación, será necesario proponer posibles adaptaciones que puedan ser aplicables dado el contexto de uso actual, para las necesidades detectadas. Una posible clasificación de los tipos de propuestas que se pueden dar sería:

1. Sugerir un cambio a otra plataforma y otra configuración del entorno (por ejemplo, en un cliente de correo electrónico, cuando el usuario indique su intención de alejarse de su PC el prototipo puede sugerir migrar el estado actual del cliente de correo a una plataforma móvil, como puede ser una tableta).
2. Sugerir el cambio a otro código ejecutable (por ejemplo, cuando el código actual de la aplicación no se pueda adaptar a los cambios que se han producido en el contexto).

3. Sugerir la ejecución de determinadas tareas (por ejemplo en un sistema de ayuda sensible al contexto).

Adaptar la interfaz de usuario manteniendo el mismo código ejecutable (por ejemplo, sería posible ocultar información no relevante para la tarea actual del usuario manteniendo el mismo código ejecutable).

Decisión: durante la fase anterior se sugieren una serie de adaptaciones plausibles. Sin embargo, normalmente no será posible la aplicación de todas las adaptaciones propuestas, sino que habrá que decidir cuáles son las mejores adaptaciones dada la situación actual.

Ejecución: finalmente, la adaptación o adaptaciones elegidas serán ejecutadas. Un factor importante cuando se realiza cualquier tipo de modificación a la interfaz de usuario sobre la que el usuario está actualmente interactuando es cómo se debe realizar la transición desde la interfaz de usuario original a la adaptada. Antes de la ejecución de una adaptación se suele ejecutar un prólogo para preparar la interfaz de usuario para la aplicación de la adaptación. Por ejemplo, si la adaptación incluye cambiar de un código a otro, la función de prólogo debería almacenar el estado actual de la aplicación, de forma que pueda ser reanudado tras la adaptación.

6.2 Elementos de la interfaz para potenciar la usabilidad, adaptación, adaptabilidad y control del prototipo.

“La interfaz vuelve accesible el carácter instrumental de los objetos y el contenido comunicativo de la información”. (Bonsiepe, 1998). El diseño se dirige hacia la interacción entre el usuario y el artefacto. El dominio del diseño es el dominio de la interfaz. La interfaz hace posible la acción eficaz para una determinada tarea, siendo el tema principal del diseño como lo dice Bonsiepe, (1998).

Considerando que la interfaz gráfica, exige por parte del usuario, una serie de condicionantes fisiológicas, y necesita del uso de dispositivos que permitan poner en contacto al sujeto con el prototipo tecnológico; llamados dispositivos de interfaz humano, como el ratón o el teclado, que permiten a través de las posibilidades fisiológicas del sujeto, producir parte de la interacción con la interfaz y por lo tanto parte fundamental de la misma, entonces estas condicionantes son parte de los elementos a tomar en cuenta al momento de proponer cualquier tipo de interfaz.

Se trata de un proceso mediante el cual, un sujeto, se acerca a un prototipo tecnológico con el que interacciona a través de los signos inscritos en dicha superficie; es decir, un proceso

interactivo, que requiere de una serie de requisitos cognitivos básicos por parte del sujeto, como percibir, decodificar, memorizar, decidir y navegar a través de la interfaz gráfica. Por lo tanto, la interfaz sólo cobraría sentido, cuanto el sujeto es capaz de comprender el significado y el proceso de interacción, y sus facultades cognitivas son capaces de interpretar adecuadamente los signos que se producen sobre la interfaz y usarlas adecuadamente.

Scolari (2004) plantea la necesidad de hablar de las metáforas de la interfaz y no de las definiciones de la misma. Las metáforas son importantes a la hora de comprender la realidad que nos rodea, siendo muy útiles al momento de aprender un nuevo concepto.

Definir un término abstracto e invisible puede ser una tarea tediosa tanto para quien lo defina como para quien lo tiene que interpretar y comprender. Es por ello que este autor, al referirse a las interfaces, lo hará en términos metafóricos para poder facilitar la comprensión de aquellas. El poder descriptivo de cada metáfora ayudará a reconocer los rasgos distintivos de las interacciones.

Scolari (2004) nos plantea la existencia de cuatro tipos de metáforas:

1. Metáfora conversacional (Interface como diálogo persona-ordenador): una de las concepciones más difundidas; según esta metáfora los seres humanos y las computadoras son considerados como socios de un diálogo. Ambas partes (persona y computadora) actúan como emisores y receptores simultáneamente. La conversación se llevaba adelante, básicamente, sobre sistemas alfanuméricos.
2. Metáfora instrumental (Interface como extensión o prótesis del cuerpo del usuario): La superación de los sistemas alfanuméricos se dió a través de la aparición de interfaces gráficas *user-friendly o amigables con el usuario*, esto es, entornos gráficos denominados WIMP (Windows, Icons, Mouse, Pointer) que se impusieron desde 1984. Los objetos interactivos logrados a través de estas interfaces fomentaron la idea de manipulación directa de los objetos ubicados en la pantalla como si se trataran de herramientas tangibles.
3. Metáfora superficial (Interface como superficie osmótica que separa y permite el intercambio hombre - computadora): Existe para muchos una concepción bastante arraigada que consideran al diseño (en particular, diseño de interface) un proceso cosmético, como algo que “acompañaba” al producto o servicio principal.
4. Metáfora espacial (Interface como entorno de interacción hombre - computadora): Esta metáfora considera a la interfaz como el espacio en donde toman lugar las interacciones entre un usuario, una acción o finalidad y un artefacto o utensilio.

De estos 4 tipos se han considerado tanto el primero como el último como los tipos más importantes de metáfora a utilizar en el modelado de la interfaz del prototipo que nos ocupa, el

agente y las herramientas de autor.
Y para facilitar el proceso entre usuarios y sistemas UI (Interfaz grafica de usuario), se aplica “la metáfora de escritorio”, que consiste en representar recursos, elementos y funciones del prototipo como ficheros, datos y archivos, a través de íconos sobre los cuales es posible el asumir de una forma virtual, la relación de trabajador en el entorno de la oficina.

Esta es la comparación más global y primaria de las que gobierna la interfaz gráfica de usuario. El escritorio es la primera metáfora, representa el espacio de trabajo donde se manipula, se mueve, y organiza la información. Con base en la metáfora del espacio-escritorio se desarrollan el resto de las comparaciones, como son las carpetas, los documentos, las herramientas, lápices y tinteros.

Los elementos de la interfaz suponen dentro de los procesos interactivos, elementos simbólicos que están inscritos en los lenguajes visuales que operan en los sistemas de comunicación de los humanos. Desde esta perspectiva, la interfaz ha generado su propia gramática de representación e interacción, suponiendo actualmente un modelo que debe ser aprendido por cualquier persona dispuesta a intercambiar información con un sistema binario.

La clave está en la consistencia en el diseño, como proceso mediante el cual se establece a la hora de estructurar la información como elementos de navegación en la interfaz, con un orden común y coherente; sea esta para ser visualizada desde un computador o desde un dispositivo móvil. De este modo, el usuario sólo tiene que aprender una sola vez donde localizar las acciones en los menús, y aunque se produzca un cambio en la aplicación, sepa localizarlos sin problemas, si se maneja el mismo concepto. La consistencia en el diseño de interfaces, es un elemento muy importante porque reduce la curva de aprendizaje del prototipo por parte del usuario.

Junto a este apartado, se presentan las ventanas, los iconos y los menús, son elementos interactivos, que pertenecen a la parte simbólico-lingüística de la interfaz. Pero también cabe mencionar que para que estas funciones se lleven a cabo se encuentra la interfaz humana o física de la interfaz gráfica.

Los ***íconos,*** a diferencia de los *símbolos* establecen una conexión material mucho más fuerte con la realidad visual, pues deben representar en ellos su consistencia y reconocimiento cultural. Son manifestaciones y concepciones de representación comunes, donde influyen muchas variables a la hora de conceptualizar su significado; desde los recuerdos que cada persona tenga, hasta el proceso que se relaciona con una imagen básica o universal, por ejemplo: al oír la palabra “cruz”, uno de inmediato se imagina dos líneas perpendiculares o simplemente las letras que conforman la palabra cruz, todo esto depende del receptor y su relación cultural con el objeto.

La imagen iconográfica (analógica) permite al ser humano apropiarse de la realidad circundante por medio de la observación y reconocimiento cultural estableciendo un puente entre lo que vemos y lo que constituye nuestro pensamiento en sociedad, existe una configuración simbólica con el uso que le damos al objeto y lo que comunica, pudiendo sustituir o traducir una realidad.

Podemos identificar algunas formas de observar el icono dependiendo de sus referencias simbólicas, relacionando algunos puntos:

La imagen visual a la que nos remite.

Su representación cultural, ya sea por medio de concepciones lingüísticas como símbolo pre-lógico (imagenmental).

La definiciones sintácticas-semánticas-conceptuales.

Lo connotativo y denotativo.

Los valores de la imagen (histórica y posición social).

La función de la imagen (simbólica, epistémica, estética).

Sus grados de iconicidad, significación y representación.

La “aprehensión” del símbolo llevado al icono, la posesión del significado en los paradigmas culturales.

Con el uso de nuevas tecnologías, la concepción icónica se ha manifestado diferente a lo que habíamos estudiado en la universidad, hoy podemos hablar de la comunicación más allá del símbolo, actuando como metalenguajes que conjugan muchos referentes y significaciones, construyendo y modelando nuevas maneras de representar el ícono, llevándonos al pensamiento visual y a lo virtual como constructor de realidades dinámicas hacia la lingüística cognitiva.

Cuando el objeto deja de ser *lo que es*, se convierte en una adaptación dinámica de lo real, extrapolando el símbolo hacia lo significante, diferenciando el pensamiento visual desde los paradigmas hacia la realidad virtual. La llegada de los metalenguajes digitales nos llevan a *extrapolar la dimensión del ícono* y reformular su estado material, centrándonos en el pensamiento y la percepción, aplicando la imagen real al nivel de relación significativa de sus referentes con el medio digital por el uso de las nuevas tecnologías de información.

La construcción de un ícono es un proceso histórico, que debe complementarse a la inserción y adaptación del medio social en el que está, que va modificando su interrelación y lugar que ocupa en la función de la imagen.

Las representaciones culturales que impregnan de sentido las cosas, configuran la esencia simbólica en variables gráficas que constituyen el referente cultural de los íconos, formando parte de nuestra vida diaria y son la manera de simplificar el pensamiento abstracto; modelan y cambian, sustituyen un paradigma y lo llevan a su representación material y adquieren una carga social histórica. Las adaptaciones residuales operan desde la mutabilidad del contexto icónico hasta la realidad modeladora, guían y completan la imagen visual cargándola de sentido y aprehensión social (Chang, D., Dooley L.,&Tuovinen E. J., 2001).

Una de las características fundamentales de todo lenguaje icónico es su grado de isomorfismo (similitud de forma) en relación a las características de los referentes, dicho grado varía a lo largo de una amplia escala:

> «Una gama continua de formas va de los medios menos isomórficos a los que lo son más; incluye elementos intermedios como los sonidos onomatopéyicos del lenguaje, los ideogramas, las alegorías y otros símbolos convencionales.»

Es ese grado de isomorfismo el que explica, en principio, la mayor universalidad de los lenguajes icónicos respecto al lenguaje hablado, mucho más arbitrario y por consiguiente más sujeto a convenciones. Aunque en este trabajo defenderemos que la cuestión del parecido o similitud está también sujeta a convenciones y que, por lo tanto, la iconicidad, como nos lo advierte Joan Costa, es en buena medida resultado de la subjetividad y la personalidad creativa de quienes manejan las técnicas de producción icónica y de cómo las usan, lo que nos parece fuera de toda duda es que existe una relación objetivable, mensurable, entre determinados tipos de imágenes y sus referentes:

> «Una fotografía en color es más icónica que una fotografía en blanco y negro; un retrato es más icónico que una caricatura; un mapa o el plano de una ciudad son menos icónicos que una fotografía aérea; un esquema, un diagrama o un organigrama apenas son ¡cónicos de aquello que representan; una fórmula química o matemática o una página escrita son todavía menos ¡cónicos, menos semejantes a lo que representan -grado cero de iconicidad- (...)»

Considerando estos conceptos, se ha propuesto como aportación significativa una matriz de diseño para el sistema icónico del EVEATI y las heramientas de autor. Su diseño se basa en una interpretación sintáctico-simbólica referenciada no sólo a la iconocidad o nivel de iconicidad

que debe representar cada ícono o herramienta, sino a lo que yo llamo los “usos y costumbres de la interfaz” porque nos hemos habituado o quizá forzado a asociar íconos con funciones y no podemos disociar esa relación.

6.3 Recomendaciones para el diseño, modelado e implementación de la interfaz

Para el modelado e implementación de la interfaz del EVEATI, se partió de la revisión y selección de normas y estándares de calidad en relación al aspecto, funcionamiento, interpretación, simulación y función que debe cumplir la interfaz general, considerándose también que al final se debe realizar la evaluación.

Para llevar a cabo ésta evaluación ha de enunciarse previamente un conjunto de *criterios o indicadores* que contribuya a determinar la *calidad* de los entornos de formación “on-line”.

En este sentido, alrededor del término evaluación de la calidad se retomaron las *normas y estándares*, como ISO 9241, UNE 139801 y UNE 139802, etc, que guían el proceso de valoración de todos aquellos aspectos técnicos y estéticos que condicionan la eficacia de un entorno virtual, pero adicionalmente también se han planteado criterios particulares desde la perspectiva del diseño.

En el ámbito empresarial todas estas normativas parecen que marcan un estilo, pero en el caso de entornos académicos o didácticos, pareciera por el contrario que estos elementos no son tomados en cuenta, puesto que en realidad no hay manera de condicionar o limitar la publicación o bien condicionar que se activen día con día. Generalmente y esto dicho con todo respeto, los que diseñan los cursos, ambientes, etornos y demás sistemas para el aprendizaje o entrenamiento en línea, no cuentan con la preparación en el campo de la ergonomía visual y el diseño gráfico, como para implementar o considerar todos estos estándares, por lo que también como parte de la intenciones de esta propuesta es tratar de proporcionar una serie de elementos, indicadores y requerimientos mínimos a considerar para el diseño.

6.3.1 Diseño General de la interfaz

La interfaz de usuario está condicionada como se refirió en los capítulos previos, por una serie de parámetros, tales como el tipo de información que se presenta, el sistema de interacción, los recursos hipermedia que se combinan, etc., considerando las posturas de Díaz, Catenazzi y Aedo (1996). El objetivo de la etapa de diseño es expresar los resultados abstractos del análisis, en especificaciones computacionales tangibles. Si el modelo conceptual del usuario es considerado

por el diseñador, entonces podrá definir para la interfaz, un claro mapeo entre las tareas y los objetos semánticos con los representados en la pantalla, y permitirá que el conocimiento requerido al usuario para utilizar el prototipo, sea lo más mínimo posible.

Más aún, se propone trabajar no sólo con el modelo del usuario sino con los demás estudios realizados, tanto sobre el ambiente como sobre la aplicación, para de esta forma lograr soluciones a nivel de interacción muchas más adecuadas y coherentes.

Sabiendo que, por lo general, no se podrá contar con especificaciones a nivel de interfaz completas, entonces en esta fase, el diseño partió del proceso de análisis, integración, refinación, conjugación de los modelos que se analizaron en la etapa previa, en pos de lograr una aproximación inicial de la misma.

La interfaz, especificada a través de diferentes medios como son los prototipos y escenarios, expresa las nociones extraídas de dichos modelos e integradas en el modelo propuesto,para así permitir la evaluación por parte de los usuarios, de la usabilidad y legibilidad del diseño.

La prototipación o diseño del prototipo, permite desarrollar una visión prematura del sistema, donde se puntualiza en un principio, la visualización, la interacción y el comportamiento global del mismo, abstrayéndose de las características computacionales y proveyendo un marco para que los futuros usuarios del sistema puedan observar el comportamiento del mismo e interactuar con él antes de que sea finalmente desarrollado.

Para el modelado de la interfaz se parte de una estructura que se presenta en la Tabla 6. 1 y se han generado los prototipos funcionales tanto de tipo MockUps, como escenarios y Herramientas software. A continuación se describen las cualidades de los 3 niveles de prototipado y sus función en la propuesta.

1. MockUps: son prototipos muy sencillos que únicamente describen interacciones particulares con el sistema. Permiten implementar un escenario, una situación particular de una interacción o un diálogo con el sistema. Consiste de una serie de diseños de pantallas presentadas al, a través de las cuáles son analizadas y evaluadas heurísticamente. Se utilizaron tanto para la prueba de usuario, como para la evaluación heurística.
2. Escenarios: Pueden ser impresos en papel o en video, describen la situación actual de un proceso de trabajo. Se utilizaron en las entrevistas realizadas a los expertos. Para el caso que nos ocupa, directamente se utlizaron materiales impresos de la interfaz. Se trata de una sucesión de pantallas estáticas en papel.
3. Herramientas de software: Basadas en una técnica de representación formal, definieron la versión ejecutable del sistema. Se trató literalmente del prototipo o célula funcional del

sistema que se evaluó con los alumnos y se presentó a los expertos. Se consideró como una especificación ejecutable del diseño. Sirvió para describir el comportamiento y los aspectos dinámicos de la interfaz. Representa la interacción, el control del diálogo y el estilo del diálogo.

Etapa	Consideraciones
Análisis Funcional del Sistema	El primer paso para el desarrollo de la interfaz del usuario comenzó con el análisis las características de los componente funcionales. Se debe tener claro conocimiento de los requerimientos funcionales, el objetivo del sistema de software y las características que se pretenden de la aplicación. Hay que tener en cuenta, se trata de diseñar la interfaz de un prototipo educativo, con todas sus particularidades, por lo que fue neceario primero determinar la arquitectura y todas las funciones y componentes que requiere, para luego determinar la interfaz.
Análisis del Usuario	Es fundamental que el diseñador conozca al usuario e investigue sus características individuales, si pretende brindarle una interfaz adecuada a sus necesidades. Esta etapa, constituye la esencia del Diseño Centrado en el Usuario. Es importante estudiar la clase de usuarios que van a interactuar con el prototipo y categorizar a los mismos. Se debe considerar si son usuarios novatos o con experiencia, si han utilizado computadoras con anterioridad o no, qué edad tienen,etc. por eso se realizó un diagnóstico preliminar y antes de involucrar al usuario con el prototipo se hizo un sondeo sobre consideraciones demográficas y el uso de las TICs.
Investigación del Ambiente:	El ambiente o contexto, que rodeará al usuario mientras interactúa con el prototipo, también debe ser estudiado cuidadosamente y afectará notablemente el diseño de la interfaz. Se determinaron las siguientes condiciones: 1.- Características del ambiente físico: Hay que realizar un modelo del ambiente, estudiar el lugar físico donde se va a montar el prototipo y analizar el grado de concurrencia, ruido, interferencias, nivel de distracción, y cuestiones de luminosidad, principalmente. 2.- Características de la Organización y del Entorno Grupal: modelar al usuario dentro de un marco organizativo, es decir, estudiar su posición, su rol en la institución, sus responsabilidades, estudiar el grupo que lo rodea, su formación, el nivel de conocimiento de los demás. 3.- Características Técnicas: se debe investigar las características técnicas y los recursos que el o los usuarios cuenten en el momento que se instale el prototipo, además, inventariar el tipo de máquinas que poseen, el tipo de monitor, resoluciones posibles, tipos de conexiones que tienen y demás aspectos tecnológicos y finalmente considerar los dispositivos de hardware que se van a destinar para la interacción hombre-computadora.

Determinar los componentes de la interfaz de usuario a partir del cumplimiento de objetivos	La interfaz del usuario debe cumplir con los objetivos de: Simplicidad Confiabilidad Flexibilidad Transparencia Ergonomía
Decisiones de Diseño:	Definir cada una de las decisiones de Diseño que se deben poner en práctica
a) Determinar el estilo de interacción	Para el diseño que nos ocupa se determino que fuera mixto, osea : de diálogo secuencial, con manipulación directa y basado en comandos.
b) Decidir sobre el tipo de interfaz a proveer	Se trata de una interfaz con características híbridas. que brindará aspectos de adaptación, evolución, y/o de inteligencia.
c) Decidir cuestiones de visualización	Se decidió sobre los aspectos de las pantallas y representación de la información, de los estados, de las entradas y salidas, objetos de interacción y ayudas.
d) Especificar cuestiones de comportamiento	Se planifició el comportamiento de los objetos, el feedback, el control de la interacción, el sistema de asistencia, el manejo de errores. También se planificó la adaptación que se desee proveer o cuestiones no tradicionales como aspectos inteligentes, de inferencia, de evolución. También se planeó, la relación de la interfaz con la aplicación, definiendo formas de integración e invocación de funciones.
e) Diseñar las funcionalidades y servicios a nivel de interfaz	Todos los aspectos dinámicos y comportamiento de la interfaz que son básico para la estructura mínima se deben considerar y programar en el prototipo.

Tabla 6.1 Metodología para el diseño de la interfaz y desarrollo de prototipos.

a)Diseño de las pantallas

Requerimientos e Indicadores de calidad:
El diseño de las pantallas deberá ser amigable, presentando una interfaz atractiva, que contribuya a que el aprendiz se sienta cómodo con el formato visual a través del cual se presenta la información.

En este sentido, tan importante es la calidad de la información que se presenta en el EVEATI, como la forma de organizarla y de presentarla. El propio formato visual del prototipo será el que propicie la interacción del aprendiz con dicho entorno, el cual condicionará en gran medida el que éste capte y posteriormente asimile aquella información que le resulte más atractiva visualmente.

Al igual que en la enseñanza presencial muchos aprendices “on-line” ven su proceso formativo condicionado por el entorno virtual en el que se encuentran inmersos, provocando, en algunas ocasiones, el abandono de la acción instructiva al no sentirse cómodos con el formato visual

del prototipo, por este motivo, es de máxima relevancia atender a las características estéticas y técnicas en un EVEATI.

Como parte de la metodología de diseño se desarrollaron las plantillas que regirán el desarrollo de la interfaz. Como primera actividad se definieron formalmente la tipología de las pantallas, esto es, se estableció la cantidad de clases de pantallas que se desarrollaron (mientras menor el número es mejor), para posteriormente generar una plantilla general para cada una de ellas. En estas plantillas se consideraron como constantes de diseño:

Ubicación del título de la pantalla y logotipo.
Ubicación de las alarmas del proceso
Ubicación de funciones genéricas, tales como confirmación de alarmas
Considerar al centro de la pantalla como el espacio de más alta visibilidad
La información miscelánea o complementaria debe ir abajo a la izquierda

Así, para el prototipo DECANO se ha diseñado una interfaz gráfica amigable y sencilla que facilita la incursión del aprendiz en el entorno, como podemos observar en las figuras 6.1, 6.2 y 6.3. Asimismo, tal como se refiere en la figura 6.4, el diseño se ha basado en medidas y retículas estandarizadas, que ya han sido probadas con eficacia puesto que se estructuran en formatos y medidas universalmente reconocidas. Inicialmente se parte de un formato horizontal de dispositivo, considerando como medida física estandarizada o más popularizado a 1024 px de ancho por 768 pixeles de alto (Figura 6.5).

Figura 6.1: Interfaz de entrada al EV del DECANO

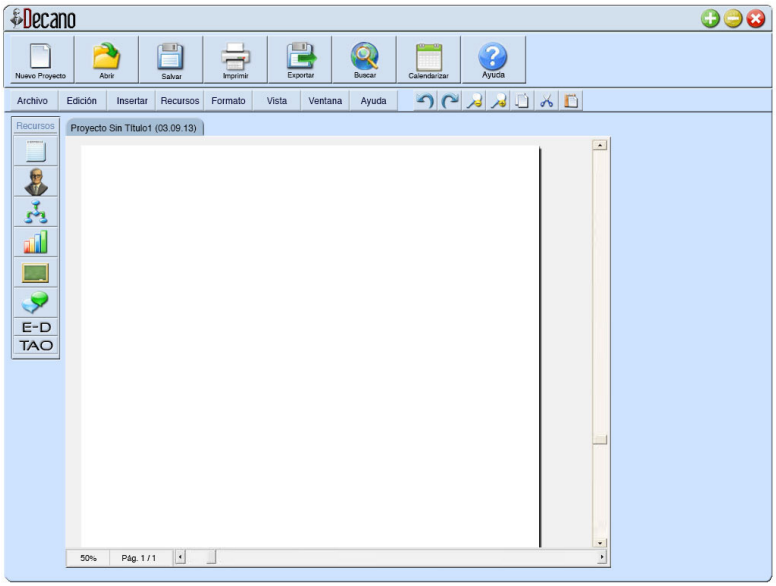


Figura 6.2: Interfaz de la Ventana Principal del Prototipo

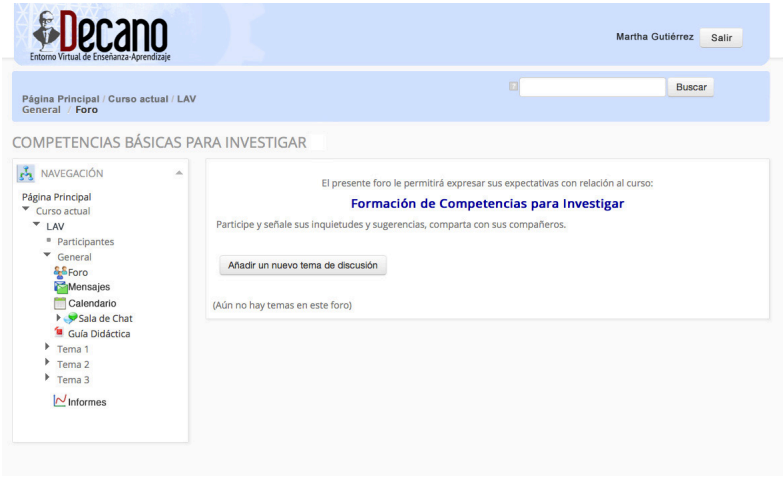


Figura 6.3: Interfaz de la Sección General del Entorno Vitual

Para la construcción del layout se tuvo que establecer la anatomía y jerarquías de los elementos, de esta forma el prototipo funciona como un conector entre el usuario y el contenido tal y como lo refiere Beaird (2007).

Antes de designar los espacios en el layout, se determinó la retícula o wireframe[1] que en diseño sirve como sistema de organización que facilita la organización significativa de una superficie o de un espacio, como lo plantea Müller-Brockmann (1982). Las dimensiones y longitud dependen del tamaño de los monitores para aprovechar la “zona segura”, como sugieren Lynch y Horton,

1 Término utilizado para el esquema de una página, es una guía visual que representa el esqueleto o estructura visual. (Brown, 2011)

(2000) de modo que la retícula seha creado en función de la medida estándar arriba mencionada de 1024×768 pixeles.

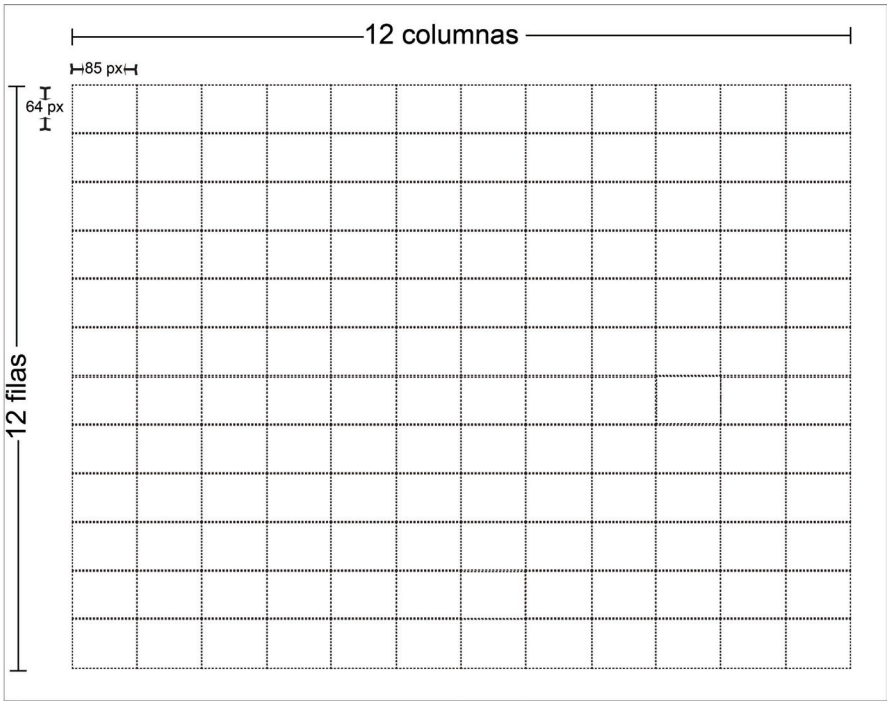


Figura 6.4: La retícula se encuentra compuesta por unidades de 85 x 64px y la altura designada para los botones de navegación, celdas de búsqueda y cuadros de texto es de 32 px.

La construcción de la retícula se dividió en 12 columnas y 12 filas (Figura 6.3) que es un estándar recomendable para la división a partir de columnas (Müller-Brockmann, 1982).

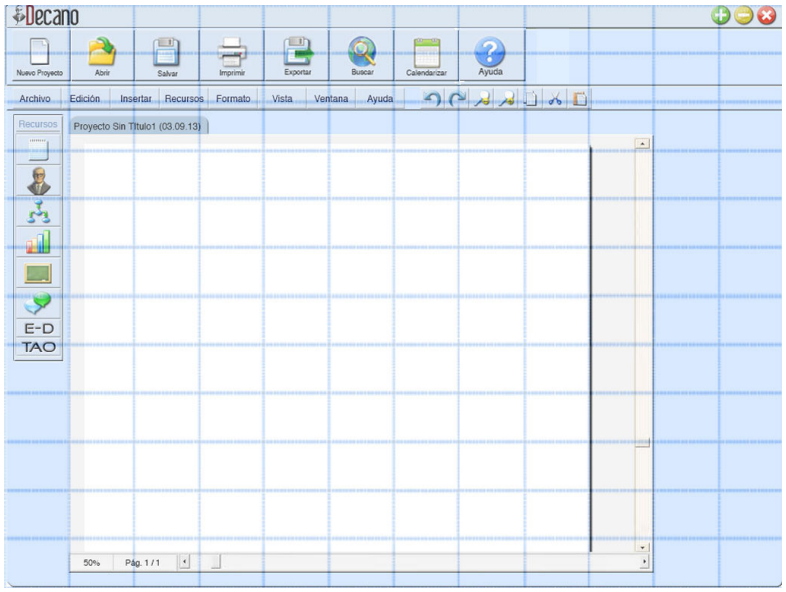


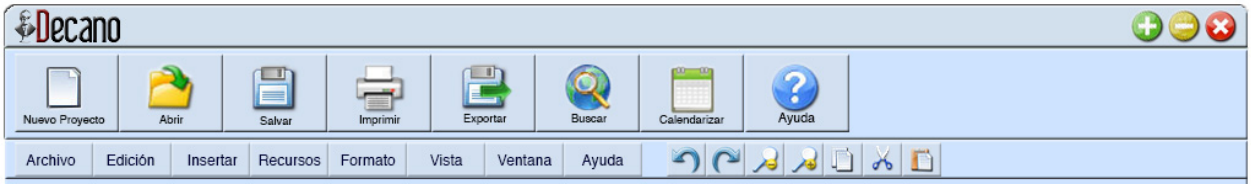
Figura 6.5: Layout gráfico del prototipo sobre la retícula elegida.

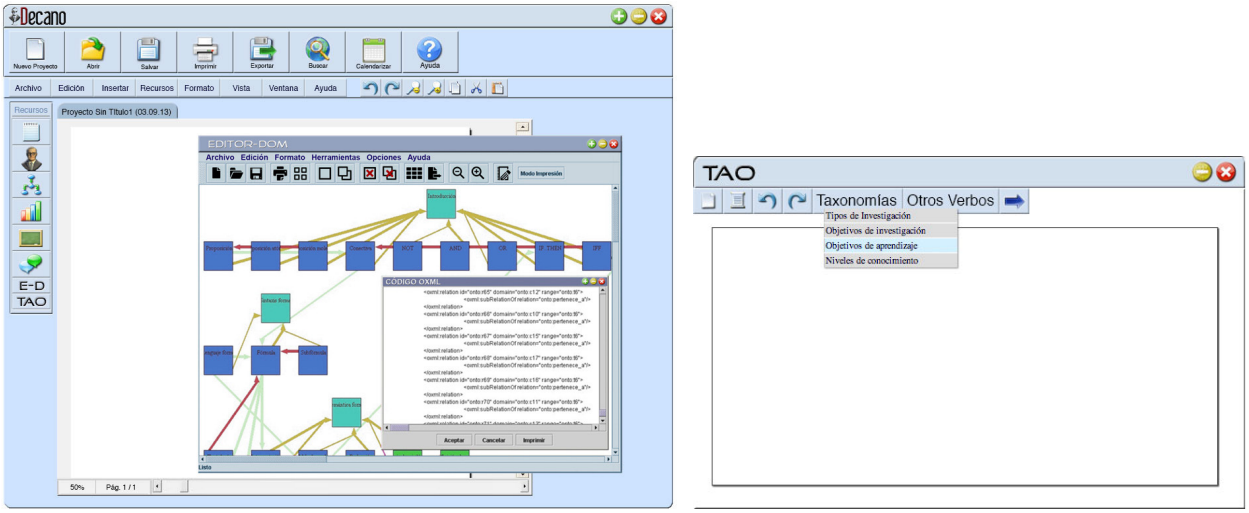
b) La gama cromática empleada

Indicador de calidad
Emplear una gama cromática que facilite la percepción de los contenidos, evitando el uso de demasiados colores para elementos básicos que pueden dificultar la lectura de la información. Además se elegirán colores neutros principalmente para menús y fondos de interfaces génericas, tanto la del prototipo, como las de las herramientas de autor, y sólo los íconos considerarán una mayor cantidad de gamas tonales puesto que se trata de elementos descriptivos y asociativos para las funciones.

Crear entornos cómodos y fácilmente navegables, favorece y facilita el proceso de enseñanza–aprendizaje, para ello, es necesario utilizar una gama cromática que contribuya a generar estos ambientes. Así, en el proceso de diseño de contenidos formativos o, más concretamente, de asignaturas para red, es conveniente emplear colores en tonos claros, tanto en el fondo de pantallas como en el diseño de los diferentes esquemas, y contrastar con tipografía en tonos obscuros, para subrayar el contraste con el fondo, contribuyendo con ello a facilitar la visualización y asimilación de la información mostrada.

Por otro lado, la utilización del color en los Entornos Virtuales de Enseñanza-Aprendizaje no sólo tiene una función estética, sino que también puede emplearse como vehículo para facilitar la navegación al diferenciar módulos o lecciones de contenido según la gama cromática presentada y siempre adoptando el criterio de homogeneidad para no hacer perder la orientación al estudiante o usuario.





Figuras 6.6, 6.7, 6.8. Utilización del color para menús y contenidos en el prototipo DECANO y las herramientas de autor. El color sirve también para diferenciar la interfaz principal y las herramientas de autor. La gama de los fondo, tanto el general como el de botones y herramientas es neutro y el color se aplica en funcion de los íconos.

En este sentido, se han delimitado los módulos de contenido mediante el uso de diferentes colores con el fin de identificar y diferenciar cada uno de ellos. En las figuras 6.5, 6.6 y 6.7, podemos observar como el primer bloque didáctico emplea el azul como color representativo, donde todos los elementos asociados a éste, tales como componentes del submenú, ventanas emergentes, etc, aparecen identificados con esta tonalidad, siguiendo con el criterio de homogeneidad. Lo mismo sucede con elementos asociativos como los botones de ventana representados e identificados con los colores rojo, verde y amarillo, en asociación a los colores de la señal de tránsito y que en simulaciones digitales se han aprendido a reconocer como cerrar, abrir o crecer y minimizar, para el caso de los botones de ventana.

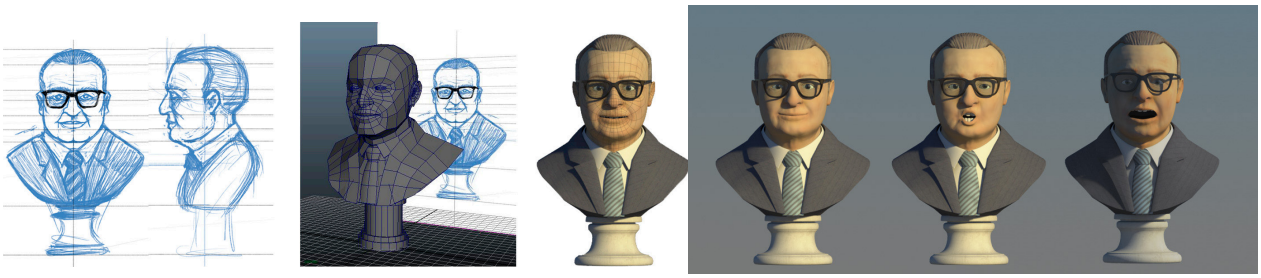
c) La presentación de texto, imágenes, sonido, etc.

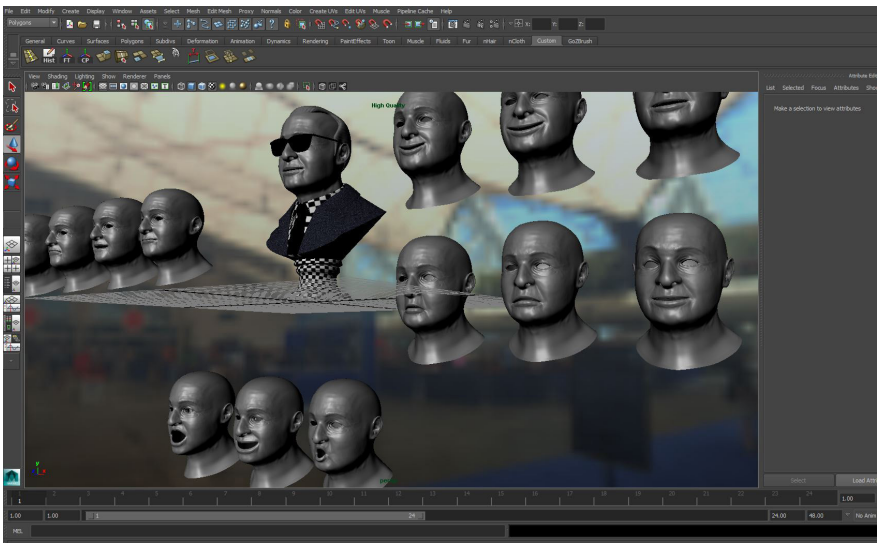
Requerimientos e indicadores de calidad:
A lo largo de la aplicación no se deberá abusar en la presentación de los contenidos didácticos de imágenes, documentos textuales o sonoros excesivamente voluminosos, etc., porque ralentizan su presentación.

Una incorporación excesiva de imágenes, vídeos, sonidos, etc., puede contribuir negativamente en el aprendizaje y convertirse en elementos distractores, de ahí que la selección que se haga deba atender a requerimientos didácticos concretos. En el caso que nos ocupa, se han proyectado bibliotecas de recursos y repositorios que irán adosados a las herramientas de autor y recursos y

sólo se incluirán o activirán a demanda del estudiante.
Igualmente, la inclusión en un EVEATI de numerosos elementos multimedia provoca un incremento considerable de la dimensión y volumen del material didáctico, lo cual puede dificultar su consulta y visionado, sobre todo en aquellas computadoras o dispositivos cuyas conexiones a Internet sean muy lentas.

En numerosas ocasiones, para reducir el tamaño del material se opta por disminuir la calidad de los diferentes recursos multimedia. Con ello, únicamente se propicia que los aprendices manejen contenidos carentes de calidad técnica provocando confusiones debido a la escasa nitidez y definición del vídeo, del audio o de las imágenes incorporadas. En nuestro caso, se piensa más en la posiblidad de que el aprendiz pueda interactuar con la web de manera tal que si requiere elementos más específicos o necesita otros recursos los pueda adquirir a través de ese medio. Podemos inferir que al momento de la evaluación, el prototipo tiene cubiertas tanto las simulaciones, imágenes y demostraciones mínimas necesarias para la temática que nos ocupa. Y particularmente para el caso del tutor, ahí es dónde está la mayor cantidad de posibilidades multimedia, porque al ser un reactivo, se han generado un gran volumen de posibles respuestas, así como actitudes, gestos y demás elementos que permitan hacerlo empático y lo más parecido a un tutor real y logre cumplir con su misión (Figuras 6.9, 6.10 y 6.11).





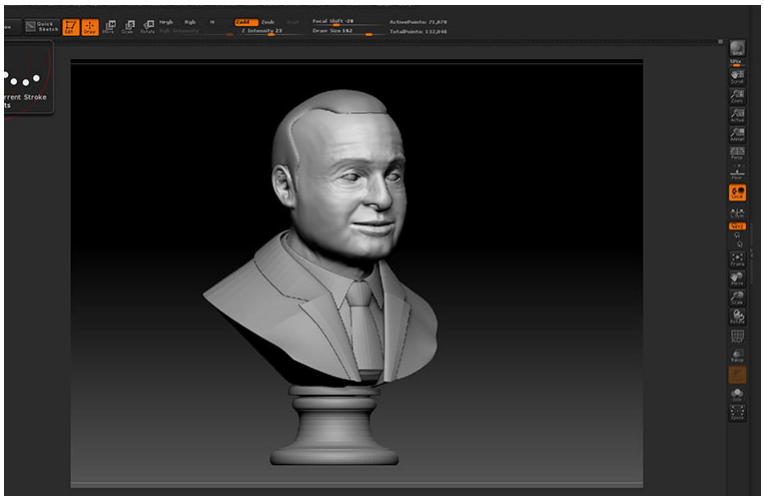


Figura 6.9 y 6.10 y 6.11: Proceso de creación del tutor y modelado del mismo, así como las expresiones necesarias para las respuestas registradas en el prototipo.

d) Íconos y mapas de navegación

Requerimientos e indicadores de calidad:
Tanto para la ventana principal del prototipo, como para todas las herramientas de autor y las pantallas de inicio y presentación, así como para el repositorio de datos se han desarrollado matrices y un sistema de íconos consistente, homogéneo, semejante, basado en directrices que permite su crecimiento y genera una metáfora visual fácil de interpretar por los usuarios.

La estructuración de los contenidos didácticos facilita, sin duda, la exploración de la información, sin embargo, es necesario que ésta se encuentre organizada y se pueda llegar a ella a través de mecanismos de navegación eficaces y accesibles, que faciliten el del usuario por el entorno virtual.

Para que el estudiante sea capaz de interiorizar la organización interna del prototipo, se debe diseñar mapas e iconos que permitan un acceso rápido a los diferentes contenidos, empleando en todo momento una terminología y recursos gráficos semejantes, de forma que se propicie la comprensión de la misma como sugieren Díaz, Catenazzi y Aedo (1996).

Del mismo modo, debemos ofrecer un mapa de navegación intuitivo que facilite al aprendiz conocer en todo momento dónde se encuentra, dónde ha estado y hacia dónde puede dirigirse

(Nielsen, 2002).

En este sentido, resulta muy útil introducir en el EVEATI un árbol de contenidos a través del cual el discente pueda analizar la información presentada, dejando constancia de los elementos, la situación actual y, por supuesto, todos aquellos contenidos que aún no ha examinado. El árbol de contenidos incorporado en el DECANO, aun no ha sido elaborado con la totalidad de los elementos pensados para el prototipo, presenta algunos de los elementos y herramientas de autor y sus recursos, con el fin de atender a estos requerimientos de navegación y en el se expone tanto la el entorno web, como la herrramienta central del prototipo y las demás herramientas de autor.

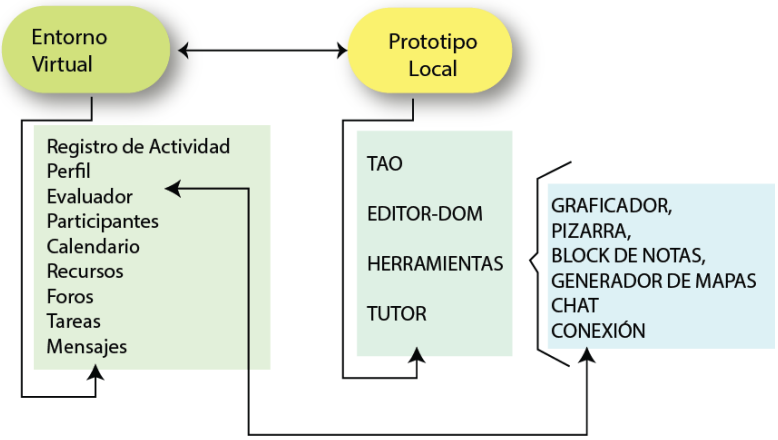


Figura 6.12: Árbol de contenidos del DECANO

En cuanto al sistema de íconos, se ha elaborado una matriz igual que la presentada previamente dentro de este capítulo, con el fin de respetar los principios y requerimientos planteados para estos elementos. Como se mencionó ya, los íconos en el contexto de las interfaces gráficas son signos esquemáticos que representan algún tipo de fichero, carpeta, aplicación, o dispositivos de un sistema informático.

Los íconos, tal cual se ha defendido en el marco teórico de este trabajo, son signos interactivos y por lo tanto inscritos en una gramática especial que debe ser aprendida por el usuario. Un ícono es un signo que mantiene una relación de semejanza con el objeto representado y por tanto, resulta generalmente fácil de entender. Por ejemplo, casi cualquier persona, incluso los niños, entenderían imágenes como las que incluye la Figura 6.8. Sin embargo, para entender los iconos, incluso los más sencillos, hace falta un cierto entrenamiento o bien cierto acondicionamiento para lograr establecer esa relación de asociación o semejanza.

Figura 6.13: Ejemplo de íconos del Prototipo DECANO

Podemos partir de que todo cuanto vemos a nuestro alrededor contiene un valor icónico intrínseco, una representación mental capaz de ser identificada en diferentes contextos. Los iconos son sustitutos visuales de un concepto, idea o acción, siendo perfectas metáforas que ilustran el mundo real.

En el ámbito que nos ocupa, los iconos son utilizados para potenciar el atractivo y entendimiento de los contenidos de la página, aplicación o software, convirtiéndose en una guía visual para el lector.

Para conseguir la representación icónica de elementos y funcionalidaes, se plantearon los cuestionamientos que propone Susan Kare[2] para realizar íconos.

> ¿Qué imágenes son literales y cuáles se pueden beneficiar de una metáfora?
>
> ¿Cómo funcionarán en su conjunto?
>
> ¿Qué símbolos existentes pueden servir o son aceptados universalmente?
>
> ¿Cómo se pueden evitar yuxtaposiciones extrañas de iconos de teléfono en un teléfono, o de auriculares tradicionales en móviles? (Kare, 2011)

Respondiendo a estos cuestionamientos, los íconos representados por metáforas son los que corresponden al sistema general y las herramientas de autor. Estos son fenómenos abstractos de los cuales no se tiene una experiencia sensible, física y palpable a través de los cinco sentidos. Más adelante se complementa este aspecto con la explicación de la matriz para su construcción.

Para valorar todos estos aspectos en el diseño de un entorno virtual de aprendizaje, se han especificado un conjunto de indicadores de calidad en torno a tres puntos:

2 Diseñadora de los iconos originales de interfaz de usuario de Mac (Kare, 2008)

e) Facilidad de acceso a los distintos menús

Indicador de calidad:
El aprendiz podrá acceder en todo momento al menú, facilitando de esta manera la acciones, tareas y navegabilidad.

En los entornos virtuales de aprendizaje, el acceso ágil y rápido a la información facilita considerablemente la incursión del discente en los contenidos y recursos que cada asignatura o curso presenta.

Para ello, debemos ofrecer un sistema de navegación accesible e intuitivo que permita al aprendiz interactuar libremente con el entorno y con los materiales recogidos en éste, de tal manera que, el discente pueda acceder y profundizar en los objetos de aprendizaje ("Learning Object" LO) con mayor facilidad.

En el caso del DECANO el mapa de navegación se halla visible en todo momento anclado en la parte superior derecha de la pantalla, de manera que el aprendiz pueda consultarlo desde cualquier lugar, favoreciendo una navegación flexible en la que el discente sea quien determine en todo momento el itinerario a seguir en su proceso de aprendizaje.

Igualmente, los menú y barras de navegación incorporadas en la interfaz gráfica deben facilitar un sistema de localización, que permita al discente conocer en todo momento su situación en el entorno virtual de aprendizaje, sus herramientas y recursos. La barra superior es una de las funcionalidades que resulta familiar y cercana, incluso no siendo usuarios expertos, porque existe un aprendizaje generalizado, en fución de que es una práctica común y afín a aplicaciones, entornos, sitios web, herramientas, etc., incluso interfaz operativas o sistemas operativos de infinidad de dispositivos, razón por la cuál se asume como un elemento básico que no debe faltar en el prototipo que nos ocupa.

f) Iconos fácilmente reconocibles

Indicador de calidad
Los iconos han de ser fácilmente reconocibles para los discentes, de otro modo, se incrementaría la complejidad cognitiva.

Las plataformas de formación así como las diferentes aplicaciones informáticas, a menudo, introducen en sus entornos íconos a modo de botones o imágenes hipervinculadas a través de los cuales se accede a las diversas herramientas que incluyen. Igualmente los docentes

al implementar sus objetos de aprendizaje virtuales, diseñan íconos gráficos para representar determinados recursos, creando ambientes de aprendizaje más amigables, lo cual repercute positivamente en el nivel de motivación de los aprendices al generar un entorno más atractivo.

En ese mismo sentido, se debe desestimar la incorporación de aquellos iconos gráficos que sólo incrementan considerablemente el tamaño del material didáctico, junto con los que no sean fácilmente reconocibles por los aprendices, dado que les exige una mayor actividad cognitiva para identificar la estructura organizativa que les permita acceder a la información presentada. Más allá de su poder de atracción, los iconos tienen una función modular capaz de estructurar el contenido en bloques según sus dimensiones y orden de presentación. Normalmente los iconos siguen un patrón de uso para que resulten familiares (un icono de una papelera se traduce en *borrar*), pero en ocasiones el desconocimiento del usuario provoca errores de interpretación.

El tándem texto-icono mejora la facilidad de uso y ayuda a contextualizar su función específica Para el caso del DECANO y como ya se documentó previamente en este mismo capítulo y se complementará en la fase de evaluación, se realizó una prueba diagnóstica previa con el fin de sondear qué tan habituados están los estudiantes a los íconos y qué tan fácil o complicada resulta la metáfora para cada función o herramienta, por lo menos de los acciones más comúnes (copiar, cortar, pegar, cerrar, ampliar, entre otros)(Cosultar ANEXO: Evaluación y Pruebas). Este diagnóstico permitió confirmar el potencial de los gráficos e íconos como principal metáfora que permite el reconocimiento y aprehensión de la interfaz.

Como afirma Fló (2010), sintéticamente, podríamos reducir las posturas respecto a los modos de “leer” e interpretar de las imágenes en dos tipos: una -que podríamos llamar “naturalista”- que defiende el espontáneo reconocimiento de las semejanzas entre las imágenes y los modelos que éstas representan o denotan; la otra, que afirma que la asociación entre representación y representado en las imágenes icónicas, es el resultado de una convención, no existiendo forma alguna de reconocimiento espontáneo y natural. Ambas posturas presuponen la función denotativa de la imagen icónica. Y ambas son formas de interpretar que se inducen en los entornos tecnológicos a través de las metáforas con el fin de que se vuelvan amables e intuitivas. Adicionalmente y para el caso que nos ocupa, también se ha recurrido a la Teoría de la Gestalt y la de la percepción, con el fin de justificar el hecho de que son las que permiten predecir como las personas reaccionarán a elementos de diseño y que de la misma manera se pueden aplicar en el diseño de interfaces.

La teoría de la Gestalt permite a los diseñadores predecir como las personas responderan a los elementos de diseño. Basados en la Teoría de la percepción, los principios de la Gestalt fueron desarrollados al inicio del siglo 20 por psicólogos que creían que las imágenes eran percibidas como algo mas que la suma de las partes. Estos mismos principios podemos, además, aplicarlos al diseño de Interfaces, como un modo de ayudar a nuestros usuarios a comprender de forma

más rápida el funcionamiento de un sotfware aplicando la interacción con Patrones de Diseño.

Los patrones de diseño son una solución a un problema que se usa repetidamente en contextos similares con algunas variantes en la implementación. Estos se obtienen a partir de una abstracción de ejemplos específicos de diseño, considerando que, para ser un patrón, debe ser eficaz, o sea, haber demostrado que sirve y resuelve de modo satisfactorio el problema, y por otro lado reutilizable, que pueda ser aplicado a diferentes casos. Por esta razón es posible afirmar que usando los patrones de diseño es factible generar diseños centrados en la usabilidad, en la eficacia, la eficacia y la satisfacción del usuario final tal como lo proponen Karpich Zardalevich (2005) y Yussef Hassan (2005).

Principalmente podemos distinguir dos tipos de patrones, el primero orientado a la funcionalidad, los Patrones de Diseño de Software, y el segundo orientado a la usabilidad, los Patrones de Diseño de Interacción. Estos modelos se complementan y han resultado ser exitosos a la hora e aplicarlos al Desarrollo de aplicaciones Interactivas, por esta razón se han aplicado para el diseño de nuestro modelo de interfaz. Para el caso particular del DECANO y las herramientas de autor, se desarrolló una matriz de diseño basada en niveles de representación, que partió de una lista de funciones, herramientas y propiedades. A partir de esta lista se decidió diseñar un ícono para cada herramienta o función o bien establecer una relación sintáctica-simbólica por medio de la palabra genérica que la describe, así por ejemplo para casos como salvar, lo comúnmente contextualizado es guardar y se asocia a una carpeta o fichero por representación natural y como parte ya estandarizada. En la figura 6.14 y 6.15 se puede ver un ejemplo de como se trabajó para todos los íconos. Y en la figura 6.16 se ve una imagen de los bocetos preliminares para algunos de los elementos que luego se convirtieron en dibujos.

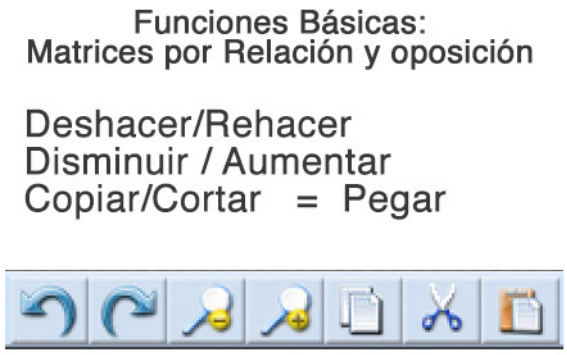


Figura 6.14 Matrices conceptuales para establecer las funciones y relaciones básicas de los íconos

Proceso básico para determinar la el nivel de representación de los íconos.

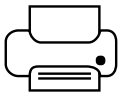
HERRAMIENTA O FUNCIÓN/ACCIÓN	NIVELES DE REPRESENTACIÓN				
	REALISTA O FIGURATIVO	TIPOGRÁFICO	GEOMÉTRICO O LINEAL	ABSTRACTO O SIMPLIFICADO	FIGURA/FONDO POSITIVO/NEGATIVO
NUEVO		Nuevo			
IMPRIMIR IMPRESIÓN		Imprimir			
SALVAR		Guardar Salvar			
EXPORTAR		Exportar			

Figura 6.15 Matriz de diseño propuesta para el trabajo con los elementos gráficos

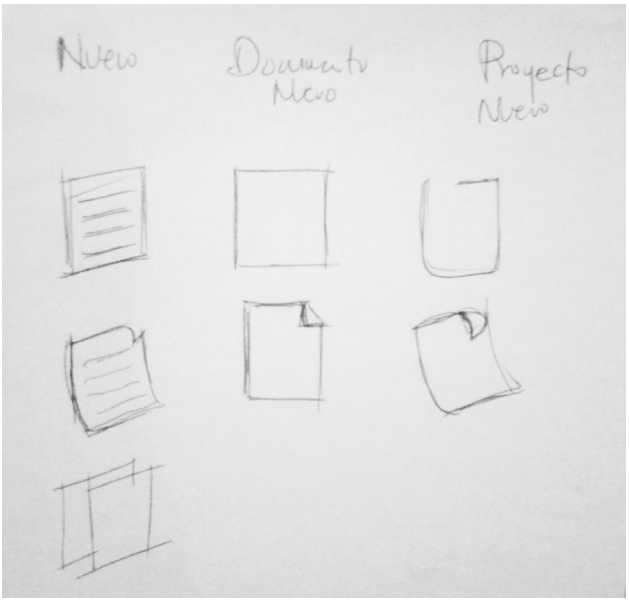


Figura 6.16 Ejemplo del Bocetaje prelimitar

g) Zonas hipervinculadas o con interacción perfectamente identificada.

Indicador de calidad
Los usuarios deben reconocer en todo momento donde pueden encontrar elementos vinculados, apelando a un elementos tipográficos diferentes o mediante imágenes fácilmente reconocibles.

Existen unas convenciones con relación a facilitar la navegación interna por entornos creados para la Red, una de ellas en relación a las zonas hipervinculadas, consiste en identificar los vínculos con textos subrayados en color azul o rojo según lo expresa Nielsen (2002) y Krug (2001), así como también mediante la transformación del puntero del ratón en una mano que señala con el dedo, al pasar por una zona interactiva.

Cuando los hipervínculos no aparecen claros, y se omite esa regla, puede suceder que gran parte de éstos no sean reconocidos por los aprendices, por lo que, se reduce su conectividad, limitando el acceso a la información complementaria y anexa al material didáctico.

Del mismo modo, cuando se incorporan enlaces hipertextuales en el EVATI se debe atender también a los siguientes criterios de calidad:

El texto hipervinculado debe aportar datos complementarios suficientemente significativos para la información que se va a mostrar al aprendiz cuando éste presione sobre el enlace.

El número de palabras hipervinculadas para un mismo enlace, parece recomendable que no supere los cuatro vocablos, para no recargar excesivamente de información al lector, y siempre procurando respetar la coherencia del discurso.

Para el prototipo que nos ocupa se ha apostado por la fórmula de introducción de imágenes hipervinculadas, lo que hizo indispensable emplear gráficos clarificadores y representativos del contenido con el que se va a encontrar el aprendiz al activarlo.

6.3.2 Metáforas del entorno

Mediante la utilización de metáforas conocidas por el estudiante en el entorno virtual de aprendizaje, se facilitará la comprensión de la estructura y la asimilación de los diferentes mecanismos de interacción presentes en el mismo (Barker y Manji, 1991).

Para ello, es necesario que estas metáforas tengan presente un conjunto de prerrequisitos necesarios para favorecer eficazmente el proceso de enseñanza-aprendizaje:

Indicador de calidad
Se deberá asegurar que la metáfora empleada sea familiar al aprendiz, y su presentación sea suficientemente explícita.

Cuando la metáfora seleccionada es conocida por el aprendiz facilita la comprensión del entorno “on-line”, y simplifica la actividad cognitiva del discente, propiciando que el ambiente de aprendizaje no sea un obstáculo para que tenga lugar.En el proceso de selección del tipo de metáforas que van a integrarse en un EVATI, se procuró que éstas tuvieran una serie de características tal como las que definen Gary y Mazur (1991):

- Fácilmente reconocibles.
- Propiciadoras de un aprendizaje significativo.
- Con gran flexibilidad para adaptarse a los diferentes niveles cognitivos de los aprendices.
- Capaces de aplicar aprendizajes anteriores a las nuevas situaciones.

La gran mayoría de ellas son ya reconocidas por los usuarios, así que se buscó respetear lo más posible esta condición. Desde el planteamiento de íconos y elementos visuales, se procuró establecer una relación lógico-semántica con cada uno de los elementos y funcionalidades, con el fin de que se incorporan elementos fácilmente comprensible o de inmediata comprensión. En la tabla siguiente (Tabla 6.1) se expone esta relación, que en la mayoría de los casos está determinada por los elementos que ya son reconocidos en toda interfaz gráfica y para el caso particular de ciertas herramientas y funcionalidades especiales para el prototipo que nos ocupa, se generaron metáforas que fueran de simple asociación.

Función	Metáfora	Resultado gráfico
Nuevo Proyecto o recurso nuevo	“Hoja en blanco”	
Imprimir Generar una impresión del trabajo	“Impresora con hoja”	
Asistencia o ayuda del tutor	Figura del tutor “Profesor Virtual”	

Figura 6.17: Ejemplo de las funciones con su metáfora visual y el resultado gráfico obtenido.

6.3.3 Presentación de la información

En los procesos de captación y asimilación de información, el sujeto pone en funcionamiento diferentes actividades cognitivas que condicionan la percepción visual del mensaje que se pretende transmitir.

En este sentido, numerosos expertos en usabilidad y accesibilidad determinan que la asimilación de la información está fuertemente influenciada por la distribución y la organización del contenido en el entorno visual, así como por las características culturales de occidente que condicionan el recorrido cognitivo al explorar un continente de información; afirmando que el usuario inicia la exploración del entorno por la esquina superior izquierda, pasando posteriormente hacia la derecha, y de ahí hacia el resto de los contenidos presentados en pantalla. Por este motivo, se hizo necesario situar a aquellos elementos considerados prioritarios, tales como las herramientas principales, los sistemas de ayuda, los elementos y funcionalidas más importantes, la información más destacada ..., en el lugar adecuado para que el estudiante los ubique de forma inmediata. Tal y como afirman, Hassan Montero(2005) y Jacob Nielsen[3](2006), existe una estructura de lectura tradicional, que se ha respetado para el caso del prototipo.

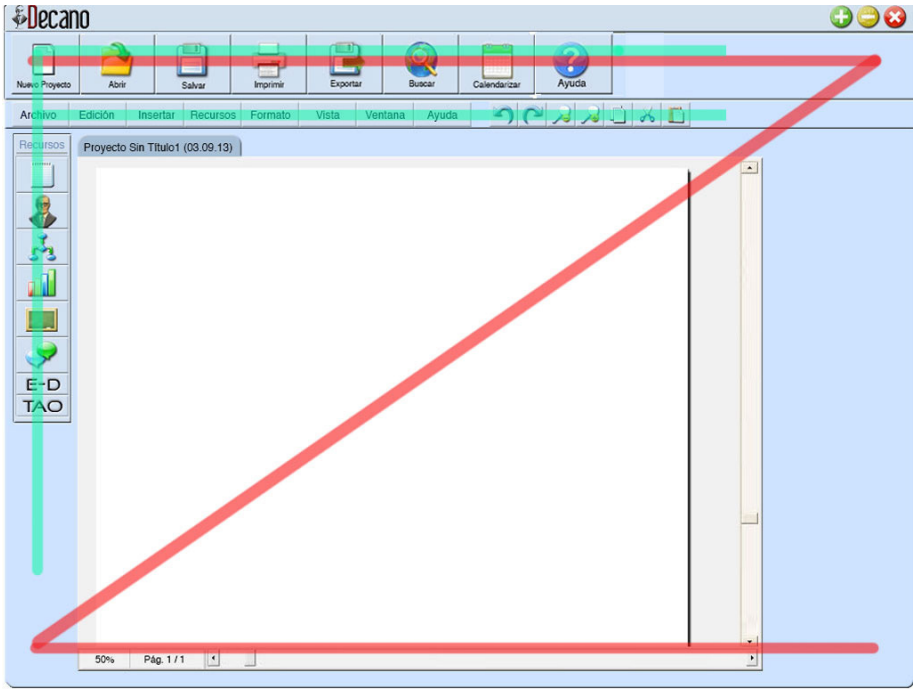


Fig. 6.18: Para la estructura del prototipo y las ventanas tanto de las herramientas de autor, como de los funcionalidades siguen los patrones de recorrido visual de sistemas y sitios web.

3 Jacob Nielsen, realizó un estudio en abril de 2006 titulado “F-Shaped Pattern For Reading Web Content”. En él se analizaban los movimientos de la mirada de 232 usuarios al mostrarles varios sitios web. Del análisis se desprende la teoria del “Patrón en forma de F”.

Por todos es sabido que, la lectura en pantalla supone para la mayoría de los usuarios un gran esfuerzo, ya que normalmente leer a través de este medio es un proceso más lento que el que implica leer en soporte papel. En consecuencia, es conveniente limitar la cantidad de información escrita y para apoyar esta situación en el prototipo se ha decidido introducir esquemas, gráficos, palabras resaltadas, ... que faciliten la visualización y retención de la información (Nielsen, 2002). La opción de introducir mapas conceptuales (sobretodo en las herramientas de autor) puede servir como recurso para compilar la información de forma sintética.

En el diseño de los materiales multimedia destinados a la formación "on-line" se deberán tener presentes algunas recomendaciones enumeradas por Nielsen (2002) y aplicadas por Del Moral (2001) al campo pedagógico y didáctico en cuanto a la presentación de los contenidos:

6.3.4 Información textual

Indicador de calidad
El tamaño y tipo de letra adoptado deberá facilitar la lectura de la información presentada. Deberá evitarse intercalar animaciones innecesarias y superfluas en la presentación del texto, ya que pueden dificultar la lectura.

Para el prototipo DECANO fue muy importante seleccionar tipos y tamaños de letra que permitieran la correcta visualización de los contenidos, ya que debemos tener presente que la lectura en pantalla incrementa considerablemente la fatiga visual, más aún si la grafía no es la adecuada. Tanto para el apoyo de los íconos al nombrarlos, los menús textuales, contextuales, ayuda, recomendaciones del tutor, ventanas, etc. Se cuidó este aspecto y principalmente se consideraron fuentes tipográficas lineales, fáciles de leer y que en cuanto al tamaño y la presentación pudieran facilitar en todo momento la lectura.

Específicamente, las directrices consideradas para la definición de las fuentes fueron las siguientes:

- No utilizar más de tres fuentes en la interfaz
- No usar más de tres tamaños de la misma fuente
- Preferentemente se decidió hacer uso fuentes *sans serif*
- El tamaño de la fuente fue calculado para poderse leer a distancia por el operador. (Una fuente menor a 8 es difícil de leer)
- Se procuró la combinación de mayúsculas con minúsculas en la interfaz, ayudas, diálogos escritos, editores, comandos, menús, etc.
- Se evitó utilizar énfasis en el texto (subrayado, itálico, sombreado) salvo en casosespeciales
- El color del texto se eligió por contraste con el fondo de la pantalla y debe respetar el

código de colores previamente definido
Cuando se usa color en el texto se debe usar en toda la palabra y no solo en ciertos caracteres
Se sugirió alinear el texto en pantalla.

6.3.4.1 Lenguaje empleado

Indicador de calidad
Los textos se presentarán sin faltas ortográficas, con construcciones gramaticales adecuadas, las cuales estarán dotadas de un vocabulario apropiado y comprensible.

A pesar de considerar como una condición obvia el que los textos se presenten sin faltas ortográficas, encontramos como algunos materiales sobretodo en Internet muestran faltas de ortografía que, no sólo dificultan la comprensión del texto, sino que también reducen poderosamente la calidad del mismo. A pesar de que son pocos los textos largos que se enuncian dentro del prototipo, existen muchos conceptos y elementos que se incluyen, en el orden de lo textual, como por ejemplo las taxonomías de objetivos, que deben ser cuidadas al máximo con respecto a la ortografía y la construcción gramatical.

Del mismo modo, se hizo una cuidadosa selección de los términos del vocabulario que se incluyeron, desde el nombre de los botones, hasta ventanas de diálogo, herramientas y recursos, tratando de presentarlos de forma sencilla y correcta, contribuyendo con ello, a que el estudiante logre un mayor entendimiento de los conceptos, herramientas o procedimientos que se muestren a través de los objetos de aprendizaje.

6.3.5 Información audiovisual

Indicador de calidad
Las imágenes incorporadas deberán adecuarse a la información textual presentada. Y del mismo modo, los documentos sonoros deberán ser claros y nítidos, primando la coherencia y la complementariedad entre éstos y el texto.

En este sentido, la condición era que existiera una concordancia entre lo que se dice mediante la palabra y lo que se ilustra con las imágenes que se acompañan.

Por tanto, se hizo indispensable que los ficheros cargados de información audiovisual que se introdujeron en el Entorno Virtual de Enseñanza-Aprendizaje tuvieran no sólo d gran calidad

técnica, sino que también su inclusión se hiciera a partir de los criterios de pertinencia y oportunidad.

De igual modo, los documentos gráficos incorporados en el entorno de aprendizaje fueron organizados de tal manera que no dificultaran la lectura de la información textual incorporada en el material didáctico.

La utilización de Internet como medio para llevar a cabo una acción formativa precisa contemplar un conjunto de requerimientos que faciliten la interpretación y asimilación de los contenidos didácticos por parte de los aprendices. Entre ellos cabe destacar:

Diseñar un entorno gráfico amigable, sencillo e intuitivo que facilite y propicie la interacción del aprendiz con el sistema.

Generar contenidos adaptables no sólo a las características individuales del estilo de aprendizaje de los discentes, sino también a las nuevas exigencias técnicas que van inherentes a la formación que se desarrolla a través de la Red.

Crear un Entorno Virtual de Enseñanza-Aprendizaje que favorezca y propicie el desarrollo cognitivo, mediante la elaboración de un escenario formativo centrado en el aprendiz.

En definitiva, en el diseño e implementación del entorno se consideraron una serie de indicadores relativos al diseño del interfaz, presentación de la información, iconos y navegación e incorporación de metáforas, dado que van a condicionar la calidad del mismo, y por ende, la permanencia del aprendiz dentro de él.

6.4 Criterios generales de usabilidad de la interfaz del Prototipo DECANO

A continuación se describen cada uno de los criterios y subcriterios de usabilidad considerados para la interfaz del prototipo propuesto:

1. **Compatibilidad:** el criterio de compatibilidad se refiere a la correspondencia entre las características del usuario (memoria, percepción, costumbres, habilidades, expectativas, etc) y las características de las tareas, y la organización de las entradas, salidas y el diálogo para una aplicación dada.
2. **Completado de las tareas:** todas las tareas disponibles antes de realizar una adaptación consideran ser accesibles tras la aplicación de la adaptación.
3. **Accesibilidad:** supone que cualquier persona sea capaz de interactuar con la interfaz de usuario.

4. **Visibilidad:** es la capacidad del sistema de mostrar todos los objetos necesarios para realizar una tarea de interacción. Para el DECANO, una de las claves en la definición de la visibilidad de una interfaz de usuario consistió en restringir la información mostrada en cada momento a la información relevante para la tarea actual. Por lo tanto, cualquier adaptación aplicada debería intentar restringir la visibilidad a los datos relevantes. Los datos relevantes son como máximo el conjunto de datos que son accesibles en cada momento, es decir, el conjunto de datos del dominio que necesitan el conjunto de tareas activas (*enabled task set*).
5. **Multiplicidad de dispositivos:** es la capacidad del sistema de ofrecer distintas posibilidades de dispositivos de entrada y salida. Por ejemplo, muchas órdenes en la interfaz propuesta pueden ser realizadas con el teclado o con el ratón (principio WIMP).
6. **Consistencia:** el criterio de consistencia se refiere a la manera en que las elecciones de diseño (código, nombrado, formatos, procedimientos, etc.) se mantienen, condición que se cumple en el sistema y que es parte de los indicadores de calidad. En el caso que nos ocupa, donde tenemos una presentación antes de la adaptación y otra presentación distinta tras la adaptación, tendremos que centrarnos en este segundo tipo de consistencia descrita.
7. **Consistencia en la distribución (layout):** se refiere a la manera en que los elementos de la interfaz de usuario son distribuidos para mejorar la consistencia de la interfaz de usuario. Este criterio considera el espacio físico de despliegue como el soporte para la distribución. Para este caso se sugiere que se base en los estándares recomendados, tal y como se han explicado y aplicado en el prototipo propuesto.
8. **Uniformidad en la distribución:** representa la uniformidad en la distribución de los elementos de la interfaz de usuario. Se evalúa de acuerdo a las alturas, anchuras y alineaciones y generalmente se recomienda expresarse a través de una retícula o framework, como es el caso particular de la propuesta.
9. **Densidad de contenencia:** determina cuántos elementos están contenidos en cada contenedor. Un número alto para este criterio significa que hay muchos elementos dentro de cada contenedor, lo cual puede llegar a confundir al usuario debido a la falta de una estructura clara y de una sobrecarga de información, por esta razón se dosificó el número de herrramientas por sistema o subsistema (como en el caso del TAO, por ejemplo).
10. **Márgenes:** A nivel perceptual, el hombre busca reconocer este elemento como indicativo de orden y estabilidad, por lo que es deseable mantener los mismos márgenes a lo largo de la interfaz en los diferentes diálogos y ventanas, tal cual se ha propuesto en el proyecto DECANO. Este es un elemento consistente en todas las ventanas del prototipo.
11. **Relación de aspecto:** las ventanas de diálogo para una misma tarea deben mantener la misma relación de aspecto para hacer fácil la diferenciación de unas tareas a otras.
12. **Equilibrio de áreas:** especifica la distribución de los componentes en las ventanas.

Distribuciones equilibradas mejoran la estructura de la información y reducen la carga cognitiva.

13. **Continuidad:** la continuidad en la interacción no se da cuando un usuario se ve obligado a dividir su atención entre dos entidades. Por ejemplo, la continuidad se preserva en gran medida si una adaptación sustituye un elemento por oto, que ocupa el mismo espacio en la pantalla que el original, ya que el usuario no necesita desplazar su atención de la misma zona de la pantalla. Sin embargo, si un elemento, por ejemplo una lista desplegable, es sustituida por un grupo de botones de radio con una cantidad moderada de elementos, el usuario necesita volver a centrar su atención en el lugar apropiado antes de continuar con el uso normal de la aplicación.
14. **Carga de trabajo:** este criterio tiene que ver con los elementos que influyen en la reducción de la carga cognitiva o perceptiva que el usuario debe emplear para utilizar la interfaz de usuario. Parte de esta condición se trabajó tanto en el diseño de los íconos como en la determinación de las metaforas del entorno y las funciones.
15. **Migrabilidad:** es la capacidad del sistema para permitir al usuario/sistema transferir la responsabilidad de una tarea. Esta es especialmente interesante en interfaces de usuario como la desarrollada, que incluyen agentes software, donde un agente, por ejemplo, el usuario, puede pasar el control de una tarea a otro agente para completar la tarea. Por ejemplo, en el nivel físico de abstracción, el usuario puede delegar la tarea de completar la escritura de las órdenes al sistema. En el nivel funcional de abstracción, guardar un fichero puede ser hecho explícitamente por el usuario o se puede permitir al sistema manejar la tarea de guardar el fichero en disco de vez en cuando conforme el prototipo piense que es necesario.
16. **Control de diálogo:** el criterio de control de diálogo se refiere tanto al procesamiento por parte del prototipo de las acciones explícitas del usuario, como al control que los usuarios tienen sobre cómo realizar sus acciones el sistema.
17. **Multitarea:** es la capacidad del sistema de permitir que el usuario realice varias tareas a la misma vez, de forma que se superponen en el tiempo. Para el caso del EVEATI, se permite por superposición, realizar más de una tarea e implementar más de una herramienta.
18. **Adaptación:** la adaptación del sistema se refiere a la capacidad del mismo para comportarse de acuerdo al contexto de uso de la aplicación.
19. **Adaptabilidad:** la capacidad del sistema de permitir la personalización de la interacción por parte del usuario. Normalmente, la adaptabilidad es tratada en tiempo de diseño, tal y como se explica en el diseño de la arquitectura. Un sistema diseñado para ser adaptable también será más fácilmente adaptativo, ya que para permitir la adaptabilidad es necesario proporcionar un motor que puede ser usado de igual manera para producir adaptividad
20. **Multiplicidad de perfiles:** es la capacidad del sistema de permitir al usuario asumir distintos perfiles en el sistema. Ello significa que el usuario puede pasar de un perfil a

otro a lo largo del uso del sistema, o que distintos usuarios tendrán distintos perfiles en el sistema.

21. **Preservación:** es la capacidad de preservar, tanto como sea posible, el estado de la interfaz de usuario antes de la adaptación. Este criterio permite evaluar cuán conservativa es una adaptación.
22. **Representatividad:** describe la relación entre un término y/o símbolo y su referencia. Los códigos y nombres usados son significativos para los usuarios cuando tienen una fuerte relación semántica entre los códigos y los objetos o acciones que representan. Como se ha mencionada previamente, esta es una de las condiciones que más se han cuidado y sobre la que incluso se ha propuesto una matriz para du diseño.
23. **Consistencia del esquema de colores:** la consistencia del esquema de colores está relacionada con los elementos de la interfaz de usuario. Es deseable usar el mismo esquema de colores para tareas similares. De igual manera estas condiciones se preservaron en todo el prototipo y subsistemas.
24. **Consistencia del esquema de texto:** la consistencia del esquema de texto está relacionada con los distintos tipos de presentaciones usadas por los elementos de texto en una interfaz de usuario. Un número reducido de fuentes es deseable en la mayoría de los dominios de aplicación.
25. **Consistencia tipográfica:** preservar tanto el tipo de letra, como los puntajes y estilos para indicar cada uno de los elementos, funcionales, menús, ventanas, ayudas, advertencias, reportes, etc.
26. **Consistencia del esquema de botones/enlaces:** se refiere a la forma en que los botones/enlaces son presentadas en la interfaz de usuario. Los botones/enlaces se han diseñado y presentado de la misma forma. También, este criterio está relacionado con los patrones de botones/enlaces recurrentes que habitualmente se usan en las interfaces de usuario, como por ejemplo, el habitual patrón “Aceptar”, “Cancelar” y “Ayuda”.
27. **Patrón de botones/enlaces:** describe la distribución para los patrones de botones/enlaces.
28. **Esquema de colores de botones/enlaces:** los esquemas de colores usados para presentar los botones/enlaces.
29. **Esquema de texto de botones/enlaces:** el esquema de texto usado para presentar los botones/enlaces (incluyendo la terminología – el mismo concepto debe ser representado con la misma palabra (*Quitar*, *Salir*, *Cerrar*, ...).
30. **Orientación:** Orientar al usuario tiene que ver con los medios disponibles para aconsejar, orientar, informar, instruir y guiar a los usuarios durante su interacción con la computadora (mensajes, alarmas, etiquetas, ...).
31. **Preentividad:** un sistema es preentivo cuando no permite al usuario realizar las tareas en cualquier orden, sin seguir ninguna secuencia determinada. Sin embargo, un nivel bajo de

preentividad puede producir en el usuario una sensación de “perdido en el espacio”, donde no exista ninguna guía clara sobre cómo volver a casa (realizar las tareas). Tradicionalmente los sistemas adaptativos, y especialmente los sistemas de autorización inteligentes, han estado enfocados a guiar al usuario para evitar precisamente la sensación de “perdido en el espacio”, pero dicha tarea debe ser compaginada con el objetivo de permitir al usuario aprender a su manera. Por lo tanto, la preentividad se hace necesaria dentro del diseño de la interfaz de un prototipo como el que nos ocupa.

32. **Reutilización de entradas y salidas:** es la capacidad del sistema de permitir la utilización de entradas y salidas como futuras entradas. Las órdenes de “Copiar & Pegar” son ejemplos típicos de este tipo de reutilización de entradas y salidas. El motor de adaptación debe proporcionar automáticamente la reutilización de las entradas cuando este criterio sea seleccionado y sobretodo en el caso de las funciones básicas.
33. **Manejo de errores:** este criterio se refiere a los medios disponibles para evitar o reducir los errores y recuperarse de ellos cuando estos ocurren.
34. **Tolerancia a anomalías:** no importa lo bien que un prototipo esté diseñado, los usuarios cometerán errores de los cuales querrán recuperarse. Los sistemas tolerantes a anomalías de uso pueden: (1) detectar los estados erróneos o “peligrosos”, (2) evitar que se entre en un estado erróneo, (3) corregir descuidos y errores. Esto fue básico en el caso que nos ocupa, puesto que al tratarse de una herramienta nueva y con caracterísiticas muy particulares, se hacía necesario incluir este aspecto. Cuando los usuarios pueden confiar en que el prototipo les advertirá de aquellas acciones que sean “peligrosas”, y recibirán ayuda para recuperarse de pequeños errores, se sentirán más libres de explorar la interfaz de usuario sin miedo. En este caso la noción de esfuerzo proporcional de Thimbleby (1990) es interesante aquí: si un error es fácil de cometer, entonces su efecto también debe ser fácil de recuperar. La tolerancia a las anomalías de uso en las interfaces de usuario adaptativas puede ser mejorada de distintas formas: (1) corregir entradas de texto, (2) inferir el objetivo adecuado de los clics del ratón, (3) el usuario puede ser avisado cuando se realicen tareas “peligrosas”, (4) detectar situaciones “arriesgadas” y advertir al usuario sobre el posible resultado indeseable. La figura 6.10 se describe un ejemplo de compromiso de usabilidad donde se han considerado todos los criterios descritos a lo largo de esta sección, así como las interrelaciones que surgen entre ellos. Sin embargo, debe ser el diseñador el que establezca los parámetros de usabilidad apropiados para su aplicación dentro del prototipo y todos su subsistemas. El compromiso no sólo puede ser especificado para cada plataforma, sino que puede ser especificado para los distintos entornos físicos o tipos de usuarios.

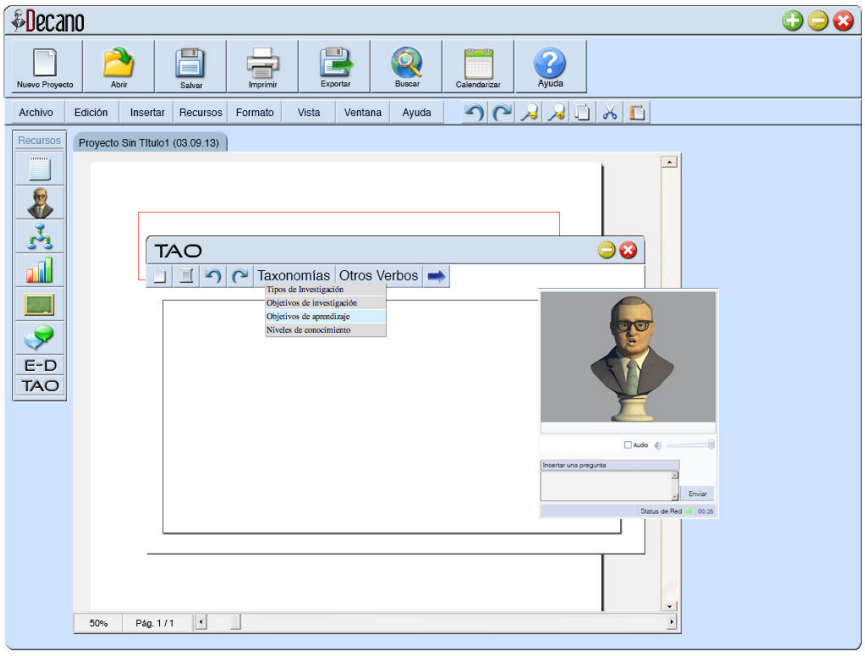


Fig. 6.19: Presentación de la interfaz del prototipo y subsitemas.

6.5 Requerimientos generales de la interfaz para la interacción con el prototipo:

El primero consiste en conseguir la comunicación entre el sistema y los usuarios. Las principales características que debe presentar la interfaz son: facilidad de aprendizaje y facilidad de uso, por ello resulta útil beneficiar al mayor número de usuarios, señalando de forma muy especial la capacidad de adaptación a las características que éstos puedan presentar.

El segundo consiste en implementar diferentes modos de liberar a las personas del uso de una interfaz rígida y unimodal, eliminando las barreras de comunicación presentes en la mayoría de los sistemas actuales. El principal objetivo de una interfaz multimodal es llegar a una interacción natural entre persona y máquina, esto es, una interacción que se asemeje en la medida de lo posible a la interacción persona-persona

El enfoque antropomórfico fue el seleccionado como el óptimo para la construcción de la interfaz del prototipo computacional, para el trabajo en grupo asistido por un agente basado en el modelo instruccional que estamos planteando ya que persigue mejorar la interacción o comunicación humano-computadora e intenta imitar el proceso natural de comunicación persona-persona.

6.6 Requerimientos de la interfaz en función del prototipo al que se aplicará

Los sistemas de interfaz para la interacción con agentes inteligentes, derivados del enfoque antropomórfico y los modelos de interacción natural poseen diferentes características (Johnson et al., 2000) y son al mismo tiempo requerimientos que son de gran utilidad para diseñar una interfaz de esta naturaleza, algunas características son:

1. Estos sistemas permiten actuar y dialogar con los agentes, de manera que en el momento de realizarse acciones, estas pueden ser vistas de diferentes ángulos.
2. Los estudiantes pueden realizar preguntas en cualquier momento
3. El agente esta en todo momento “observando” el actuar del estudiante
4. El agente puede reconstruir y redefinir su actuar en cada momento a partir del actuar del estudiante
5. El agente puede adaptar su actuar a situaciones inesperadas.
6. El estudiante puede tomar el control en cualquier momento.
7. En caso de errores el agente ayuda a que el estudiante aprenda de ellos.
8. Existen otras variadas ventajas de estos agentes tales como el manejo de emociones, apoyo efectivo del trabajo colaborativo, interacciones pedagógicas adaptables, etc. (Ryokai, 2002).
9. Los agentes animados con interfaces conversacionales proporcionan un paradigma intuitivo de interacción ya que el usuario no necesita adquirir nuevos conocimientos. Estas interfaces presentan redundancia y complementariedad en los modos de entrada, por lo que aumenta la fiabilidad de la comunicación entre sistema y usuario.
10. Los usuarios encuentran estos sistemas más amigables y cooperativos. Los agentes autónomos pueden utilizar esa ventaja para entablar una conversación con los usuarios de forma más natural.
11. Poseen elementos de comunicación no verbal.
12. Son capaces de tomar la iniciativa, dar sensación de presencia

Las conductas que deben mostrar los agentes para conseguir desarrollar una conversación que resulte natural son muy extensas, para el agente propuesto se han considerado como mínimo las siguientes:

Emoción: El personaje puede exhibir expresiones faciales que denoten emoción y gestos expresivos, por ejemplo, para aconsejar, animar o enfatizar algo al usuario.

Personalidad: La personalidad de un Agente ha de ser elegida teniendo en cuenta sus tareas específicas en un contexto dado, pues la personalidad presenta al personaje con

sus propias parcelas de conocimiento y perfiles de interés.

Se deben considerar los objetivos de la conversación: La razón por la que el hablante está comunicando una cosa (es decir, el objetivo que tiene en mente el hablante) y la forma de esta comunicación están muy relacionadas. Las contribuciones a una conversación pueden ser de propuesta y de interacción como sugiere Montoro Manrique (2000).

A partir de este análisis se pueden inferir que ciertos modelos son de aplicabilidad directa en los agentes. Así, teniendo en cuenta lo que un personaje quiere expresar, se puede acompañar su discurso con cierta expresión facial o gesto corporal. Para el modelo que nos ocupa esta condición fue tomada en cuenta par la interfaz del agente tutor, tal y como se ve en la siguiente figura:

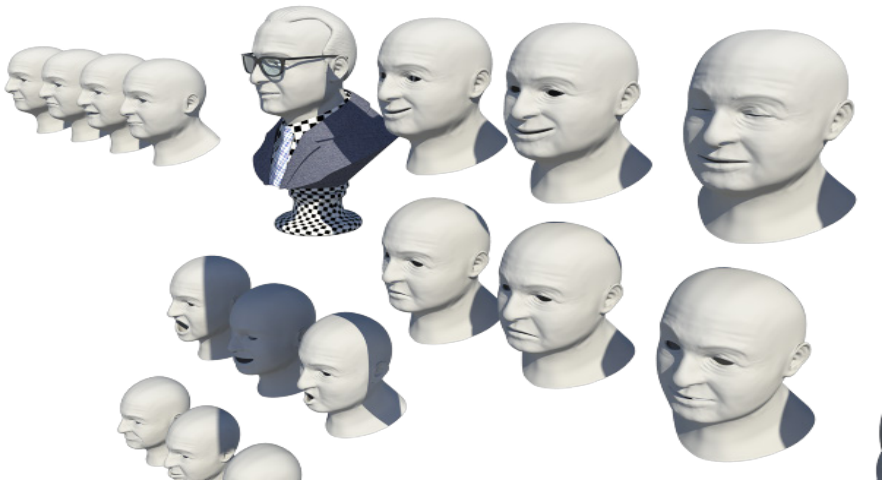


Fig. 6.20: Expresiones del Agente Tutor

Para el modelo prouesto de agente tutor, se consideró fundamenteal que el agente sea capaz de:

1. Ofrecer explicaciones conceptuales;
2. Dar consejos para solución de problemas;
3. Hacer recordatorios o sugerencias
4. Realizar la introducción a los problemas (contextualización);
5. Ofrecer también sugerencias con nivel de prioridad (en novatos se sugiere y se toma el control);
6. Hacer interjecciones (puntualiza aspectos importantes del progreso p.e. felicitaciones, ánimo)

La acciones del tutor virtual se caracterizan por transmitir información al alumno. Esta interacción con el alumno, se ha decido se realice de formas distintas, complementarias:

- Mensaje de texto, proporcionado en forma escrita
- Mensaje hablado, transmitido a través de la voz del tutor virtual
- Mensaje gestual, transmitido mediante expresión facial

Para que así cada uno de los niveles aporte mayor riqueza a la expresividad del tutor.
Dentro de las acciones anteriores, se han seleccionado dos, que se analizaron y prepararon a profundidad. Estas acciones son:

- Mensaje de Bienvenida. Consiste en ofrecer un saludo al alumno, cada vez que inicia una sesión. Pues se considera de especial relevancia la primera forma de contacto del alumno con el sistema.

Informarle del progreso en el curso. Consiste en ofrecer al alumno un mensaje de Información y Seguimiento al final de cada hito relevante dentro del curso. Por ejemplo, al completar una actividad, al finalizar un módulo.

Se han seleccionado estas actividades porque, aún siendo sencillas, se estima que tienen un impacto positivo en la motivación del alumno. Cuando el alumno inicia una sesión, el tutor le saluda, con un Mensaje de Bienvenida.

En el nivel reactivo de razonamiento del tutor, se configuran una serie de reglas que determinan este mensaje, en función de las características transitorias del alumno y según el contexto global.

Contenido del Mensaje	Condicionante	Ejemplo en el prototipo
Saludo	Según la hora del día	Buenos días Buenas tardes Buenas noches
- Presentación formal del agente pedagógico virtual. - Explicarle cómo puede ayudarle	Si es la primera vez que el alumno entra en el sistema	Soy un agente pedagógico virtual y mi misión es ayudarte en la realización de este curso.
Según la continuidad en el Tiempo - tiempo desde la última sesión - número de sesiones a corto plazo - tiempo total invertido en el corto plazo	Mensaje de motivación	Con trabajo constante consiguirás lo que te propongas.

Tabla contenido del mensaje de bienvenida

Cuando el alumno completa una actividad, el tutor le ofrece un Mensaje de Seguimiento con información sobre:

- la dedicación (tiempo y número de sesiones)

• un mensaje de motivación

La voz con la que se expresa el agente pedagógico se determinó principalemente por la configuración inicial del mismo, incluida en los aspectos físicos (que junto con los rasgos de personalidad forman en Cognitiva las características definitorias). Esta voz se generó acorde con los rasgos de personalidad; y se modulará según los estados transitorios (emociones, estados físicos y otros).

La expresión facial, tiene unas influencias similares a la voz. En el caso de las tareas seleccionadas, no se requieren movimientos. Pero para el caso general de acciones de movimiento en un entorno 3D, rasgos de personalidad y estados físicos tendrán influencia en la forma de moverse del agente pedagógico. Las actitudes hacia los individuos (en este caso el alumno) tienen unos valores que perduran a lo largo del tiempo, y los estados físicos se calculan según el contexto, la hora del día con lo que pueden influir.

6.7 Requerimientos del prototipo para la interacción

6.7.1 Gestión de entradas del usuario

La gestión de entradas se puede hacer mediante distintas técnicas de interacción entre aplicación–dispositivo. Esta interacción se puede realizar en diferentes modos: pregunta–respuesta, órdenes, etc. Los eventos es el principal mecanismo para la comunicación entre el usuario y el sistema interactivo. Cuando el usuario interacciona con los dispositivos, estas acciones se trasladan a eventos software que se distribuyen a la ventana apropiada (en un sistema de ventanas). Todos los eventos están identificados (mediante un tipo de evento) que permite al software que los recibe distinguir la información que se pretende comunicar. Para el prototipo diseñado se establecieron 3 mecanismos de comunicación entre usuario y aplicación:

Petición (*request*). El programa espera hasta que se produzca una entrada. Los Dispositivos están en espera. Es un diálogo dirigido por la aplicación.

Muestreo (*sample*). Ambos trabajan concurrentemente. Consulta del estado del dispositivo al realizar una petición. Los datos no son almacenados, por lo que se consulta el estado actual.

Evento (*event*). Se provee de una cola de sucesos por parte del dispositivo. La aplicación está dirigida por los datos, y permite entradas asíncronas.

6.7.2 Comunicación entre objetos

Un punto importante de la gestión de eventos es la comunicación con otros objetos interactivos. La organización de la información afecta a la impresión general del interfaz. Por lo que para establecer la comunicacón se incluyeron los siguientes elementos:

- Diseño (formato de pantalla, organización general)
- Representación de la información (métodos de codificación)
- Realimentación (seguimiento)
- Comunicación con el usuario (errores, mensajes de estado, etc.)

En el nivel de presentación se debe mostrar la información de forma que sea comprensible y asimilable por el usuario. Hay que tener en cuenta que el espacio donde se presenta la información es escaso, por lo que deberemos hacer un uso racional de este recurso.

Para la representación de objetos se propuso la utilización de una simbología y codificación para que sea fácilmente identificable, económica en recursos (tamaño en pantalla) y consistente (iconografía, colores, texto, etc.)

6.7.3 Diseño de la presentación

El significado de una imagen puede ser más fácilmente percibido por el observador si posee claridad visual. Deberemos por tanto enfatizar la organización lógica de la información. Para conseguir una buena organización se han utilizado las reglas de Gestalt ya previamente mencionadas, que permiten mejorar la claridad visual de la información. Estas reglas se basan en cómo organiza el observador los estímulos visuales (de forma global) lo que permite mejorar la interpretación de la presentación y la asimilación de la interfaz.

6.7.4 Realimentación

La realimentación es de gran importancia en los sistemas interactivos, ya que el usuario debe estar informado en cada momento de las acciones que realiza. Cuando por ejemplo una tarea tarda más tiempo del razonable, se deberá informar mediante algún tipo de mensaje de ese proceso para no provocar incertidumbre. Sin embargo, un problema que deberemos tener en cuenta es que tenga una gestión rápida, ya que en tal caso puede no coincidir el estímulo con la respuesta del sistema. Algunos ejemplos de realimentación propuestos para el prototipo son:

- Sobre las órdenes. Mostrar efectos, errores, confirmación, indicadores
- Sobre la selección. Resaltar de forma no ambigua la orden activa

Esta realimentación debe ser fácil de leer y entender. Para ello se debe fomentar la consistencia, y en algunos casos puede condicionar la estructura de datos del modelo, ya que sea necesario almacenar información adicional para realizar el feedback.

Para su diseño, se estudiaron las acciones de cada tarea y como se llevaba a cabo la interacción (realimentación del propio gesto), confirmación (selección, mensajes, iluminación) y posibles errores (pantalla de aviso). La realimentación informa al usuario acerca de su interacción con el prototipo. Indica qué está haciendo y le ayuda a realizarlo correctamente. Se puede utilizar cualquier combinación de canales sensoriales (sonoro, visual, táctil, etc.).

6.7.5 Manejo de errores

El manejo de errores se implementa como elemento preponderante en un EV apoyado de un agente con características reactivas. Existen muchas recomendaciones, lo más comun que se haga de la siguiente manera:

1. Decidir la acción correctiva del error: Esto implica especificar si se envía un mensaje de error o se lanza un evento de error para disparar una transición.
2. Definir la respuesta de error: Esto implica signar la información encontrada en el prototipo, cuando se lanza un error o excepción durante el servicio.

Dentro del prototipo, un factor crítico desde el punto de vista del usuario, es cómo se organizan los mensajes de error y su explicación. Normalmente ocurren por un desconocimiento por parte del usuario que puede ser motivado por diferentes causas:

- Errores por acciones del usuario. Error en la traslación entre la intención del usuario y su acción (intención correcta, pero realización incorrecta). La solución es mejorar el diseño ergonómico, mejorar los aspectos físicos del diseño (ubicación de menús, tamaño, posición, visibilidad, color, etc.)
- Errores por las intenciones del usuario. El usuario realiza una acción equivocada. El modelo de usuario no es correcto. La solución es mejorar el modelo mental. Es importante buscar posibles causas ya que el usuario está asumiendo un modelo mental incorrecto.

El usuario, ante un error, debe reconocer qué ha sucedido, para evitar confusión. Algunas técnicas que se deben evitar en los mensajes de error son:

- Tono imperativo. Aumenta la ansiedad del usuario. Dificulta la corrección del error («Acción ilegal», «error fatal», “terminación anormal del programa”).
- Mensajes genéricos o confusos. Ofrecen poca información («error sintáctico», “run time error n. XXXX”).

En ambos casos, como parte fundamental dela interfaz se debe considerar la forma de exponer y presentar esta tipo de elementos, puesto que pueden perjudicar la apreciación general por parte de los usuarios, quienes no estarían cómodos o se sentirían ofendidos si un diálogo, ventana o respuesta fuera descalificativo o bien incluso lo sintieran ofensivo.

Se han implementado dos niveles sintaxis para el manejo de errores, por un lado las frases que vienen en las ventanas de diálogo y los textos contextuales cuando el alumno está realizando una accion equivocada, o cuando por ejemplo no ha completado una instrucción o tarea y por otro lado el apoyo del agente tutor, quien a manera de consejos, le recomienda o sugiere al alumno utilizar otro elemento, componente, etc. y que siempre lo hace como una invitación o recomedación, más que como un regaño.

Conclusiones

En este capítulo hemos visto las características más relevantes de los sistemas interactivos y las dificultades que se plantean a la hora de realizar un diseño efectivo. Si bien para el diseño de sistemas interactivos existe abundante bibliografía, la mayoría está basada en recomendaciones, consejos y guías de estilo para el diseño que están recogidas de la experiencia de los autores. Sin embargo, raramente aportan una metodología clara para llevar a cabo de forma sistemática el proceso de diseño.

Particularmente en este capítulo se han incluido diferentes propuestas metodológicas basadas en teorías cognitivas para el diseño defectivo de aplicaciones interactivas, lo que permite implementar un modelo más completo para el diseño de interfaces semejantes al caso de aplicación desarrollado.

Para ello, deberemos aplicar notaciones que permitan analizar las tareas que el usuario realiza en su entorno de trabajo, y utilizar técnicas para su traducción a conceptos de diseño. Se expresan también una serie de técnicas que nos permiten trasladar de forma efectiva el modelo conceptual del sistema a un modelo computacional basado en eventos y componentes gráficos, que se trata de un modelo centrado en el usuario. Son muchos los aspectos que se seben considerar, pero lo que se ha presentado en este capítulo se puede considerar como una “lista” de elementos clave para el desarrollo de la interfaz de sistemas semejantes.

Cuando se usa una herramienta o se ingresa a un prototipo existe algo entre el objeto de interacción y uno mismo: la interfaz que es la manija para abrir la puerta, es algo que nos informa qué acciones se pueden ejecutar y cuáles no, cuáles son los cambios realizados y que finalmente

nos permite acceder a un prototipo o una herramienta. La interfaz es un elemento de conexión que facilita el intercambio de datos.

El concepto de software que apoya el proceso creativo está fundamentado en valores de usabilidad, definida la usabilidad del prototipo o de la herramienta como la utilidad, la facilidad de uso, de aprendizaje, y la habilidad para realizar una tarea por parte de un usuario en un contexto dado. Una interfaz atractiva y de fácil manejo permite que el usuario logre una excelente interacción con el software soportada por un sistema inteligente que se fundamenta en la gestión de ideas y que de manera atractiva pretende capturar la atención del usuario a partir del diseño gráfico.

La reflexión más profunda sobre el desarrollo gráfico y conceptual del software, concluyó que la interfaz debería cumplir con el principio de utilidad, entendido este como la capacidad que tiene una herramienta para ayudar a desempeñar tareas especificas, y que a su vez debía caracterizarse por la facilidad de uso relacionada directamente con la eficiencia y la efectividad, de manera que el usuario al interactuar con la herramienta pueda aprender fácilmente en un tiempo corto.

Partiendo de la premisa de que la interfaz de usuario es el mecanismo que permite establecer un diálogo entre el entorno y el usuario, se definieron los siguientes valores de usabilidad para su desarrollo:

Fácil de Aprender: Facilidad con la que nuevos usuarios desarrollan una interacción efectiva con el sistema. El diseño del prototipo se ha creado para garantizar que su manejo sea fácil, hasta el punto de ser intuitivo.

Flexible: El prototipo cuenta con una considerable variedad de posibilidades con las que el usuario y el sistema pueden intercambiar información. Se ha tratado de garantizar una multiplicidad de vías para realizar una tarea, de manera que todos los procesos nuevos tengan similitud con tareas anteriores.

Robusto: Es el nivel de apoyo que el prototipo presta al usuario para facilitar el cumplimiento de sus objetivos. Este apoyo se relaciona con la capacidad de observación del usuario, de recuperación de información y de adaptación de la tarea al usuario.

Entendible: Este valor le permite al usuario entender efectivamente cuales son las diferentes funciones del prototipo, para sacar el mayor provecho del mismo en los diferentes entornos en los que se use.

Novedoso: Que tiene implementada una interfaz de usuario diferente, muy atractiva y de fácil manejo.

Inteligente y atractivo: El prototipo brinda una ayuda inteligente para los usuarios creativos, gracias a la integración de diferentes algoritmos con diseño visual orientado a la gestión de ideas.

A partir de estas características conceptuales y con el objetivo de verificar la transferencia y aplicación de conceptos relacionados con el diseño, se centró el desarrollo visual de la interfaz gráfica en los siguientes principios:

• Metáforas Visuales: Se busca que los íconos de la aplicación tengan una representación en acción con los objetos del mundo real, de tal forma que los nuevos usuarios puedan captar rápidamente cómo funciona la aplicación; se incluyen además controles de reproducción, neuronas como metáfora de conexión de información y botones estándar para afianzar la facilidad de uso.

• Experiencia de usuario: El usuario debe sentir que está controlando el prototipo, de tal forma que comprenda con mayor facilidad las acciones que realiza sobre los objetos. Así se veló porque el usuario conozca inmediatamente el resultado de las acciones que realiza mediante elementos visuales como un medio para proporcionar información y mejorar la experiencia del usuario. Para incrementar la sensación de manipulación de la interfaz se incluyó el sistema de arrastre (Drag and Drop) de elementos, permaneciendo estos visibles mientras se realizan acciones en ellos.

Intentar reflejar el mundo real del usuario o su representación mental del problema sobre el espacio de la pantalla, no es una tarea sencilla. El alto grado alto de percepción y de expresión que pueden proveer las interfaces icónicas, implica un importante trabajo de diseño para obtener el mejor aprovechamiento de esas cualidades.

Debido a las restricciones y condicionamientos impuestos dentro de la definición de diseño icónico sobre las representaciones visuales a utilizar y la manipulación a proveer, es lógico pensar sobre las dificultades que un diseñador puede atravesar en el momento de encarar el diseño de una interfaz icónica.

CAPÍTULO 7:

EVALUACIÓN

CAPÍTULO 7: EVALUACIÓN DE LA PROPUESTA

Una vez diseñadas la arquitectura y la interfaz, la evaluación de ambas permitirá verificar que cumplen con los objetivos planteados al comienzo de este trabajo y sobretodo permitirá comprobar las hipótesis.

Para el caso de la arquitectura, existen dos aproximaciones para llevar a cabo esta evaluación.

- La primera consiste en realizar medidas cuantitativas relacionadas con los distintos atributos de calidad de la arquitectura. La ventaja de utilizar esta técnica la constituye el hecho de que el resultado estará formado por una serie de medidas objetivas acerca de la calidad de la solución propuesta. La desventaja, como sucede con muchas medidas numéricas, es que la utilización de medidas numéricas para evaluar la arquitectura elimina mucha información que puede resultar relevante para conocer las características de la misma.

- La segunda manera de realizar la evaluación consiste en llevar a cabo un análisis cualitativo de la arquitectura, de manera que el resultado no es una medida de la calidad de la solución, sino un informe que recoge las características de la arquitectura, los posibles riesgos y puntos de conflicto y una estimación acerca del cumplimiento de los objetivos de la misma. Consideramos esta segunda forma, como la más adecuada para nuestro planteamiento.

Y para el caso de la interfaz, también se hará una evaluación en función de principios de usabilidad y diseño, traducidos en escalas, con el fin de que el resultados de la evaluación se convierta en un informe que describa los puntos positivos del diseño y los focos de atención.

Como se planteó inicialmente, para este modelo, se diseñó un prototipo funcional que fue monitoreado y evaluado por varios meses y adicionalmente se complementó el planteamiento metodológico para el diseño de la interfaz, como resultado de los aspectos cualitativos y cuatitativos, resultado de las pruebas de usuario, la aplicación de indicadores de calidad y la evaluación de expertos o heurística.

De esta manera se llevaron a cabo dos tipos de evaluación: la formativa y la sumativa. La evaluación *formativa* se centra en el diseño del prototipo y su objetivo principal es detectar problemas y realizar las modificaciones necesarias para solventarlos a lo largo del proceso de desarrollo, en definitiva, mejorar el funcionamiento del prototipo y asegurar que el resultado final cumple con los requisitos de diseño. La evaluación *sumativa* se orienta al resultado final y trata

de probar algún aspecto formal del prototipo. Según (Littman and Soloway, 1988), la evaluació formativa se enfrenta a la pregunta: “¿Cuál es la relación entre la arquitectura del prototipo y su comportamiento?”, mientras que la sumativa trata de responder: “¿Cuál es el impacto educativo del prototipo en el alumno?”.

Así, el método de evaluación usado para DECANO se basa las recomendaciones de Murray (Murray, 1993) para el diseño de sistemas educativos inteligentes:

- Diseño iterativo: partiendo de un prototipo inicial la funcionalidad del prototipo se va probando y refinando en sucesivas iteraciones.

- Diseño participativo: con el fin de asegurar que el prototipo resultante sea utilizable se involucra a los usuarios en su desarrollo.

- En contraposición al enfo que cuantitativo de la evaluación, que intenta encontrar las causas y las consecuencias de las características del prototipo, se ha realizado un enfoque cualitativo, tratando simplemente de identificar esas características. Para ello se documentan todos los cambios realizados en cada una de las iteraciones del diseño: las motivaciones de esos cambios, los resultados y cualquier otro aspecto encontrado durante el estudio.

7.1 Selección del Método de Evaluación para la Arquitectura

Los métodos que finalmente se consideraron inicialmente para evaluar la arquitectura fueron ARID (Active Reviews for Intermediate Designs), SAAM (Software Architecture Analysis Method) y ATAM (Architecture Tradeoff Analysis Method), todos ellos desarrollados por el grupo de arquitecturas software del SEI (Software Engineering Institute - Carnegie Mellon University).

Tras un primer contacto con los tres, se descartó el primero de ellos, ARID, porque su utilidad se encuentra en la evaluación de arquitecturas sin completar o trozos de arquitecturas, lo cual no se ajusta a nuestras necesidades de evaluación.

De entre los otros dos métodos, se ha preferido ATAM principalmente por dos motivos. El primero de ellos es que SAAM se centra sobre todo en la evaluación de la modificabilidad de la arquitectura. Aunque la modificabilidad ha constituido el principal atributo de calidad que ha dirigido el diseño de la arquitectura, no ha sido el único, pues también se ha tenido en cuenta el rendimiento. ATAM da la posibilidad de que surjan cuestiones relativas a atributos de calidad que no se habían considerado inicialmente.

En segundo lugar, ATAM es un método de evaluación más moderno que SAAM, al que engloba, por lo que todo lo que pueda evaluarse con SAAM debe poder evaluarse también con ATAM.

7.2 Evaluación de la Arquitectura

ATAM, como se mencionó al inicio, es un método para evaluar arquitecturas software que consta de 9 pasos agrupados en 4 fases, mismas que se seguieron para la evaluación de la arquitectura propuesta. A continuación se describe brevemente en qué consistió cada una de ellas (Clements et al., 2002b).

Fase 0: Presentación. Explicación del proceso de evaluación ante todos los participantes

Se presentó ATAM a los participantes en la evaluación.

Además se presentaron también las motivaciones para desarrollar el prototipo.

Y finalmente se presentó la arquitectura propuesta.

Fase 1: Investigación y Análisis. Identificación de los atributos de calidad relevantes y soluciones arquitectónicas con el equipo de desarrollo.

Se identificaron los principales mecanismos arquitectónicos utilizados en el diseño.

Se generó la lista de los atributos de calidad.

Se realizó un análisis de las soluciones arquitectónicas en función de los atributos de calidad con mayor peso.

Fase 2: Pruebas. Comparación de los resultados con las necesidades de todos los interesados

Se llevo a cabo un proceso de identificación y priorización de nuevos escenarios.

Se analizaron las soluciones arquitectónicas en función de los nuevos escenarios con mayor peso.

Fase 3: Informes.

Se presentaron los resultados de ATAM 9. Finalmente se generó un informe con los resultados de la evaluación, este informe se constituye en unalista dividida en 3 aspectos: los puntos de cuidado, los puntos sin riesgosy los aspectos delicados o sensibles, con el fin de que trabajos posteriores revisen esta lista y enfaticen el desarrollo en función de estos aspectos.

Lista de Puntos de Cuidado

A continuación y luego de la evaluación ATAM9 se presentan los puntos de riesgo que puede presentar la arquitectura propuesta:

1. La planificación puede requerir mucho tiempo de cálculo, por lo que se deben elegir planificadores que funcionen con lenguajes compilados.

2. Si habitualmente existen varias acciones correctas para una situación, se puede dar el caso de que haya que recalcular muy a menudo la siguiente acción antes de dar una respuesta al alumno, lo que puede hacer que el prototipo vaya más lento de lo requerido.

3. En caso de no poder utilizar respuestas prediseñadas, es posible que sea necesario realizar consultas al agente global o mundo que precisen razonamiento sobre la ontología, lo que podría llegar a requerir más tiempo de los tres segundos propuestos para dar esa respuesta.

4. Al existir un único tutor las acciones se procesan en secuencia. Existe el peligro de que los retardos en la red de dos alumnos o más, sean distintos y las acciones lleguen desordenadas. Queda abierta la sugerencia de utilizar marcas de tiempo para las acciones.

5. Si el elemento no está preparado para funcionar dentro de un agente mediante la prestación de servicios, entonces aumentará el tiempo necesario para integrarlo debido a la adaptación necesaria.

6. La separación arquitectónica entre la estrategia de tutoría y el modelo del estudiante no implica eliminar la dependencia entre ellos, por lo que la sustitución de uno de ellos puede afectar mucho al otro. Puede suceder que el modelo de estudiante no proporcione la información necesaria para la tutoría para tomar decisiones, o que la tutoría no proporcione la información necesaria para elaborar el modelo del estudiante.

7. La solicitud de servicios con paso de mensajes entraña el riesgo de que, aunque la información sea correcta a nivel sintáctico, puede no serlo a nivel semántico, por lo que debe comprobarse dicha corrección. El uso de un planificador externo conlleva cierta dependencia de la forma en que hay que modelar el dominio, los operadores y, en general, todos los datos, por lo que los agentes que manejen esos datos se pueden ver afectados por un cambio en el planificador. Para evitarlo debe diseñarse un formato para ellos que no se base en la representación del planificador.

8. Dependiendo del algoritmo utilizado, la representación de las rutas puede variar, generalmente en forma de una lista de coordenadas o una lista de puntos de interés. Para

que el prototipo no dependa de este formato, se deberá disponer de un mecanismo que permita traducir coordenadas a puntos de interés y viceversa.

9. En general, un cambio del simulador se debe a un cambio en el dominio de aprendizaje. Esto siempre supone un riesgo alto, pues la naturaleza de los simuladores puede ser muy distinta. Por tanto, la manera de integrar un nuevo simulador queda planteada como una cuestión abierta. Considerando el dominio de aprendizaje trabajado para este prototipo el simulador funciona como se ha planteado, pero no se debe de dejar de considerar este punto.

10. Al tratarse de una arquitectura cognitiva incorpora mecanismos complejos para dotar al agente de rasgos de personalidad y emociones, por lo que puede constituir un elemento demasiado pesado una vez que queden todos los diálogos y reacciones incorporadas, lo que puede afectar el rendimiento del prototipo, especialmente si se utiliza en más de un agente.

11. El cambio de la estrategia de tutoría es una modificación de una parte fundamental del prototipo, por lo que es probable que su eliminación requiera un análisis cuidadoso de todas las interacciones que tiene con otros elementos del prototipo.

12. Existe la posibilidad de que el razonador que trabaja sobre la ontología no proporcione la información necesaria para que el generador de lenguaje natural pueda construir frases naturales con sentido, lo que implicaría un altísimo riesgo para la adaptación y empatía.

13. La alternativa más realista para que el prototipo se ejecute en un dispositivo móvil es que sea sólo el EV el que lo haga, conectándose por medio de una red wifi al centro de mensajes. Aún así, es necesario hacer uso de un EV simplificado y con características muy reducidas respecto a lo que puede ejecutarse en una computadora personal, por esta razón no se ha planteado como posibilidad o recomendación la consulta del EV en dispositivos móviles, a pesar de que sea esta la tendencia.

14. Si se desea instalar la aplicación completa en el dispositivo móvil, hay plataformas, como JADE, que se ejecutan en estos dispositivos. Sin embargo, dada la menor capacidad de un dispositivo móvil en comparación con computadora de escritorio o portátil, habría que eliminar muchas funcionalidades, especialmente las más complejas, como la planificación y la planificación de rutas.

15. Abrir la plataforma de agentes supone la posibilidad de dejar entrar agentes maliciosos o simplemente inconvenientes que pueden provocar un mal funcionamiento del prototipo. Abrir el prototipo implica implementar medidas de seguridad basadas en la certificación

del origen de los agentes para que sólo los agentes autorizados puedan entrar en el prototipo. Por lo que tampoco cabe la posibilidad de que sea un prototipo abierto.

Lista de aspectos o puntos sin Riesgo

1. En caso de respuestas prediseñadas, la acción se ejecuta rápidamente, pusto que tres segundos es suficiente para dar una respuesta.

2. Al prestar el servicio desde un comportamiento, se puede desactivar dicho comportamiento de manera dinámica sin consecuencias.

3. El hecho de situar un elemento dentro (o detrás) de un agente permite que se utilicen los mecanismos introducidos para facilitar la modificabilidad, como el paso de mensajes o la edición-suscripción.

4. La comunicación a través de la solicitud de servicios con intercambio de mensajes debilita la relación entre quien solicita el servicio y quien lo presta, pues ésta se establece en tiempo de ejecución, lo que favorece la modificabilidad.

5. La comunicación a través del Centro de Mensajes mediante un mecanismo de suscripción por tipo de mensaje hace que el resto del prototipo no se vea afectado si se sustituye un elemento por otro que proporcione la misma información. De la misma manera, el nuevo elemento puede recibir todos los mensajes que necesite sin preocuparse de quién hace el envío.

6. La introducción de un nuevo agente puede realizarse utilizando los mismos mecanismos que con el resto de agentes, por lo que técnicamente sólo está supeditado a que existan los servicios que le proporcionen la información que necesita para realizar su trabajo.

7. La eliminación de un agente no genera riesgos siempre que nos aseguremos de que sus servicios no los utilizaba ningún otro agente.

8. Haciendo uso de los mecanismos existentes se puede introducir el nuevo comportamiento, que puede solicitar los servicios que requiera de otros agentes.

Lista de aspectos Delicados o Sensibles

1. Para tardar menos tiempo en proporcionarle una respuesta al estudiante, cuando se lleva a cabo una acción se le solicita al planificador la siguiente acción a realizar, de manera que se tenga preparada para cuando el estudiante la realice. Si hay varias acciones posibles, quizá que el estudiante realice otra.

2. Dependemos de la potencia de cálculo de la máquina en que se realice la ejecución y de la eficiencia del planificador.

3. No siempre se puede hacer uso de respuestas prediseñadas.

4. El escenario general del prototipo depende de muchos factores que no son arquitectónicos, como la implementación del motor de render y la complejidad del modelado 3D que fue dibujado para el tutor y el tiempo de ejecución de las secuencias prediseñadas.

5. Para que dos acciones simultáneas se procesen bien dependen en gran medida de que lleguen en el orden adecuado al STI, en caso de que este orden sea importante.

6. Para poder cumplir con la estimación son necesarias varias cuestiones, como que la estrategia de tutoría se encuentre implementada y que admita el enfoque orientado a servicios que se ha planteado como solución. También depende de la compatibilidad con el modelo del estudiante en turno.

7. Para poder realizar la sustitución se necesita que el nuevo modelo sea compatible con la estrategia de tutoría, y también que se le proporcione la información necesaria en un formato que al menos pueda transformar en el que utiliza.

8. El hecho de que el algoritmo de cálculo de rutas trabaje con grafos o con coordenadas puede hacer nacesaria la realización de conversiones entre ambos formatos.

9. Debe ser posible hacer que desde el EV se envíe al Centro de Mensajes la información que necesita el STI y, a ser posible, en el mismo formato que se esté utilizando.

10. Añadir un tutor virtual con aspecto y comportamiento simples se puede realizar en un tiempo corto, por lo que el cumplimiento está supeditado al realismo del que se quiera dotar al personaje y a lo que permita hacer el EV.

11. Se depende de la existencia de otros elementos, como una representación virtual con cierta capacidad expresiva.

12. La estrategia de tutoría es una pieza fundamental dentro del STI. Aunque la única dependencia notable se encuentra en el modelo del estudiante, pues le proporciona la información para completarlo, esta sustitución puede suponer un cambio significativo en el funcionamiento del prototipo.

13. La introducción del modelado de grupo requiere extender el modelo del estudiante con aspectos que reflejen el comportamiento del grupo. También requiere que la estrategia de tutoría pueda tener en cuenta aspectos relativos al grupo en su conjunto.

14. Transformar un sistema de aprendizaje depende de que el agente experto maneje el conocimiento acerca del mismo.

15. La distribución en distintas máquinas debe realizarse cuidadosamente para que no se produzcan errores por una mala sincronización entre los agentes. Los mejores candidatos a ser distribuidos son los agentes de estudiante y los de comunicación, que son los que se replican en función del número de estudiantes.

16. Replicar los agentes por cada sesión de aprendizaje supone que se puede elevar mucho su número, con la consiguiente penalización para el rendimiento de la aplicación. Si, para solucionarlo, se opta por distribuir los agentes, probablemente tenga más sentido ejecutar cada sesión de aprendizaje por separado, en máquinas diferentes.

17. Si se ejecutan varias sesiones juntas, los mensajes deberían identificarse para que se sepa a qué sesión pertenecen y quién los debe recibir. Existe la posibilidad de que este cambio, aunque de forma ligera, afecte a todos los agentes del prototipo.

7.3 Resultados de la Evaluación de la Arquitectura

La evaluación de la arquitectura descrita ha proporcionado resultados muy parecidos a lo que se esperaba antes de su realización y que se constituyó como la esencia de la hipótesis y los atributos de diseño y construcción.

Tal como afirman los autores del método ATAM en (Clements et al., 2002b), los resultados de la realización de la evaluación dependen de los participantes en dicha evaluación.

Aunque a primera vista podría parecer un inconveniente, puesto que los resultados no son objetivos, en realidad sucede lo contrario. Teniendo en cuenta que los participantes en la evaluación son los interesados en el desarrollo del posterior sistema, los factores que hacen acto de presencia son los que conciernen a las personas implicadas, y son por lo tanto los que de una u otra manera afectan a la arquitectura. No habrá, por el contrario, factores ajenos a los interesados que intervengan en el proceso de evaluación.De esta manera, muchos de los puntos significativos identificados en la evaluación han sido previamente tratados en el proceso de diseño, por lo que se puede esperar que sus implicaciones y sus efectos sean los ya descritos en la documentación de la arquitectura.

Sin embargo, la participación en la evaluación de miembros ajenos al equipo de desarrollo, así como la generación de escenarios de evolución, han permitido identificar algunos factores que no se habían tenido en cuenta previamente y que pueden conducir a algunas mejoras en el diseño

de la arquitectura.

La generación de escenarios de evolución también ha dado pie a la identificación de puntos sensibles en lo relativo a la flexibilidad de la arquitectura, aunque algunas de las modificaciones propuestas llevan al prototipo hasta unos límites que hacen poco probable que la arquitectura se adecué a los fines previstos, a pesar de que se pueda llegar a obtener un sistema en funcionamiento.

7.4 Evaluación de la Interfaz Gráfica de Usuario

Para descubrir el «mejor diseño» y para ir optimizando progresivamente nuestras interfaces necesitamos ver cómo los usuarios trabajan, cómo ejecutan las tareas que hemos diseñado, y todo ello siguiendo tres “normas” ineludibles, que Nielsen (2001) destaca siguiendo la línea que hemos marcado hasta ahora:

1) Observar qué hace y cómo se comporta realmente la gente.
2) No creer aquello que dicen que hacen.
3) No creer aquello que la gente predice del futuro y de lo que pueden llegar a hacer.

Cuando se está creando un producto o servicio interactivo, puede suponerse que éste será útil y fácil de usar. Solemos creer que hemos colocado el botón en el lugar más adecuado, o que se han ordenado los contenidos de la manera más lógica posible.

Todas las ideas preconcebidas desaparecen en el momento en que un usuario se encuentra por primera vez con este producto o servicio. Quizá no encuentre el botón porque pensaba que debía estar en un lugar determinado de la interfaz, no logra entender la aplicacón o bien no encuentra conceptos básicos dentro del árbol de contenidos, que han sido estratégicamente pensados y analizados.

Es entonces cuando nos damos cuenta de que evaluar nuestros productos o servicios con usuarios reales puede ser una manera adecuada de saber si vamos por el buen camino, es decir, si el producto o el servicio que desarrollamos es fácil, eficiente y agradable de usar. Por estas razones se decidieron hacer tanto el diagnóstico, como la evaluación de usuarios y las pruebas heurísticas.

7.4.1 Muestreo

Realizar el muestreo es tomar una porción de una población o universo como representativa de dicha población, indica Kerlinger (Kerlinger, F., 1994). El muestreo aleatorio es el método de seleccionar una porción o muestra de una población que permite que cada miembro de la población o universo tenga la misma oportunidad de ser elegido. Formalmente, es el método de elegir una porción de una población que permite que todas las muestras de un tamaño fijo n tengan la misma probabilidad de ser seleccionadas. En este caso, la muestra coincide con la población.

Para el prototipo que nos ocupa y la realización de las pruebas de usuario, los estudiantes fueron asignados al azar a los grupos experimentales y los tratamientos experimentales también fueron asignados al azar a los grupos. Se dispuso de 25 alumnos, 12 de ellos son del turno matutino y 13 de turno verspertino. Los estudiantes del turno matutino se distribuyeron aleatoriamente en 2 grupos A y B. A partir de la lista alfabética de los alumnos, se decidió qué grupo le corresponde mediante el uso de la tabla de números aleatorios que ofrece Kerlinger (Kerlinger, F., 1994). El alumno se asignó al grupo Matutino 1A (M1A) si el número que le corresponde es impar o el grupo Matutino 1B (M1B) si el número es par. Todos los alumnos del turno vespertino integraron el grupo Vespertino 1A (V1A). Se puso como restricción primero, que no se podían intercambiar alumnos de los turnos matutinos y vespertino debido a preferencias y restricciones establecidas por los propios alumnos al momento de inscribirse.

TURNO	GRUPO	No. de alumnos asignados
Matutino	1A (M1A)	6
Matutino	1B (MiB)	6
Vespertino	1A(V1A)	13

Tabla 7.1: Dis*tribución de alumnos para la prueba.*

Así como la restricción de no intercambiar alumnos entre los turnos podría influir en el resultado final de la investigación, se deja constancia que puede también existir influencia de otros factores (como por ejemplo el sexo o la edad), pero no se considerarán a los efectos de esta investigación. Se tuvo en cuenta también la posible existencia de otros factores, como por ejemplo el hecho de completar el semestre. El salón de clase, el equipamiento y el docente son idénticos para todos los grupos. La asignación de tratamientos se realizó en forma aleatoria también.

En forma similar a la asignación de alumnos a los grupos, a cada uno de los grupos M1A, M1B y V1A le corresponderá un número aleatorio de la tabla, que según sea múltiplo de 3 será el tratamiento 1, si es múltiplo de 3 más uno será el tratamiento 2 y si es múltiplo de 3 más dos le

corresponderá el tratamiento 3. Si ya fue asignado un tratamiento, se seleccionará el siguiente número aleatorio. Al aplicar este proceso, resultó que el grupo V1A recibió el tratamiento 1 (mantener el curso actual), el grupo M1A tuvo el tratamiento 2 (recibir la capacitación) y el grupo M1B el tratamiento 3 (la capacitación y el entorno).

Grupo	Tratamiento
V1A	Tratamiento 1 (mantener el curso actual)
M1A	Tratamiento 2 (recibir capacitación con DECANO)
M1B	Tratamiento 3 (recibir capacitación y utilizar el EV)

7.4.2 Tamaño de la muestra

Como indican Mason y Lind (Mason, R. *et al.,* 1998), los factores que determinan el tamaño de la muestra son: el grado de confianza seleccionado, por lo general de 0,95 o de 0,99; el máximo error permisible, es el máximo error tolerable en un nivel de confianza específico y la variación de la población.

En este caso se han tomado todos los estudiantes, pues no se puede intervenir en el tamaño de la muestra dado que los grupos son fijos (matutino y vespertino) y no hay cantidad excesiva de alumnos. Así pues, no se ha extraído una muestra, sino que se ha tomado el universo como muestra, considerando la recomendación de Andina (2002)

Los datos arrojados se consideran primarios, o sea obtenidos directamente, de toda la población.

7.4.3 Instrumentos y Recolección de Datos

El material de apoyo, el diseño de la interfaz como un modelo visual y la célula básica del entorno estuvieron disponibles, en versiones preliminares, durante todo el semestre agosto/diciembre 2013. Para la experimentación se seleccionaron algunos temas sencillos del dominio del prototipo DECANO, que aborda las competencias básicas para la investigación.

Para la recolección de datos se realizaron ejercicios o pruebas a todos los alumnos, con el objetivo de medir la naturaleza y grado de diferencias individuales (Salkind, N., 1998). Posteriores a la presentación y los sondeos o diagnósticos, se propusieron 2 pruebas (sin límite de tiempo) sobre toda la población, la primera de ellas, en la semana 5-6 del curso, para identificar características

y el punto de partida y la segunda, al finalizar el semestre, para analizar los posibles efectos del uso (o no) del material y del entorno. Se tuvo en cuenta, además del efecto Hawthorne citado, la recomendación de Aaker y colegas (1989) acerca de los efectos de la medición del "antes", pues puede alertar a los entrevistados de que están siendo estudiados, aumentando la curiosidad y atención. Este hecho se supone que no afectará pues en forma sistemática en todos los cursos de Metodología de la Investigación de la Universidad Autónoma de Querétaro es común realizar encuestas y, o, entrevistas a los alumnos, hecho que conocen.

Se realizó el "pre-test o pretesteo" de instrumentos con los estudiantes, habituados con el uso de tecnologías y que se consideró podían colaborar, tanto en el diagnósitico de elementos sintáctico-simbólicos y funcionales de las interfaces gráficas, así como en la detección de posibles problemas de comprensión de las preguntas o ejercicios planteados. Las pruebas preliminares, tanto de la percepción sobre interfaces y sus elementos visuales, así como la relativa a las competencias básicas de investigación y desarrollo de proyectos, no fueron anónimas porque se pretendía mantener la vinculación con la segunda prueba, ya sobre el prototipo DECANO. También se pIdió la colaboración a docentes de la materia a los efectos de validar el contenido, o sea el grado en que una prueba representa el universo de reactivos del cual se extrajo y para evaluar la utilidad de las pruebas que muestrean un área de conocimientos en particular, como lo demuestra Salkind, (1998).

7.4.4 Evaluación y Pruebas estadísticas

Como ya se indicó, la variable independiente es el uso del entorno y las variables dependientes son: la comprensión del problema, las formas de resolver un problema y la transferencia del conocimiento.

Se quería determinar cómo varía la comprensión, resolución de problemas y transferencia de conocimiento en lo que se mantiene estable, que es el entorno. Se trataba de determinar si hay una relación causa-efecto. Por ejemplo, se buscaban efectos del estilo: "repercute en que el alumno se vuelve más habilidoso en la búsqueda de soluciones" o "repercute en que el alumno mejora su capacidad de resolución de problemas". Adicionalmente se pretendía identificar la influenicia de la interfaz como aterfacto mediador en el proceso de enseñanza-aprendizaje.

Estadística no paramétrica

"El término 'estadística no paramétrica' no tiene una definición estándar que acepten todos los estadísticos", señalan Mendenhall y colegas (Mendenhall, W. *et al.*, 1994). Definen los métodos *paramétricos* como aquellos que se aplican a problemas para los cuales se especifica(n) la(s)

muestra(s), con excepción de un número finito de parámetros. Así, los métodos *no paramétricos* se aplican en todos los demás casos. O sea, los métodos no paramétricos son independientes de las distribuciones de la población y de los parámetros asociados. Son especialmente útiles para datos no numéricos, por ejemplo, cuando se establecen preferencias (Spiegel, M., 1991).

En este caso, no se realizan supuestos de normalidad y los tamaños de la muestras son relativamente pequeños (n < 23), por lo que de acuerdo con Meneses (Meneses, B., 1997), es recomendable utilizar pruebas correspondientes a la estadística no paramétrica.

También se harán análisis cualitativos, pues se tratará de "decir dos veces las cosas", se otorga una prioridad absoluta a lo cuantitativo frente a lo cualitativo. Más aún, pues incluso se piensa que una disciplina cualquiera no es realmente científica mientras no use conceptos cuantitativos.

7.4.5 Evaluación y escalas

"La evaluación es la reunión sistemática de evidencias a fin de determinar si en realidad se producen ciertos cambios en los alumnos y establecer el grado de cambio en cada estudiante", refieren Bloom y colaboradores (1975). "Para conocer el peso de un objeto es preciso ponerlo en la balanza. Se realizaron pruebas que incluían ejercicios. Se utilizó una escala ordinal de Salkind (1996), es decir, las variables se pueden ordenar sobre algún tipo de continuo. A cada elemento de esas pruebas se le asignó un valor de una escala de clasificación.

Para la elección de este tipo de escala se tuvo en cuenta las ventajas y desventajas de las mismas. Como señala Kerlinger (1983), una escala de clasificación es un instrumento de medición que requiere que el observador asigne el objeto a categorías o espacios continuos que tienen valores numéricos asignados a ellos. Como ventajas tienen que son fáciles de construir y rápidas en su uso, pero como desventajas cita el efecto de halo o la tendencia a clasificar un objeto en la dirección constante de una impresión del objeto; el error de severidad o la tendencia general a clasificar a todos los individuos bajo en todas las características o el error de lenidad, que es la tendencia general opuesta. Otra desventaja posible es el error de tendencia central, que es la tendencia general a evitar todos los juicios extremos y a calificar a un nivel intermedio en una escala de clasificación.

Para tratar de minimizar los posibles efectos negativos citados, la valoración de cada trabajo propuesto fue realizada en forma independiente por 2 docentes de la materia, confrontándose y unificándose luego los resultados.

7.4.6 Pruebas estadísticas

Los datos fueron ordinales, como se indicó. De esta forma es posible realizar conteos de frecuencia y también es apropiado calcular la media. Además, las pruebas estadísticas libres de distribución y que necesitan datos cuanto menos de nivel ordinal, que se aplicaron a los datos recolectados fueron, como sugieren Weiers (1986) y Mendenhall y colegas (1994):

prueba U de Mann-Whitney (1994): utiliza la suma de los rangos de dos muestras. Se trataba de analizar si las distribuciones de frecuencias relativas poblacionales de dos muestras son idénticas o no.

prueba H de suma de rangos o prueba de Kruskal-Wallis para comparar k muestras independientes. El objetivo fue probar si las k distribuciones poblaciones eran idénticas (Mendenhall, W. *et al.,* 1994).

Si bien es cierto que, como indican Freund y colegas (1990), cuanto menos se supone, menos se puede inferir, también es cierto que cuanto menos se supone, más se amplia la aplicabilidad de un método. Como un elemento más a analizar, se compararon los totales de aprobación y deserción del presente curso con los datos históricos de cursos similares ofrecidos en la Universidad desde el año 2010 considerando turno matutino y vespertino.

	2010-1	2010-2	2011-1	2011-2	2012-1	2012-2	2013-1
aprobados	19	17	17	18	16	17	16
reprobados	1	3	4	5	4	3	2
deserciones	2	1	2	0	3	3	4
total alumnos	22	21	23	23	23	23	22

Tabla 7.2: comparativa de índices de aprobación y reprobación del curso

aprobados	19	17	17	18	16	17	16	Media 17.08743571
porcentaje %	86.36	80.95	73.91	78.26	69.56	73.91	72.72	76.17%

Tabla 7.3 Cálculo de la Media para el índice de aprobación y porcentaje sobre el promedio de alumnos

Este comparativo nos ofreció un diagnóstico preliminar con respecto a la asignatura y permitió identificar como primer indicador el índice de aprobación promedio de la asignatura, que tras los primeros datos nos sitúa en un 76%, es decir que existe un índice de reprobación y abandono del 24%. Tratándose una asignatura de competencias fundamentales para la formación profesional,

puesto que no sólo se imparten contenidos relativos a la metodología para realizar investigación, sino también se ofrecen temáticas relativas al campo profesional, la presentación de informes, el desarrollo de un curriculum vitae, la formación de competencias para el liderazgo de proyectos y la organización y gestión de proyectos, se considera que estos porcentajes son críticos, puesto que repercuten en la eficiencia terminal.

Este primer comparativo se contrastó con los resultados de los alumnos al finalizar el experimento y un elemento altamente significativo, fue el hecho de que los dos grupos con que tuvieron contacto con el programa no abandonaron la asignatura, lo que repercutió de forma directa en el índice de decersión.

Adicionalmente a este diagnóstico preliminar, se realizaron otras pruebas y se hizo un seguimiento permanente con una bitácora.

	2013-2
aprobados	20
reprobados	2
deserciones	0
total alumnos	22

Tabla 7.4: ïndice de aprobación, reprobación y decersión al finalizar el experimento.

Primera prueba/ Prueba diagnóstica.

Como primera prueba o prueba de sondeo (“prueba diagnóstica” según los términos de Bloom (1975) a aplicar a todos los alumnos se propuso la siguiente:

Las pruebas consistieron en la evaluación por parte del usuario con respecto a la viabilidad del modelo propuesto y sus reglas determinado con esto la aceptación del modelo, el apoyo que se puede lograr con el uso de la herramienta respecto a los objetivos esperados, la manera en que se promueve la colaboración a través de medios remotos y la disponibilidad de realizar trabajos colaborativos, la evaluación previa al prototipo formal con respecto a la interfaz de usuario, facilidad de uso, navegación y acceso a datos , etc.

La figura 7.4 muestra un resumen de la evaluación obtenida.

En general los usuarios evaluaron el desempeño del prototipo como muy aceptable, sin dejar de tomar en cuenta que la evaluación se hace al prototipo.

Algunas otras observaciones que los usuarios expresaron fuera de formato fueron entre otras, que se requieren mas herramientas que permitan interactividad más directa, que el usuario pueda

modificar su información particular, que el prototipo cuente con herramientas que permitan al usuario realizar cambio a los objetos directamente desde el prototipo.

A manera de resumen se puede especificar que como resultado a las pruebas de Módulos, Integración y Función se obtuvieron las interfaces generales que determinarán las plantillas cuando se desarrolle el sistema formal. Al programar los módulos de manera independiente pudo observarse que la programación con la base de datos y los datos que se obtienen, determinan el tipo de controles que se utilizarán en la programación basada en Web. Los resultados de la prueba de Integración permitieron especificar adecuadamente los vínculos entre todos los módulos y controles de tal manera que en una sesión de usuario se pudiera navegar entre todas las opciones que le permita su rol.

Los resultados de las prueba de Desempeño y Aceptación determinaron la viabilidad de la programación del sistema formal. Los resultados obtenidos en una encuesta realizada a los usuarios que probaron el prototipo, determinaron que el grado de utilidad es muy aceptable, lo que indica que el prototipo del modelo puede ser implementado. Los resultados de la prueba de Instalación permitieron corroborar que el mejor ambiente para la instalación del prototipo es vía Web contando con un servidor y un lenguaje de programación que no generen costos para la institución.

Este tipo de análisis permite obtener información objetiva y subjetiva del usuario, y comprobar de una manera más fiable si la base de datos evaluada resulta usable.

El perfil medio de los usuarios de este estudio responde a las siguientes características:

– Edad: entre 20 y 25 años.

– Formación o área profesional: estudios universitarios en Artes Visuales.

– Conocimientos de Internet: conocimientos avanzados.

– Antigüedad de navegación: entre 5 y 10 años.

– Horas diarias de navegación: más de 4 horas de navegación.

– Uso de Internet: apoyo para el estudio, socialización y entretenimiento, primordialmente

– Conocimientos sobre aplicaciones y entornos educativos y de entretenimiento: avanzado.

El objetivo de este estudio con usuarios consistió en analizar la eficiencia, eficacia y satisfacción con que trabajan los usuarios habituales de estos sistemas. Para ello se utilizaron tanto, el test de

usuario, el cuestionario y la entrevista.

a) Eficiencia

Por medio de la observación directa basada en pruebas con usuarios, los evaluadores expertos pudieron determinar si los usuarios conseguían completar las tareas encomendadas; en concreto, se consideró que el prototipo resultaba efectivos si se el usuario lograba familiarizarse y adaptarse conla interfaz gráfica.

b) Eficacia

La eficacia se evaluó a través del siguiente cuestionario de respuestas cerradas («sí», «no» y «a veces»):

1. ¿Le ha costado mucho tiempo entender e identificar los íconos, botones y menús del prototipo?

2. ¿Le parece adecuada la organización de la interfaz?

3. ¿Le ha parecido fácil de usar el prototipo, tanto el entorno en red, como las herramientas instaladas?

4. ¿Ha realizado las tareas de forma rápida?

c) Satisfacción

El grado de satisfacción del usuario se evaluó mediante una entrevista cuyas repuestas se habían categorizado como «sí», «no» y «a veces».

1. ¿Le ha parecido fácil de usar el prototipo?

2. ¿Le ha resultado fácil interpretar los iconos, los textos de los menús y la estructura del prototipo?

3. ¿Resulta fácil acceder a herramientas y funciones?

Y para el pre-diagnóstico, se aplicó de forma tradicional también un cuestionario a todos los estudiantes con el fin de identificar concocimiento preliminar sobre el dominio.

Nombre:	Fecha:	Grupo:
	Horas de trabajo semanales: (0: no trabaja)	
Conocimientos previos sobre metodología y métodos de trabajo. Detallar		
La finalidad de esta evaluación es realizar un sondeo de nivel y de estrategias de trabajo. Se agradece su colaboración. Por favor escribir con letra clara y utilizar todo el espacio que necesite para las respuestas.		
1) SITUACIÓN UNO: Se tiene una lista con todos los alumnos aprobados en la asignatura durante los últimos 4 años. Cada alumno conoce sus calificaciones. Se desea obtener un listado alfabético de aquellos con las más altas puntuaciones. Indicar lo más detalladamente posible los pasos que seguiría para resolver esta situación.		
2) SITUACIÓN DOS: Ud. está colaborando en el desarrollo de un sistema informático y le sugieren que incluya un elemento X que usted desconoce. ¿Qué haría? Detalle todos los pasos que seguiría.		

3) SITUACIÓN TRES: Se desea realizar un sistema para la administración de una tienda de mascotas. Enumere posibles temas y elementos que usted considera que debe incluir.	

Objetivo y evaluación de las preguntas:

El principal objetivo era identificar si se llevaba a cabo comprensión de un problema, es decir " dado un problema, ver que haga lo que se pide" y obviamente considerar el nivel de profundidad, especificidad y detalle con que lo resolvía, además de que la actividad servía como parte del preámbulo para el desarrollo de competencias básicas en investigación, que es el centro de la propuesta.

7.4.7 Bitácora de la Experimentación

Preparación: Distribución de alumnos en los grupos

Previo al comienzo del semestre en agosto de 2013, se realizó la distribución aleatoria de los alumnos entre los grupos matutinos y se mantuvo el único grupo vespertino. Se evitó informar a los alumnos del desarrollo de las pruebas y del entorno, para evitar el efecto "de prueba", o sea la conciencia de estar en una prueba según Aaker y colegas (1989), el efecto Hawthorne (Pazos, J., 2002b) y el efecto de la medición previa, pues se aclaró en las clases que es política del curso hacer pruebas de seguimiento (experimento ciego).

Como ya se mencionó se propusieron tres tipos de evaluación que se gestionarían a partir de la presentación y luego del uso del prototipo

a) Evaluación diagnóstica. En donde no sólo se verificaron los conocimientos previos del estudiante sobre la materia y el uso de las TIC.

b) Evaluaciones parciales. Realizadas a lo largo del trimestre, teniendo como base las diferentes estrategias y actividades, a partir de las cuales se emplearo diversos instrumentos como resúmenes, mapas conceptuales, mapas mentales, participación en foros, ensayos, etc. La idea es considerar el proceso de trabajo, no sólo los resultados finales.

c) Evaluación final. Que en apoyo a las actividades realizadas a nivel presencial, permiten verificar los conocimientos adquiridos a lo largo del semestre.

De manera paralela a estos momentos de evaluación, se considera importante fomentar la autoevaluación del estudiante, porque esto supone el que adquiera una mirada crítica a su trabajo y sea consciente de los objetivos buscados.

El 11 de septiembre 2013 se realizó la prueba de sondeo a todos los alumnos durante su correspondiente hora de clase, o sea en condiciones similares. Se trató de minimizar la interacción entre el examinador y los estudiantes. No se ofreció ningún tipo de sugerencia o recomendación, ni antes, ni durante la prueba. Tampoco se avisó previamente de su realización. Para la valoración de las respuestas, se utilizó la escala propuesta en el diseño y cada respuesta fue evaluada por 2 docentes de la materia, el asignado y otro con experiencia en esa misma asignatura. En la primera semana de octubre se brindó realimentación a los alumnos acerca de la prueba, discutiéndose individualmente los resultados durante las clases. La mayoría de los alumnos mostró interés sobre sus propios resultados.

Adicionalmente a esta prueba, se realizó una pequeña matriz de asociaciones, con el fin de identificar el valor simbólico de las interfaces y la memorabilidad y recordación de los íconos y conceptos comúnmente usados en las interfaces gráficas de usuario (consultar ANEXO: Evaluación y Pruebas).

7.4.8 Material de Gestión del Conocimiento y Resolución de Problemas.

En los grupos matutinos se utilizó el material de resolución de problemas. En la primera semana de octubre se dedicó aproximadamente una hora de clase para la resolución del ejercicio sobre la estructura y redacción de los objetivos. (Para una descripción detallada del modelo y las taxonomías, consultar los ANEXO: HERRAMIENTAS DEL Prototipo). Se entregó a cada equipo de 2 o 3 estudiantes un juego de material impreso y se explicó el problema. Cada equipo intentó una solución. En los dos grupos matutinos se realizaron similares observaciones. En el grupo M1A hubo 4 grupos y en el M1B 2 grupos. Algunos grupos comenzaron a trabajar antes de tener en claro las reglas, lo que los llevó a soluciones apresuradas y erróneas, otros equipos (2 en cada grupo) sugirieron “aplicar inducción completa”, simplificando inicialmente el problema. Sólo un equipo desistió. Finalmente, todos los grupos que no desistieron lograron solucionar el problema.

En la clase siguiente, se explicó nuevamente en los grupos matutinos cuál fue el objetivo de la prueba de sondeo y se vinculó con el ejercicio de los materiales impresos. También se presentó

material sobre resolución de problemas. Se analizaron los elementos que componen un problema, cuáles son las estrategias recomendadas y cuáles son algunos posibles problemas que hay que enfrentar.

Durante todo el período de clases se repasaron los conceptos e ideas fundamentales de gestión del conocimiento y resolución de problemas. En particular, se trató de que por lo menos una vez a la semana se dedicara en la clase cierto tiempo (mínimo 15 minutos) a la discusión y tratamiento de dichos temas. Se utilizaron, entre otros materiales, los materiales incluidos en los Apéndices y una antología de lecturas propuesta por los docentes de la asignatura.

Entorno

Se instaló en un laboratorio con 20 máquinas el prototipo del Entorno DECANO, con su respectiva sincronización a la WEB, de tal manera que cada estudiante, además de acceder al espacio colaborativo en red, puediera también ingresar al prototipo DECANO y trabajar en sincronía.

El laboratorio estu disponible de 8.00 a 2:.00 hs. de lunes a viernes. La primera versión del entorno cuenta con varios tipos de conocimiento, y se incluyeron más de 50 conocimientos diferentes y relacionados sobre el tema. En la tercera semana de octubre se realizó la presentación formal del prototipo. Se hizo una presentación en el salón de clases y se concurrió al laboratorio para que lo probara cada estudiante del grupo M1B.

Luego de la presentación y uso inicial, se les pidió que cada uno entregara por escrito una primera impresión sobre el prototipo. No todos los estudiantes incluyeron comentarios. Los que sí lo hicieron indicaron lo siguiente:

> *“El programa me parece muy bueno e innovador ya que nos puede contestar dudas que tengamos, se facilita encontrar lo que buscas y a la vez puedes ver cosas nuevas”*
>
> *“Se entiende fácil y de primera impresión parece bueno”*
>
> *“Me parece muy bueno y útil, pero tengo que aprender a manejarlo mejor, es decir dedicarle más tiempo por que tiene muchas cosas”*
>
> *“Me gustó el programa y con todo gusto lo voy a utilizar. Es interesante la cantidad de relaciones que tiene y si va a tener más, mejor”.*

“El programa está bien y me parece de gran utilidad, debido a que no es

complicado usarlo. También tiene cosas muy interesantes”.

“El programa está muy bueno. Me gustó la idea de crear una especie de enciclopedia de los conocimientos. El tipo de conocimiento que más utilicé fue el heurístico, pues son útiles a la hora de sentarse a escribir”

“Está muy interesante, es bueno poder adquirir más didácticamente los conocimientos”.

A efectos de fomentar el uso del entorno, se le entregó a cada estudiante que lo solicitó (13 en total) una clave para que ellos mismos conserven su bitácora. Se les dio instrucciones de cómo registrar la bitácora de trabajo de cada prueba y cómo hace consultas sobre su desempeño y el uso del prototipo. Periódicamente en clase se les fue consultando sobre los avances en el uso.

En noviembre de 2013 se agregaron más conocimientos al entorno. Cuenta ahora con más de 100 recursos, entre conceptos y plantillas. Se les avisó a los alumnos de esta incorporación.

Una vez concluido el sondeo se inhabilitaron las claves, pero algunos alumnos solicitaron poder seguir ingresando al entorno para complementar las opciones y mantener el registro de su bitácora.

7.4.9 Evaluación final

A principios de diciembre de 2013 se realizó la prueba final a todos los alumnos. Inicialmente estaba planificada para semanas anteriores, pero las funciones finales que se le incorporaron y los conocimientos que quedaron pendientes y que agregaron posteriormente impidieron realizarla antes.

7.4.10 Resultados

Para la evaluación de cada prueba, se siguió el criterio indicado previamente. Se elaboró una planilla donde se colocó cada evaluación. Cada nota fue asignada en forma independiente por dos docentes y posteriormente se compararon los valores. En los casos de diferencia de puntuación (3 casos) se analizó y unificó el valor.

Finalmente se hizo un comparativo de los resultados entre los dos docentes y de forma esperada, cada uno de los grupos arrojó resultados muy diferentes. Tanto el grupo al que se le capacitó con el prototipo, aunque no lo usaron, como al que utilizó durante las sesiones el apoyo del prototipo, mostraron un gran interés en el uso de la tecnología y concretamente del prototipo para mejorar sus rendimiento y comprensión de conceptos asociados a la asignatura. En el caso de los estudiantes que no fueron confrontados con el prototipo los resultados fueron semejantes a los que se habían analizado de semestres previos y fue de este grupo de dónde se registraron los 2 casos de reprobación o abandono de la asignatura.

Grupo	Tratamiento	Aprobados	Reprobados	Desertaron
V1A	Tratamiento 1 (mantener el curso actual)	11	2*	1*
M1A	Tratamiento 2 (recibir capacitación con DECANO)	6	0	0
M1B	Tratamiento 3 (recibir capacitación y utilizar el EV)	6	0	0

Tabla 7.5 Resultados de la prueba

7.5 Tipología de la Evaluación heurística

La evaluación de expertos consistió en utilizar un conjunto de especialistas en la evaluación de usabilidad para comprobar de acuerdo a su experiencia y juicio si el sitio en evaluación cumple o no las reglas establecidas por el método utilizado (véase 7.4.1). En esta metodología se han seleccionado cuatro tipos de métodos de evaluación de inspección: inspección de heurísticas, de consistencia, de estándares y de guías que pueden ser utilizados y combinados en diferentes etapas de desarrollo.

Al realizar esta prueba se planteron los siguientes objetivos:

1. Examinar a través de una metodología de análisis heurístico el grado de usabilidad del prototipo desde el punto de vista de la navegación y la claridad arquitectónica.
2. Analizar el grado de eficiencia, eficacia y satisfacción que presentan el prototipo a través de una metodología de análisis basada en pruebas con usuarios (tests, entrevistas y cuestionarios). Para el análisis se han seleccionado estos tres aspectos, ya que son los tres aspectos que recoge la definición de usabilidad de la ISO (1): Diseño de la interfaz, Diseo educativo y Diseño de contenido.

Diseño de la interfaz

• *Asegura visibilidad del estado del prototipo (I-1)*

Regla referida a sí el prototipo proporciona retroalimentación apropiada en tiempo

razonable, es decir cuando y donde sea necesario. Esta regla se verá satisfecha si el usuario puede saber donde está y a donde puede continuar.

• *Logra correspondencia entre el prototipo y el mundo real (I-2)*

Esta regla implica el conocimiento de la audiencia a fin de utilizar un lenguaje que le sea familiar. La información debe aparecer en un orden lógico y natural. La organización del contenido y navegación debe tener sentido para la audiencia.

• *Permite al usuario control del estado y libertad de navegación (I-3)*

Esta regla esta relacionado con la visibilidad del estado del prototipo, en la medida en que proporcione al usuario la información y opciones que le aseguren mantener el control de la navegación y de como orientarse y dónde se encuentra las salidas.

• *Tiene un diseño consistente y basado en estándares (I-4)*

Las expresiones utilizadas en el contenido y otros elementos de la interfaz deben ser consistentes para evitar confundir al usuario.

• *Proporciona prevención de errores (I-5)*

Debido a que la introducción de información en la Web es una fuente común de errores es importante que el prototipo proporcione una guía para reducir el riesgo de errores.

• *Facilita la identificación de elementos en lugar de tener que recordarlos (I-6)*

Esta regla esta asociada a la tarea desde el punto de vista del usuario. Los objetos, acciones y opciones deben ser visibles o fáciles de ubicar. Si todo lo que el usuario necesita para completar una tarea satisfactoriamente no se encuentra en el lugar donde está o tiene que confiar en la memoria, entonces la regla no se cumple.

• *Soporta flexibilidad y eficiencia de uso (I-7)*

Esta regla verifica si el prototipo implementa elementos que aceleren la interacción de usuarios expertos, de manera que pueda atender tanto a usuarios expertos como inexpertos, permitiéndoles adaptarse a las acciones más frecuentes.

• *Ayuda al usuario a reconocer, diagnosticar y recuperarse de sus errores*

Mediante está guía se verifica que los mensajes de error sean expresados en un lenguaje normal (no-código), indicando claramente el problema y recomendando una solución.

• *Proporciona ayuda y documentación (I-8)*

En la medida en que un prototipo se vuelve complejo puede necesitar material referencial, instrucciones o ayuda, por tanto estos deben ser claros, concisos y diseñados para responder a preguntas específicas en un contexto específico y que sean fácilmente accesibles. Aquí adicionalmente se debe cuidar el acompañamiento y ayuda que ofrece el tutor.

Diseño educativo

• *El contexto es significativo al dominio y al aprendiz (E-1)*

Las actividades incorporadas en el prototipo deben estar basadas en la práctica para

lograr
interesar y comprometer al usuario.

• *El contenido es claro y permite la navegación y profundización (E-2)*

Los contenidos presentados deben ser claros y soportar preferencias del usuario respecto al uso de diferentes vías de acceso al mismo. Además debe permitir al usuario buscar información relevante mientras esta comprometido con su actividad.

• *Soporta las actividades educativas (E-2)*

El prototipo proporciona al usuario soporte a sus actividades para permitir trabajar con la competencia existente permitiéndole guardar y recuperar pedazos de conocimiento

• *Fomenta el entendimiento del usuario (E-3)*

Los contenidos, actividades y auto-evaluaciones deben ser presentados y organizados de manera que despierte en el usuario el entendimiento y lo articule de manera conceptual como la base de la retroalimentación.

• *Fomenta la evaluación formativa (E-4)*

Verifica que el prototipo proporcione al usuario retroalimentación constructiva durante su esfuerzo educativo.

• *El desempeño es referenciado de manera crítica (E-5)*

Esto quiere decir que el prototipo soporta la evaluación del usuario proporcionando resultados claros y medibles basados en su competencia.

• *Soporta la transferencia y adquisición de habilidades de autoaprendizaje (E-6)*

El prototipo ofrece soporte para la transferencia de habilidades que permiten al usuario ser capaz de realizar auto-aprendizaje y auto-evaluación más allá del ambiente de aprendizaje ofrecido.

• *Ofrece soporte para aprendizaje colaborativo (E-7)*

El prototipo proporciona oportunidades y soporte para la interacción y discusión con otros usuarios e involucrados facilitando de esta manera el aprendizaje u otras actividades colaborativas

Diseño de contenido

• *El contenido considera la inmersión del usuario (C-1)*

El uso de imágenes, documentos y otros materiales relacionados en el sitio crea en el usuario un sentido de inmersión dentro de la realidad simulada.

• *El contenido tiene relevancia a la práctica profesional (C-2)*

Los escenarios del problema y tareas incluidas en el contenido son realistas y relevantes a la práctica profesional del profesor.

• *Los problemas representan respuestas a problemas profesionales (C-3)*

Las soluciones mostradas representan un rango realista de las respuestas de los profesores a los problemas planteados y sirve como reto a los aprendices para considerar

enfoques alternativos.

• *La referencia a los materiales utilizados es relevante al problema y nivel del usuario (C-4)*
La referencia a los materiales incluidos es relevante a los escenarios del problema y son en un nivel apropiado a los usuarios

• *Los materiales utilizados están comprometidos (C-5)*
El estilo de presentación y el contenido presentado por el prototipo anima a los usuarios a continuar trabajando a través de escenarios.

• *Utilización y presentación de recursos (C-6)*
El prototipo ofrece recursos útiles para el desarrollo profesional de profesores y la actividad educativa del usuario presentándolos en una manera interesante y accesible.

• *Efectividad global de los materiales (C-7)*
Los materiales de soporte deben ser eficientes para lograr incrementar la confianza y la capacidad para integrar la tecnología informática dentro de la enseñanza-aprendizaje

7.6 Evaluación heurística de la interfaz o evaluación de expertos.

Las evaluaciones heurísticas son útiles para detectar errores de diseño ampliamente conocidos y reiterativos. (Hassan & Martín, 2003, p. 1). El análisis heurístico a la interfaz del prototipo DECANO fue realizada por tres expertos en el área, dos de ellos Especialistas en Diseño de Interfaz y un Docente de la Facultad de Informática de la UAQ y por dos expertos en diseño instruccional, tomando como referencias las sugerencias de Nielse (2002). La prueba se basó en la Guía de Evaluación Heurística de Sitios Web de Hassan y Martín (2003) y la definición de usabilidad de la ISO (1) a partir de los 3 aspectos referidos en el apartado anterior: Diseño de la interfaz, Diseo educativo y Diseño de contenido.

Las categorías que se evaluaron se consideraron cuantitavamente con valores distintos, según las circunstancias que las caracterizan.

1. Aspectos generales: objetivos, look y feel, coherencia y nivel de actualización de contenidos.
2. Información: identidad del sitio e información proporcionada sobre el proveedor y la autoría de contenidos.
3. Significación y familiaridad de los contenidos.
4. Estructura y navegación
5. Diseño visual, distribución y aspecto de los elementos de navegación e información en la interfaz gráfica.
6. Grado de adecuación de los contenidos al entorno y nivel de adecuación del tutor al

estudiante

7. Accesibilidad: cumplimiento de directrices de accesibilidad.

8. Control y realimentación: libertad del usuario en el uso y la navegación.

Cada evaluador, realizó la prueba de forma independiente asignando un valor (del 1 al 5) a cada una de las preguntas de las que se conforma cada categoría, en la siguiente tabla se detallan los valores asignados para cada cifra.

Valor	Observaciones
1	Se da la mínima expresión del heurístico
2	Se da una expresión baja del heurístico
3	Se da una expresión media del heurístico
4	Se da una expresión alta del heurístico
5	Se da la máxima expresión del heurístico

Tabla 7.6: descripción de los valores para la evaluación.

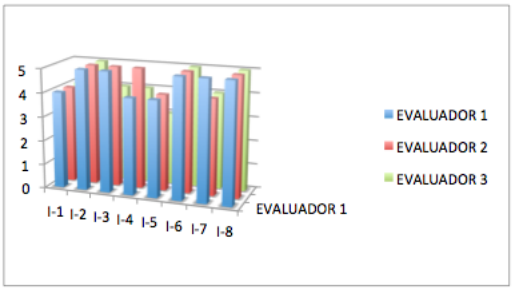
Tras el análisis de cada uno de los parámetros, se graficaron los resultados en una tabla que reuniera el puntaje total de cada categoría para cada evaluador, obteniendo así una visión amplia de cuál era el máximo valor que la categoría podía obtener y cuánto fue lo que se obtuvo.

Como se puede observar en las gráficas de resultados, la mayoría de los criterios o indicadores resultaron con una evaluación alta, esto evidencia que el modelo aplicado al diseño de la interfaz cumple con los indicadores de calidad de presentación de la información, así como con los atributos necesarios, como son la accesibilidad y presentación, a lo que se suma en esta prueba también el diseo del contenido y el diseño educativo, que complementan los objetivos propuestos.

Las puntuaciones obtenidas se graficaron teniendo los siguientes resultados:

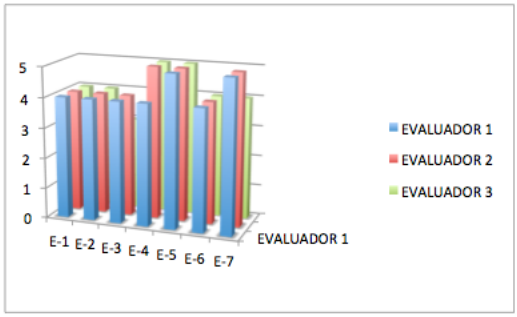
DISEÑO DE LA INTERFAZ

INDICADORES	EVALUADOR 1	EVALUADOR 2	EVALUADOR 3
I-1	4	4	4
I-2	5	5	5
I-3	5	5	4
I-4	4	5	4
I-5	4	4	3
I-6	5	5	5
I-7	5	4	4
I-8	5	5	5



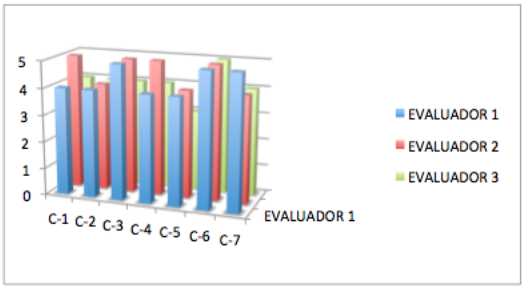
DISEÑO EDUCATIVO

INDICADORES	EVALUADOR 1	EVALUADOR 2	EVALUADOR 3
E-1	4	4	4
E-2	4	4	4
E-3	4	4	3
E-4	4	5	5
E-5	5	5	5
E-6	4	4	4
E-7	5	5	4



DISEÑO DE CONTENIDO

INDICADORES	EVALUADOR 1	EVALUADOR 2	EVALUADOR 3
C-1	4	5	4
C-2	4	4	4
C-3	5	5	4
C-4	4	5	4
C-5	4	4	3
C-6	5	5	5
C-7	5	4	4



Gráfica 7.1: Resultados de las pruebas heurísticas.

Diagnóstico general de la interfaz.

En un diagnóstico general que se desprende de la evaluación de los expertos en diseño de interfaz, se concluyó que el diseño del prototipo está orientado al usuario, considerando la navegación como la identificación de objetivos del prototipo, aportando una estructura general fácil de navegar, reconocible en cuanto a los elementos simbólicos y sintácticos del diseño y sobretodo fácilmente acondicionable al tipo de alumno que se pretende atender.

Conclusiones de la Evaluación

Partiendo del objetivo principal del trabajo de investigación podemos decir que lo pretendido inicialmente se ha logrado de una manera efectiva: hemos realizado una propuesta de Arquitectura modificable, demostrando la validez de la propuesta con la construcción de un prototipo que implementa un parte del sistema concreto basado en dicha Arquitectura. El prototipo realizado es útil y valioso, ya que demuestra que se puede llevar a la práctica la arquitectura y de esta forma permitir la interoperabilidad entre distintos sistemas de formación y repositorios distribuidos.

La arquitectura propuesta puede resolver los problemas de reutilización de los objetos de aprendizaje mediante su publicación y localización universal. Esto permite distribuir contenidos educativos entre distintas plataformas de e-learning y hacer interoperables los repositorios de las mismas. Además de todo ello, a lo largo del tiempo en el que se ha realizado la investigación, se ha obtenido una serie de conclusiones como consecuencia directa de la misma y de los problemas que se han tenido que solucionar en el desarrollo del prototipo.

Como pudimos ver, este capítulo aborda la experiencia de usuario y la forma en que fue elaborada la experimentación. En primera instancia se realizó un prototipo con una célula representativa del entorno de enseñanza. Posteriormente se seleccionaron las variables a observar y medir, como comprensión del problema, y la transferencia del conocimiento. Se eligieron y justificaron escalas ordinales así como el uso de estadística no paramétrica y finalmente, se presentó en forma detallada la manera en que se diseñó y desarrolló el experimento, incluyendo los instrumentos utilizados.

Al realizar esta experimentación se tuvo en cuenta que los objetivos de investigación se definieran claramente, que el diseño experimental estuviera relacionado con la hipótesis, que se validaran los instrumentos de evaluación, y que los datos se recolectaran de manera objetiva y se representaran también adecuadamente.

A sugerencia del equipo, se incluyó la bitácora de la experimentación y el detalle completo de los resultados obtenidos. A partir de estos datos, se resumen y se realizan varios posibles análisis, agrupándolos por diferentes conceptos. Por cada aspecto analizado, se incluyen los datos correspondientes y la interpretación de los mismos. Finalmente, se presenta un análisis global y las conclusiones de la experimentación. En pocas palabras, se detecta que el entorno resulta beneficioso en los aspectos de formas de resolución de problemas y transferencia del conocimiento.

Los resultados de la evaluación han arrojado un balance bastante favorable en cuanto a los beneficios proporcionados por la realización de un diseño sistemático y pensado, ya que gran parte de los factores que han hecho acto de presencia durante la evaluación ya se habían tenido

en cuenta a la hora de realizar el diseño. Por una lado la prueba de usuario permitió probar que los alumnos se encuentran cómodos cuando aprenden apoyados de la tecnología. Por otro lado el prototipo se muestra como una opción viable, novedosa, flexible y adaptable a las distintas necesidades tanto de los alumnos como de los dominios de aprendizaje.

Por lo tanto, resulta muy conveniente resaltar que se debe dedicar más recursos y tiempo a la realización del prototipo incluso que a la evaluación, que aunque debe realizarse, podría llevarse a cabo de una forma más ligera y ser igualmente funcional.

La solución propuesta es original en tanto brinda un modelo de entornos diferente a todos los analizados, es aplicable pues se ha demostrado su viabilidad a través del prototipo DECANO y es eficiente (en el sentido que tiene la capacidad para lograr un efecto determinado, pues, a partir del análisis de los datos obtenidos en la experimentación, el uso del entorno permite que el alumno amplíe o mejore sus formas de resolución de problemas así como sus capacidades para realizar la transferencia de conocimiento). Se cumple con los pilares de la gestión de los conocimientos, pues se permite explorarlo, encontrarle el valor y manejarlo activamente (Wiig, K., 1999).

Luego de diseñado el modelo y aplicado en una versión preliminar, puedo decir que me encuentro satisfecha del resultado. Primeramente, estoy altamente sorprendida de la tutoría inteligente, que dota al entorno de agentes de forma que colaboren en la selección y sugerencia de conocimientos. Adicionalmente, me siento altamente complacida por el resultado del modelado del tutor, pues considerando mi total inexperiencia, el prototipo es digno de valorarse y elogiarse, porque cumple con su cometido. También las posibilidades que le otorgó la web al entorno nos situán en la línea ideal del trabajo colaborativo.

Y finalmente esta experienca me permitió obtener una visión más amplia del trabajo interdisciplinario, que ahora está consolidando como equipo de trabajo formal a todos los que participamos y que al inicio simplemente nos motivaba lo que quizá podía suceder y que hoy nos mantiene mucho más entusiasmados, porque el factor sorpresa fue determinante. Este equipo integrado de forma accidentada y fortuita, hoy tiene en la mira, desarrollar la versión completa del mismo, con todos los contenidos y elementos sugeridos al inicio y ver si el DECANO todavía nos puede dar más dolores de cabeza, pero como buen hijo, también muchas satisfacciones.

CONCLUSIONES

CONCLUSIONES

Como ha quedado reflejado en el capítulo 2, hemos llevado a cabo un análisis exhaustivo de la situación actual relacionada con los estándares y recomendaciones para el diseño y desarrollo de sistemas para la formación con tecnología. Sobre la base de estas experiencias, se mantiene el criterio de que la utilización de estándares, ha sido siempre beneficiosa para el desarrollo y consolidación de tecnologías emergentes, puesto que permite un mayo control y establece desde un inicio líneas muy claras de acción. Además de ayudar a la interoperabilidad entre sistemas, establecen marcos comunes de actuación que permiten aprovechar los avances de los diferentes equipos de investigación, así como la reutilización de contenidos en otros sistemas de parecidas características.

En la medida en que seamos capaces de difundir la utilización de los estándares ya existentes y de desarrollar nuevas y útiles recomendaciones de uso y utilización de los mismos, se crearán mejores condiciones para la implantación de tecnologías que ayuden a la mejora de la enseñaza y el aprendizaje.

Como se ha comentado a lo largo del trabajo, para que un objeto de aprendizaje sea reutilizable necesitamos que su construcción se haya realizado conforme a estándares o especificaciones como los establecidos por IMS, ADL (SCORM) o IEEE (LOM). De esta forma aseguramos que, a través de su empaquetamiento y descripción, pueda ser integrado en cualquier plataforma de e-learning compatible con estos estándares. Sin embargo, al existir varios estándares sería recomendable el desarrollo de uno único que simplifique la interoperabilidad entre distintos sistemas de teleformación o repositorios. Como actualmente esto no ocurre y debido a la variedad de estándares y especificaciones existentes, se hace necesario tener utilidades y herramientas que permitan la adaptación tanto de los contenidos docentes como de la metainformación que se gestiona en el proceso de aprendizaje.

Esta es una las características que hemos integrado en la arquitectura y que permite desarrollar sistemas capaces de realizar las conversiones necesarias de formato de metainformación de forma transparente al usuario.

Sería deseable que las herramientas de preparación y creación de contenidos, cuyo número es muy elevado, incorporaran utilidades o herramientas que, centrándose en aquellos estándares que se perfilen como realmente utilizados, es decir, que lo sean “de facto”, permitan, incluso también de una manera guiada, la adaptación de los contenidos a dichos estándares. Y siguiendo con esta idea, permitir que los repositorios trabajen con diversos estándares “de facto” para conseguir la integración de contenidos educativos heterogéneos.

Siguiendo con la idea de la reutilización de los objetos de aprendizaje, además de estar construido siguiendo los estándares, debe ser localizable universalmente. De poco sirve un objeto de aprendizaje con un alto nivel de calidad si sólo es accesible por unos cuantos usuarios de una determinada plataforma o repositorio. Las instituciones educativas requieren mecanismos de interoperabilidad, ya que sería muy costoso quedar con contenido aislado en un mundo cada vez más interconectado y que clama por la colaboración institucional como mecanismo para garantizar una educación de calidad.

Si queremos que los objetos de aprendizaje sean reutilizables por un número potencialmente alto de usuarios, debemos integrarlos en un prototipo capaz de localizarlos y exponerlos a estos usuarios y prototipos. De esta forma también los proveedores de contenido podrán fácilmente reutilizarlos y distribuirlos entre diferentes sistemas. Pero para que esto ocurra no podemos obligar a que los actuales sistemas de teleformación o los repositorios existentes modifiquen su estructura y funcionamiento. Debemos proporcionarles los mecanismos necesarios para que la incorporación de esta funcionalidad sea lo menos traumática posible. De ahí la utilización de los servicios Web y las arquitecturas orientadas a servicios, tecnologías que han demostrado su principal valía en la integración de aplicaciones, de forma que añadimos cierta funcionalidad sin modificar lo ya desarrollado.

En cuanto a la definición de arquitecturas de sistemas de e-learning se ha podido comprobar que la implantación de arquitecturas modulares y flexibles (como la presentada en la tesis) ya son una realidad inmediata. De esta forma conseguiremos que en un futuro los sistemas desarrollados siguiendo este tipo de arquitectura, sean capaces de integrarse con otros sistemas de una manera muy sencilla y por lo tanto realizar sistemas interoperables. Esto se conseguirá con la integración de servicios de integración capaces de comunicarse o compartir información con otros sistemas.

Como se explicó en las conclusiones de capítulos previos, se quería desarrollar una arquitectura que tuviera una serie de características, a continuación presentamos estas características y las razones por las que se han conseguido integrar.

Abierta. La arquitectura planteada permite la creación de sistemas e-learning interoperables y conectables entre sí de forma sencilla; es decir, que sistemas y herramientas comerciales de fabricantes distintos puedan ensamblarse en un único prototipo global. Esto se ha conseguido utilizando servicios Web.

Modificable. La arquitectura está definida de tal forma que permite su crecimiento. Este crecimiento se puede ver desde dos perspectivas. Por un lado tenemos el crecimiento de datos, representados por los nuevos objetos de aprendizaje que se incorporarían al prototipo al incluir un nuevo repositorio. Y por otro lado tenemos el crecimiento en funcionalidad que se daría al incluir nuevos servicios a la arquitectura, tarea relativamente sencilla dada la naturaleza del modelo

SOA aplicado en su desarrollo.

Integrada. Un prototipo desarrollado siguiendo la arquitectura propuesta es capaz de integrarse con una gran cantidad de sistemas de e-learning existentes hoy en día (y en un futuro), consiguiendo la interoperabilidad entre todos ellos. Para conseguirlo, tan solo se tiene que desarrollar un servicio Web que de acceso a sus contenidos y registrarlo en nuestro prototipo.

Flexible y adaptable. La arquitectura propuesta tiene la capacidad de implementar nuevas soluciones sin tener que efectuar grandes cambios en la misma. Como se comentó anteriormente, esto se consigue al haber realizado una arquitectura en la que la inclusión de nuevos servicios no resulta una tarea imposible o complicada, sino todo lo contrario. A modo de resumen podemos presentar como características más destacadas de la arquitectura y el prototipo desarrollado, las siguientes: Presenta una arquitectura abierta y utiliza protocolos y formatos estándar para permitir la interoperabilidad e integración de plataformas de aprendizaje y repositorios.

La posibilidad de publicar y descubrir cualquier objeto de aprendizaje independientemente de su localización y sus metadatos.

Presenta una forma uniforme y bien definida de acceso a los objetos de aprendizaje.

Hace que los objetos de aprendizaje sean reutilizables.

Por último, destacar que todas estas conclusiones, conducen directamente al resultado de que la arquitectura propuesta es necesaria y deseable en el desarrollo actual de sistemas de e-learning, y plantea soluciones interesantes y viables para la interoperabilidad de sistemas heterogéneos y la reutilización de contenidos docentes. Además, el desarrollo de la arquitectura se ha basado en estándares y especificaciones actuales que la convierten en una opción sólida y que ayuda a la consolidación de normas. En la evolución de los sistemas de e-learning se está tendiendo hacia la idea de interoperabilidad e interconexión planteada en esta tesis, por lo tanto creemos que nuestra arquitectura puede suponer un gran avance en este sentido.

Con respecto a la interfaz, se han propuesto mecanismos para la generación de la interfaz no sólo en función del usuario, sino para que la capacidad de adaptación se mejore, la experiencia de usuario la haga más sencilla y sobretodo, a partir de estándares y métricas probadas que al evaluarse cumplen con lo que se persiguió de origen.

Desarrollar tanto la interfaz del prototipo principal, las interfaces de las herramientas de autor, la interfaz del entorno web y la interfaz y modelado del tutor han servido como referencia comprensiva, visual y estratégica para la creación y desarrollo de un proyecto, proporcionando resultados que puedan orientar a las investigaciones, apreciando multiplicidad de variables como el formato, contenido, distribución, entre otros, que pueden contribuir al estudio en desarrollo y a

otros desarrollos.

Todo esto ha permitido presentar aportaciones al diseño a partir de consideraciones en el desarrollo de interfases para sistemas como el desarrollado a lo largo de este proyecto y que pueden ser útiles como referencias o métodos de trabajo.

De nuevo se hace patente al finalizar este documento, la recurrente recomendación de integrar equipos de trabajo interdisciplinarios, para que los proyectos educativos cumplan con la calidad que hoy se está demandando. Considero que la experienca plasmada a lo largo de este documento, no hace sino reforzar esta condición.

FUTURAS LÍNEAS DE INVESTIGACIÓN

Actualmente y como consecuencia de este proyecto, el equipo de trabajo está buscando inicialmente consolidarse como cuerpo de investigación y se han propuesto abrir varias líneas de investigación en las que se está ya trabajando. La primera y más avanzada es la integración de dispositivos móviles en la Arquitectura propuesta en esta tesis. La segunda es la incorporación de la Web semántica y las ontologías como forma de mejorar las búsquedas. Una tercera propuesta es la integración de agentes inteligentes para la búsqueda personalizada y adaptada a las necesidades de los usuarios y finalmente la última gira en torno al diseño responsivo y adaptable y la interfaz de usuario, lo que permitirá mejor la sinergia entre disciplinas potenciando tanto la interdisciplinariedad como la transdisciplinariedad.

De forma muy particular, he decidido involucrarme con el diseño completo del prototipo y la documentación del modelado de interfaz, pues he podido constatar que se ha escrito muy poco al respecto y considero que al ser un campo desatendido por los diseñadores, bien vale la pena aportar un poquito más.

Quizá para cuando se lean estas líneas ya existan aún más posibilidades, pero quizá también no podamos abarcarlas todas, lo que si es interesante, son las vertientes que el diseño de interfaz está potenciando y que es dónde talvez pueda puentearse este documento y sus intenciones, con otras aún más ambiciosas, como la realidad aumentada o la simulación y quizá para allá se encaminen mis pasos.

Creo que lo único que me queda por escribir es que afortunadamente la visión ha crecido y la motivación puede llevarnos muy lejos, el esfuerzo ha sido mucho, pero me siento gratamente sorprendida porque en el camino he conocido y leído a muchos profesionales, a los que admiro

y les agradezco sus visiones que han quedado plasmadas en este documento y que nunca dejaré de reconcer y aplaudir. Jamás pensé estar involucrada en un proyecto como este, pero me complace haber sobrevivido a la experiencia.

ANEXOS

ANEXO: Metodologías de Desarrollo con Agentes

Existen ya un buen número de publicaciones que abordan temas como qué son los agentes y los sistemas multiagente, cómo desarrollarlos y cómo implementarlos (Weiss, 1999; Mas, 2005; Bellifemine et al., 2007). También, como consecuencia, han surgido diversas metodologías de desarrollo orientadas a agentes que en mayor o menor medida ayudan en la construcción de estos sistemas. Una de las principales diferencias entre ellas se sitúa en los distintos puntos de vista utilizados para diseñar el sistema.

Algunas de ellas, como Tropos (Giorgini et al., 2004; Bresciani et al., 2004), se basan en la utilización de UML (Lenguaje Unificado de Modelado) para modelar los sistemas basados en agentes. Esta metodología define cuatro fases para realizar el diseño del sistema: análisis de requisitos preliminar, análisis de requisitos final, diseño arquitectónico y diseño detallado. En la tercera fase, de diseño arquitectónico, proponen la utilización de estilos arquitectónicos para definir la organización y, consecuentemente, el diseño de la arquitectura.

Una de las primeras metodologías orientadas a agentes que se desarrollaron es AAII/BDI (Australian Artificial Intelligence Institute/Beliefs Desires Intentions) (Kinny et al., 1996), que considera dos puntos de vista, uno externo y otro interno, para proponer el desarrollo de sistemas basados en agentes. Con el punto de vista externo se identifican los agentes existentes o modelo de agentes y las relaciones entre ellos o modelo de interacciones. El punto de vista interno está basado en el modelo BDI para desarrollar agentes (Haddadi y Sundermeyer, 1996).

Existen más metodologías que consideran distintos puntos de vista para desarrollar el sistema, como por ejemplo la conocida como Ingeniería de las Vocales (Demazeau et al., 2001), que considera cinco puntos de vista: Agente, Entorno, Interacciones, Organización y Usuario (esta última de más reciente incorporación). Para desarrollar cada uno de estos aspectos se pueden utilizar distintas técnicas, y el objetivo es desarrollar librerías que permitan solucionar cada uno de dichos aspectos. Además, dependiendo del sistema que se vaya a desarrollar, se propone considerar las vocales en distinto orden, dando más importancia a las vocales correspondientes a los aspectos primordiales del sistema. Si lo más importante es la interacción entre agentes, se comenzará modelando la organización. Si, por el contrario, se comienza modelando a los agentes, la organización emergerá

como resultado de la interacción entre los agentes individuales.

También en la metodología MAS-CommonKADS (Iglesias et al., 1998), derivada de la metodología CommonKADS[1] para desarrollo de sistemas de gestión del conocimiento (Schreiber et al., 1999), se considera la utilización de distintos puntos de vista. Esta es la primera metodología que incorpora la notación de la orientación a objetos, en concreto de la metodología OMT, para estructurar el sistema.

MESSAGE es una metodología que propone la integración de características que aparecen en otras metodologías y resultados de investigación (Caire et al., 2001), y considera cinco puntos de vista similares a los de la ingeniería de las vocales. La novedad radica en el uso de metamodelos para describir los aspectos relacionados con los puntos de vista.

Una extensión de los metamodelos de esta metodología ha dado lugar a INGENIAS (Pavón y Gómez-Sanz, 2003), que incorpora además una serie de herramientas construidas sobre esos metamodelos (Gómez-Sanz y Pavón, 2006). INGENIAS considera también cinco puntos de vista para desarrollar el sistema, pero sustituye la U de las vocales (el usuario) por un punto de vista de tareas y objetivos.

Adicionalmente también se creó Prometheus, desarrollada en colaboración con la compañía Agent Oriented Software (*Jack Intelligent Agents*™). Es una metodología que describe el desarrollo de agentes BDI, en contraposición a las metodologías que tratan a los agentes como cajas negras. Incluye todas las fases de diseño e implementación. El modelo se basa en una estructuración jerárquica para facilitar la escalabilidad. Consta de 3 etapas:

Especificaciones del sistema: Consiste en la identificación de Funcionalidades básicas, Entradas (percepts), Salidas (acciones) y Fuentes de datos compartidas.

Diseño arquitectónico: Define Agentes, Incidentes e Interacciones, las estructuras de datos compartidos. Y adicionalmente, describe qué agentes son necesarios y

1 La metodología CommonKADS (Estándar para el desarrollo de sistemas en la gestión del conocimiento), es una metodología europea para el desarrollo de Sistemas Basados en Conocimientos (SBC) que ofrece teorías, métodos y técnicas científicas para representar el conocimiento y es creada para modelar los procesos mentales y se aproxima a los contenidos de conocimiento de las personas. CommonKADS también proporciona los métodos para obtener una comprensión de las estructuras mentales y de los procesos de razonamiento usados por los trabajadores de conocimiento para reproducirlos en sistemas informáticos.

cómo interaccionan los agentes.

Diseño detallado: Define capacidades, planes, estructuras de creencias, eventos. Describe cómo realiza cada tarea cada agente internamente dentro del sistema global. Se centra en el desarrollo de la estructura interna de los agentes (siguiendo el modelo BDI) y cómo alcanzar las tareas dentro del sistema.

Las funcionalidades de los agentes se pueden agrupar en capacidades que permiten hacer modular y reutilizable el diseño de los agentes.

Hay otras metodologías, como GAIA (Zambonelli et al., 2003), que en lugar de considerar el sistema desde distintos puntos de vista se centran en los roles que desempeñan los agentes y sus interacciones. Los roles están definidos por varias propiedades, y el análisis en esta metodología consiste en identificar posibles roles junto con sus propiedades. A partir de ellos, la fase de diseño produce un modelo de agentes, un modelo de servicios y un modelo de conocidos. La idea básica parte de la construcción de sistemas basados en agentes como un proceso de diseño organizativo. A esta metodología se le achaca que los modelos producidos tienen un nivel de detalle demasiado alto, y que aún se debe profundizar en el diseño para poder implementar el sistema. A pesar de ello, es una de las metodologías más citadas y utilizadas.

En la misma línea que GAIA se desarrolla MaSE (Multiagent Systems Engineering) (DeLoach et al., 2001), que a diferencia de la anterior, sí aborda todo el ciclo de desarrollo hasta la construcción del sistema. Sin embargo, MaSE está más cercana a la orientación a objetos, y considera a los agentes como una especialización de los objetos.

Varias fases de la metodología se corresponden con fases de metodologías orientadas a objetos, como un análisis de casos de uso y la consiguiente elaboración de diagramas de secuencia. Un aspecto al que MaSE presta atención son las conversaciones entre agentes, que se definen como protocolos de coordinación, para lo cual es necesario haber definido antes los roles que pueden adoptar los agentes. Al igual que otras metodologías, MaSE ha dado lugar a la elaboración de una herramienta, agentTool, que ha posibilitado una expansión bastante rápida de esta metodología.

La metodología ZEUS propone el desarrollo del sistema en cuatro fases (Collis et al.,

1998): el análisis del dominio, el diseño de los agentes (que se divide en tres subfases), la realización de los agentes y el soporte en tiempo de ejecución. ZEUS cuenta con bastante difusión debido, en parte, a que proporciona una herramienta que soporta las fases finales del desarrollo del sistema. Hasta la aparición de MaSE, Zeus constituía la herramienta de desarrollo de referencia, pero por su amplitud y resultar tan completa, MaSE como metodología ha desplazado a ZEUS a un segundo lugar.

ANEXO. Modelos de estudio para el desarrollo de una interfaz gráfica de usuario

El diseño de interfaces de usuario (HCI Human Computer Interfece o IU Interfaz de Usuario), es una tarea que ha adquirido relevancia en el desarrollo de un sistema. La calidad de la interfaz de usuario puede ser uno de los motivos que conduzca a un sistema al éxito o al fracaso. Los principios que se presentan son de utilidad para creación de interfaces funcionales y de fácil operación. A pesar de no ser capaces de resolver todos los aspectos propios del contexto con el que se esté trabajando, pueden ser combinados con el prototipado y la aplicación de heurísticas de evaluación para facilitar el proceso de diseño.

La necesidad de desplazarse entre los diversos elementos y escenarios del sistema y dentro de ellos significa elegir un modelo de navegación y un aspecto visual, es decir, una interfaz y, muy posiblemente, una metáfora. En el contexto de los entornos de enseñanza virtual ambos términos, interfaz y metáfora, se utilizan a veces indistintamente, a pesar de expresar distinto niveles de la interaccion humano-computadora(IPO).
Falgueras y otros(2002), han propuesto una clasificación en cuatro modelos de las metodologías más utilizadas en el estudio de dicha interacción: empírico, cognitivo, predictivo y antropomórfico. De todos ellos, el modelo cognitivo, el más relevante para los EVs.

1. Modelos de estudio de la interfaz gráfica de usuario

Lewis y Rieman (1993) explican que la función de la interfaz es la de mediar entre el hombre y la máquina. Desde esta perspectiva una interfaz permite utilizar una máquina para el propósito que fue construida, los autores explican que la interfaz es lo que facilita la comunicación y la interacción, entre dos sistemas de diferente naturaleza; como son el ser humano y la computadora. La interfaz es también la presentación que en pantalla ofrece el sistema al usuario para que ambos puedan interactuar. En el nivel conceptual de las ciencias de la computación la interfaz se divide en hardware y en software:

Ejemplos de interfaz del tipo hardware son el teclado, el mouse, las pantallas táctiles, etc. (es decir, superficies de contacto como las interfaces táctiles, visuales, biométricas y otras conocidas como multimodales). Las interfaces del tipo software hacen referencia a todos los programas que permiten la interacción entre el sistema operativo, la máquina y las tareas que requiera el usuario (es decir, superficie de contacto a nivel del software) En los siguientes párrafos se hace la presentación de los distintos enfoques derivados de la metodología interacción humano-computadora (IHC) para analizar el diseño y desarrollo de la interfaz y poder así fundamentar una propuesta de componentes para el diseño de la interfaz de un EVEATI apoyado por un agente tutor.

Según los trabajos de Eberts (1994) y Falgueras (2000), podemos categorizar en cuatro los modelos o enfoques de estudio de interfaz de usuario: empírico, cognitivo,

predictivo y antropomórfico. Aunque hay que tener en cuenta que estas divisiones no son necesariamente nítidas ni rígidas en todos los casos, ya que como veremos, un gran número de estudios podrían quedar encuadrados dentro de más de una de ellas. Por ejemplo, muchas técnicas del modelo predictivo están basadas en teorías cognitivas, pudiéndose por tanto clasificar bajo ambos enfoques. La siguiente figura muestra el enfoque de análisis de la metodología IHC.

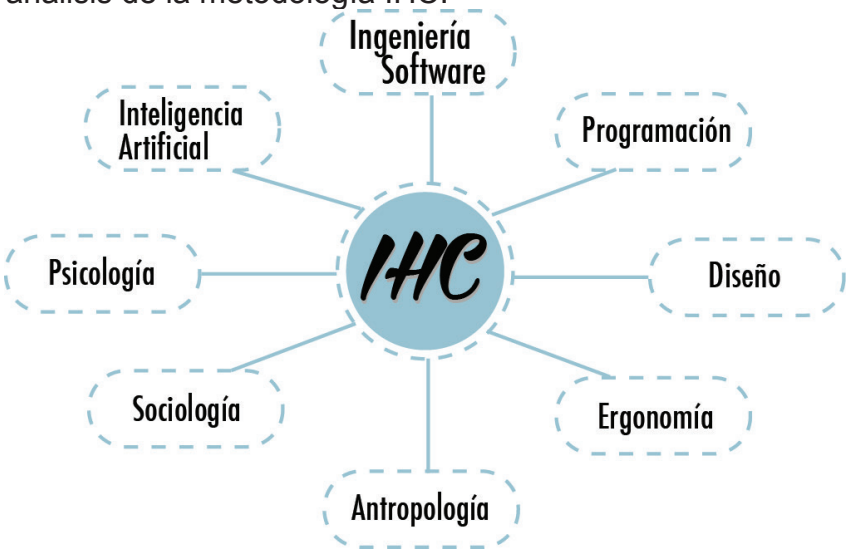


Figura 1 Diagrama de elementos de Interacción Humano-computadora

A pesar del hecho de que algunas categorías de estudio dependan muy estrechamente de un modelo formal, también existen otras que no siguen siquiera un método de formalización o bien admiten cualquiera de ellos. A continuación explicamos las características de estos modelos de acuerdo a los trabajos de Eberts (1996) y Falgueras (2000).

1.1 Modelo cognitivo

Relacionado con el estudio de la persona, las teorías cognitivas (cuyas bases se encuentran en la psicología cognitiva), establecen cómo el ser humano percibe, guarda y recupera la información de su memoria a corto plazo, cómo manipula esta información para tomar decisiones y cómo da soluciones y resuelve problemas. Además, ven al usuario como un ser adaptativo, flexible e involucrado de manera activa en la interacción con el entorno en búsqueda de soluciones. Este modelo se basa en la existencia de tres puntos de vista:
1. El modelo conceptual, o descripción del sistema desde el punto de vista del ingeniero que lo ha diseñado, por lo que se tratará de una descripción precisa y coherente.
2. El modelo mental o imagen que el usuario se hace sobre cómo es y cómo funciona el sistema, y que será el que ayude y guíe al usuario en su toma de decisiones.
3. La interfaz, que es el conjunto de estética, representaciones utilizadas, documentación, etcétera, que usa el diseñador para tratar de guiar al usuario hacia un modelo mental lo más cercano al conceptual y lo más práctico posible.

En general, desde el enfoque cognitivo, la interacción humano-computadora se concibe como la presentación de problemas a un usuario que tiene que resolverlos. Por esto muchas de las teorías que se aplican en psicología se aplican también aquí. Particularmente han resultado de gran influencia las teorías de Newell y Simon (1992), sobre la resolución de problemas, que se puede considerar como una serie interconectada de objetivos y sub-objetivos en jerarquía, lo que conduce a un modelo muy global del ser humano y relativamente sencillo de su comportamiento, y gracias a lo cual se han logrado derivar modelos formales como los de Bovair y Polson (1990), Kieras y Polson (1985), o Card y otros (1983). Son modelos cognitivos porque aportan mecanismos que permiten analizar y describir el conocimiento que el usuario posee o debe poseer del sistema. Con estos estudios se pretende entender y prever el comportamiento humano en la interacción.
El enfoque cognitivo de la interacción humano-computadora considera la capacidad del cerebro humano y la percepción sensorial, con el fin de desarrollar una interfaz de usuario que ayuda al usuario final.

a) Modelo Cognitivo/ Diseño Metafórico.
El uso de metáforas puede ser una manera eficaz de comunicar un concepto abstracto o el procedimiento para los usuarios, siempre y cuando la metáfora se utiliza con precisión. Los equipos utilizan un «escritorio» metáfora para representar los datos como archivos de documentos, carpetas y aplicaciones. Las metáforas se basan en la familiaridad del usuario con otro concepto, así como affordances[1] humanos, para ayudar a los usuarios a entender las acciones que puede realizar con sus datos basados en la forma que adopte. Por ejemplo, un usuario puede mover un archivo o una carpeta en el «cubo de basura» para borrarla.

Una de las ventajas del uso de metáforas en el diseño es que los usuarios que pueden relacionarse con la metáfora son capaces de aprender a usar un nuevo sistema muy rápidamente. Un problema potencial puede derivarse de ello, sin embargo, cuando los usuarios esperan una metáfora para estar plenamente representada en un diseño, y en realidad, sólo una parte de la metáfora se ha implementado. Por ejemplo, las computadoras Macintosh utilizan el ícono de una papelera en el escritorio, mientras que los otros equipos tienen una papelera de reciclaje. La papelera de reciclaje en realidad no «reciclar» los datos, sino que se comporta como la basura y se utiliza para eliminar permanentemente los archivos. Por otro lado, con el fin de expulsar un disco montado en una Macintosh, el usuario debe arrastrar el ícono de un CD a la papelera. Cuando esto se introdujo por primera vez, era confuso para los usuarios por temor a perder todos los datos del disco. En las versiones más recientes del sistema operativo Mac, el ícono de la papelera se convierte en un símbolo de expulsión cuando el usuario arrastra un disco montado en el bote de basura. Esto no hace que la metáfora perfecta, pero no evita la

1 Según Gibson (1979) una affordance es la relación epistémica entre el agente y su medio. Ésta surge cuando el agente detecta una información específica que le permite modular su acción. Así pues, las affordances no son más que oportunidades para la acción, una traducción al castellano, aunque no generalizada, los interpreta como “disponibilidades” o “facilitaciones”.

confusión del usuario cuando se expulsa el disco montado.

b) Modelo Cognitivo / La Atención y la Carga de Trabajo Modelos.
En el diseño de una interfaz para considerar la usabilidad, es importante tener en cuenta la atención del usuario, que puede basarse en el entorno de uso, y la carga de trabajo mental percibida, ambos elementos involucrados en la realización de una tarea. Por lo general, los usuarios pueden centrarse así en una tarea-en-un-tiempo. Por ejemplo, al diseñar un formulario web para recopilar información de un usuario, lo mejor es recoger información de contexto por separado de la demás información. El formulario puede ser dividida en «Información de contacto» y «la información de facturación», en lugar de mezclar los dos usuarios y confuso.

Por la «fragmentación» de estos datos en las distintas secciones o páginas separadas cuando hay una gran cantidad de información recopilada, la carga de trabajo que se percibe es también reducida. Si todos los datos se recogieron en un formulario único que hace el usuario desplazarse por la página para completar, el usuario puede sentirse abrumado por la cantidad de trabajo que hay que hacer para completar el formulario, y puede abandonar el sitio web. La carga de trabajo se puede medir por la cantidad de información que se comunica a cada sistema sensorial (visual, auditiva, etc.) en un momento dado.

Sobrecargar la memoria del usuario es otro problema común en. Por ejemplo, cuando hay demasiadas opciones para elegir, un usuario puede sentirse abrumado por la decisión que tienen que hacer, se sienten frustrados, y dejar sin completar su objetivo.

c)Modelo Cognitivo/Modelo de Procesamiento de Información Humana.

Esta Teoría describe el flujo de información del mundo, en la mente humana, y de nuevo en el mundo. Cuando un ser humano se fija en algo, la primera información se codifica basado en el sistema sensorial que canaliza la información (visual, auditiva, táctil, etc.) A continuación, la información se mueve en la memoria de trabajo, antes conocida como memoria a corto plazo. La memoria de trabajo puede contener una cantidad limitada de información de hasta 30 segundos aproximadamente. Repetir o ensayar la información puede aumentar esta duración. Después de la memoria de trabajo, la información puede ir a la memoria a largo plazo o simplemente ser olvidado.

Se cree que en realidad la memoria de largo plazo es un almacenamiento ilimitado, la memoria relativamente permanente. Después de que la información ha sido almacenada ahí, los seres humanos puede recuperar esa información a través de recuerdo o reconocimiento. La precisión de recordar la información se basa en las condiciones ambientales y la forma en que la información fue codificada inicialmente por los sentidos. Si un ser humano se encuentra en una experiencia sensorial similar en el momento de

la recuperación de la memoria, como lo fue durante la codificación de una experiencia previa, su recuerdo de esa experiencia será más exacta y completa.

1.2 Modelo Predictivo

Es un modelo eminentemente práctico, en el cual se da un enfoque básicamente ingenieril al proceso de diseño de la interfaz de usuario, generalmente mediante técnicas formales, y en el que se tratará de predecir la eficiencia de la interacción entre el humano y el ordenador. Es una medida de viabilidad antes del prototipo y por lo tanto, básicamente teórica basada en modelos abstractos. Además, varias de estas técnicas permiten en mayor o menor grado, tanto plantear formalmente la interfaz que se desea construir y describir su funcionamiento (función generativa), como analizarla y evaluarla con objeto de poder depurarla y mejorarla. El diseñador deberá por tanto, realizar tareas de modelado, considerar posibles diseños, modelar las tareas de estos diseños y elegir uno de ellos, el mejor diseño, basándose en predicciones cuantitativas y estimaciones precisas de tiempos para las tareas. Se basan en el análisis y por lo tanto tienen el inconveniente de requerir la interpretación
de las tareas.

1.3 Modelo Empírico

El diseño se basa en la propia experiencia del diseñador y en la recolección de principios generales de interacción y recomendaciones, propios del sistema de trabajo y en estándares oficiales (Abascal, J. y otros 2002). Los resultados tienen que verse respaldados posteriormente mediante test de usabilidad. En general, la interacción humano-computadora depende de tantos factores no formalizadlos ni previsibles, que se necesitarían comprobaciones sobre situaciones reales, con interfaces de usuario concretas ya diseñadas (en prototipos) o aplicaciones finales. El diseñador deberá recoger y analizar los resultados de estudio de campo en los que se observan comportamientos y se toman medidas durante el uso de elementos particulares de la interacción humano-computadora (Moher y Schneider 1982 y Embley 1988). La principal ventaja de este modelo es que ofrece una alternativa a la intuición en la determinación del mejor diseño. De hecho, algunos experimentos de campo han invalidado teorías altamente intuitivas como la de la ventaja de las interfaces gráficas en determinadas áreas, donde se ha encontrado experimentalmente una mayor adecuación de las interfaces textuales.

En otros casos, la intuición se puede ver corroborada por el experimento Sin embargo, presenta aspectos negativos importantes. La experimentación aislada no es adecuada en muchos casos, en otros se pueden diseñar experimentos sin una investigación previa pero si tomando como base las conductas de los usuarios, a veces los usuarios sometidos a evaluación no son los adecuados (como por ejemplo, al emplear sujetos inexpertos), lo que lleva sencillamente a hacer imposible la generalización de los resultados. En muchos casos, el modelo experimental viene a surgir de carencias de orientación teórica y no induce a crear ninguna línea nueva. La mayoría de los experimentos se planifican para resolver problemas particulares en momentos concretos del desarrollo.

La aproximación empírica a la IHC es útil para examinar y comparar la utilidad de los múltiples diseños conceptuales. Esta prueba se puede realizar durante la pre-producción de contrarrestar los conceptos de diseño y la realización de las pruebas de usabilidad de cada concepto de diseño. A menudo, los usuarios podrán apreciar los elementos específicos de cada concepto de diseño, que puede conducir al desarrollo de un diseño conceptual compuesto de prueba.

1.4 Modelo antropomórfico

El enfoque antropomórfico en la interacción humano-computadora implica el diseño de una interfaz de usuario que posee el hombre, como cualidades. Por ejemplo, una interfaz puede ser diseñada para comunicarse con los usuarios de una forma de humano a humano, como si el equipo se identificara con el usuario. Mensajes de error en la interfaz con frecuencia por escrito de esta manera, como por ejemplo, «Lo sentimos, pero la página no se puede encontrar.» Otro ejemplo es el uso de avatares en el equipo basado en la automatización, como se puede encontrar en los sistemas de telefonía automatizada. Por ejemplo, cuando un sistema de respuesta de voz no puede entender lo que el usuario ha hablado, después de varios intentos se puede responder en tono de disculpa: «Lo siento, no le puedo entender.»

La idea básica en este modelo es tratar de alcanzar una comunicación fluida con la máquina siguiendo el proceso natural de comunicación persona–persona (familiar e intuitivo), intentando dotar a la interfaz con cualidades y gestos humanos reconocibles, como podrían ser el lenguaje natural, reconocimiento de imagen, texto, movimientos, etc. Este modelo destaca la idea del trato amigable con el usuario y la naturalidad de la interfaz. Se dice que la interfaz es amigable si el sistema reacciona ante el usuario de forma comprensiva, evitando los mensajes crípticos, poniéndose en su lugar y ayudándole a trabajar cómodamente, creando un ambiente de concordia en el que los mutuos errores se acepten y corrijan de la manera más cómoda para ambos. Natural se refiere a que la persona y el ordenador actúen como compañeros, a que se equilibre la creatividad y a que interactúen de manera grata, ejemplos de este modelo son los agentes animados de Microsoft Agent.

En general se utilizan tres técnicas en este modelo:

1. En la primera la interacción real humano-humano se utiliza, después de una meticulosa observación, como modelo y medio de recogida de información vital para las decisiones en el diseño. Esta interacción se estudia generalmente en espacios o laboratorios de observación, o bien en un ambiente controlado, como un centro de trabajo o un simulador, utilizando los datos posteriormente en el diseño de la interfaz del sistema en estudio.
2. El segundo método es más intuitivo y menos costoso. En él se hace un estudio comparativo entre la interacción humano-computadora y el humano-humano, en busca de aspectos desnaturalizados y difíciles para una persona, planteando entonces, para los procesos más críticos, métodos de comunicación mediante la

voz, lenguaje natural, etc.

3. El tercer método se basa en la propia experiencia de los investigadores o desarrolladores y en la aplicación de soluciones ya existentes en anteriores aplicaciones, lo que puede representar para el usuario el encuentro con nuevas formas de ver los procesos ya conocidos, simplificando incluso los de comunicación entre personas mediante un argot o jerga familiar al usuario del sistema.

El desarrollo de la interacción antropomórfica dependerá en gran medida de los avances tecnológicos y en la mejora de las técnicas de inteligencia artificial. Por otro lado, algunos usuarios, especialmente los más expertos, pueden rechazar sistemas que les resulten excesivamente amigables o dialogantes.

Eberts (1994) describe cuatro Modalidades de diseño en HCI, enfoques de diseño que se pueden aplicar a los diseños de interfaz de usuario para crear experiencias de usuario fáciles de usar, eficaz e intuitiva para los seres humanos. Estos cuatro enfoques incluyen el enfoque antropomórfico, el enfoque cognitivo, el enfoque del modelo predictivo, y la aproximación empírica.

2 Modelos evolucionados de interfaz

I. Modelos de interfaz de última generación. Aunque ya en el Capítulo II se hace un recorrido histórico por la historia de las interfases, Porta (2002) asegura que la última generación de interfaces gráficas de usuario se puede clasificar en tres posibles categorías aunque a veces se convierten en categorías superpuestas de acuerdo con el tipo de entradas y salidas que acepten o proporcionen.

a) Interfaces de usuario multimedia: proporcionan al usuario diferentes tipos de salidas (al menos dos). Están enfocados hacia el multimedia (como texto, gráfico, sonido, etc.). Hoy en día la mayoría de interfaces desarrolladas explotan el multimedia de alguna forma.
b) Interfaces de usuario multimodales: Esta categoría se puede considerar como un subconjunto de las interfaces de usuario multimedia. Para ser más estrictos, las de multimedia normalmente se concentran en el medio utilizado y las multimodales se concentran más en los canales de percepción (vista, oído, tacto, etc.) (Turk, Robertson, 2000).

Según el W3C[2], la Interacción Multimodal o **Multimodalidad** consiste en un proceso en el cual diversos dispositivos y personas son capaces de llevar a cabo una interacción (auditiva, visual, táctil y gestual) conjunta desde cualquier sitio, en cualquier momento, utilizando cualquier dispositivo y de forma accesible, incrementando así la interacción entre personas, y entre dispositivos y personas. Una definición de interfaz multimodal aparece en Montoro M., (2000): Estos sistemas tratan de implicar varias de las posibles

---

2 El World Wide Web Consortium, abreviado W3C, es un consorcio internacional que produce recomendaciones para la World Wide Web. Está dirigida por Tim Berners-Lee, el creador original de URL, HTTP y HTML que son las principales tecnologías sobre las que se basa la Web.

modalidades de comunicación que utilizan las personas. Estas modalidades comprenden habla, reconocimiento de gestos, reconocimiento de la posición de la mirada, del movimiento de los labios, de la expresión facial, de la escritura, etc.

c) Interfaz perceptiva de usuario. La fusión las anteriores interfaces introduce el concepto de Perceptual User Interface (PUI: en español interfaz perceptiva de usuario), que busca adquirir unas interfaces más naturales e intuitivas que aprovechan conocimientos de la forma natural en que las personas interacciona entre ellas y con el mundo. Las PUI deben aprovechar las capacidades perceptuales humanas y presentar la información y el contexto en forma natural y significativa. Esto significa que debemos ser capaces de entender la visión humana, la percepción auditiva, formas de conversación convencionales, capacidades táctiles, etc. De la misma manera, las PUI deben aprovechar los avances en la visión artificial, reconocimiento de voz y sonido, sistemas de aprendizaje y comprensión del lenguaje natural.

Las PUIs pueden utilizar la vista y el leguaje humano mediante el reconocimiento y generación movimiento de iris y de la voz. En la figura siguiente se muestra un ejemplo de interfaz perceptiva de usuario.

Figura 2: Ejemplo de Interfaz PUI.

Las técnicas de visión por computadora y la inteligencia artificial, animación gráfica y visualización, sensores de tacto y de retroalimentación con fuerza (hápticos), aprendizaje, modelado de usuario y gestión del diálogo son otras áreas propias de la interfaz perceptiva de usuario. Fuente (Turk and Robertson, 2000).

d) Interfaz de realidad aumentada (RA). Es una tecnología particular que permite agregar sensaciones, imágenes e información generada por computadora a la realidad normal percibida.

En visores especiales utilizados por los individuos (webcams, smartphones), imágenes, texto y objetos virtuales son producidos, otorgando información adicional al entorno real. Un híper-entorno, tridimensional e interactivo es generado por computadora, con objetos reales y virtuales en el cual los individuos son inmersos donde:

El usuario percibe el mundo como cualquiera, pero con información adicional: textos, 3D, imágenes estáticas o en movimiento, con los cuales también se puede interactuar a través de dispositivos simples y ya existentes en el mercado.

Analistas como *Forrester Research*, así como las empresas pioneras del diseño y entretenimiento ven a los conceptos de RA como la próxima evolución de las plataformas digitales e interactivas.

Figura 3: Imagen de Dispositivo e Interfaz de Realidad Aumentada.

e) Interfaz de Interacción Natural: En computación, una interfaz natural de usuario o NUI (por sus siglas en inglés Natural Users Interface) o bien en su expresión corta Interacción Natural (IN), son los terminos comúnmente usados por diseñadores y desarrolladores de interfaces para referirse a una interfaz de usuario que es efectivamente invisible, o se vuelve invisible con las intereacciones aprendidas sucesivamente por sus usuarios. La palabra natural se utiliza porque la mayoría de las interfaces de computadora utilizan dispositivos artificiales de control, cuyo funcionamiento se tiene que aprender. Una NUI en cambio se basa en que un usuario puede rapidamente hacer la transición de principiante a experto. Mientras que la interfaz requiere de aprendizaje, por lo tanto, “Natural” se refiere a un objetivo en la experiencia del usuario, que un usuario siente “como algo natural” mientras interactua con la tecnología”

Derivado del modelo antropomórfico, las interfaces INU están conformadas por todas aquellas expresiones que puede realizar el ser humano, más todas aquellas cualidades

físicas o corporales medibles hasta el presente, es decir, todos aquellos componentes propios de un ser humano. como sucede -por ejemplo- con el sistema Sixth Sense desarrollado por Pranav Mistry en el MIT, la consola de videojuegos Wii y otros sistemas como por ejemplo el Kinetic de la Xbox 360 de Microsoft.

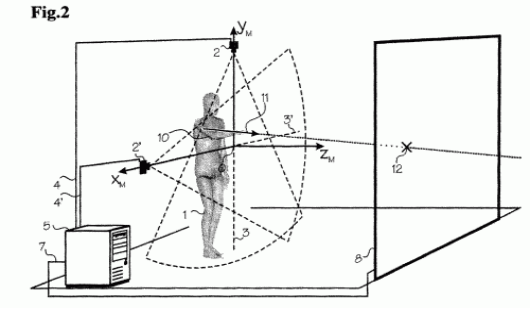


Figura 4 Diagrama de una INU

Existe un área de investigación en la Interacción humano-computadora enfocada en construir una interfaz que posibilite la participación de usuarios no entrenados tecnológicamente. Para estos usuarios potenciales, las técnicas de interacción de las que se haga uso no deben presuponer ningún conocimiento previo ni habilidad específica del usuario y ninguna habilidad tecnológica. La única habilidad interactiva del usuario es la que le permite interaccionar con otros humanos y es el tipo de interacción que se espera que desarrolle la en la *interacción natural,* (IN) que aglutina diversas disciplinas (ingeniería del conocimiento, lingüística, psicología, etc.).

Las interfaces de interacción natural, incluyen un subsistema de interacción con las siguientes características:

Autonomía de operación de sus componentes, soportada por una arquitectura multiagente.

Componentes de Interfaz**:** entrada directa de estructuras semánticas, sustituibles por módulos de procesamiento de voz y de lenguaje natural

Componente de diálogo: con procesamiento intencional y de acción combinada (Clark, 1996). La integración de los interfaces con la interacción se hará con actos comunicativos (Austin, 1962). Incluirá la implementación de varios modelos de conocimiento: diálogo, tareas, y sesión (Calle, García-Serrano, Martínez, 2006) Componente de situación: hará hincapié en la gestión de la circunstancia a través del aspecto material (espacio-temporal). Estará soportado por la tecnología de Espacio-Temporales (Bertino, Cuadra & Martínez, 2005), y aportará la información necesaria para el guiado de usuarios.

f) Interfaz con agentes pedagógicos animados con características humanas: Los sistemas de interacción con agentes inteligentes, derivados del enfoque antropomórfico y los modelos de interacción natural poseen diferentes características que son de gran

utilidad como medios instruccionales, alguna de las cuales son (Johnson et al., 2000):

1. Estos sistemas permiten actuar y dialogar con los agentes, de manera que en el momento de realizarse acciones, éstas pueden ser vistas desde diferentes ángulos.
2. Los estudiantes pueden realizar preguntas en cualquier momento
3. El agente está en todo momento *observando* el actuar del estudiante
4. El agente puede reconstruir y redefinir su actuación en cada momento a partir del actuar del estudiante.
5. El agente puede adaptar su actuación a situaciones inesperadas.
6. El estudiante puede tomar el control en cualquier momento.
7. En caso de errores el agente ayuda a que el estudiante aprenda de ellos

Figura 5 Agentes Pedagógicos inteligentes del Proyecto ***Lahystotrain***

Existen otras ventajas de estos agentes, tales como el manejo de emociones, apoyo efectivo del trabajo colaborativo, interacciones pedagógicas adaptables, etc. (Laureano-Cruces, Mora-Torres, Ramírez- Rodríguez, Gamboa-Rodríguez, 2009), (Velasco-Santos, Laureano- Cruces, Mora-Torres, Herrera-Bautista, 2010). (Ryokai et al., 2002a; Ryokai et al., 2002b).

2.1 Interacción hablada con un agente de interfaz

Este tipo de interfaz permite la interacción hablada con un agente informático, para ello incorporan reconocimiento del habla, comprensión del lenguaje natural, gestión de conversación y apariencia de personaje animado para simular mejor una interacción persona-persona (Cassel, 1999). Algunas de las motivaciones que han conducido al desarrollo de estas interfaces son:

1. Los agentes animados con interfaces conversacionales proporcionan un paradigma intuitivo de interacción ya que el usuario no necesita adquirir nuevos conocimientos.
2. Estas interfaces presentan redundancia y complementariedad en los modos de

entrada. Lo que aumenta la fiabilidad de la comunicación entre sistema y usuario.

3. Los usuarios encuentran estos sistemas más amigables y cooperativos. Los agentes autónomos pueden utilizar esa ventaja para entablar una conversación con los usuarios de forma más natural
4. Cuando se diseñan interfaces con personajes, la motivación última es conseguir trabajar con computadoras sin teclados, las cuales acepten entradas naturales no entrenadas y respondan en consecuencia, se necesitan personajes bien implementados para poder interactuar con ellos usando las múltiples facetas de la conversación natural. Una conversación cara a cara se caracteriza principalmente por el empleo del lenguaje, pero se emplean, otras técnicas o habilidades:
5. Los interlocutores emplean gestos con las manos para enfatizar o representar ideas, se dirigen miradas expresivas.
6. Utilizan variaciones en el tono o melodía de las palabras o frases articuladas. En la figura 6.6 se puede apreciar un ejemplo de expresiones de un agente:

Figura 6 La imagen muesta una de vistas del agente animado utilizado y modelado específicamente para la propuesta DECANO.

Según Cassel, (2000), Nickerson en 1976, fue de los primeros que argumentó acerca de la utilidad de la aplicación de esta metáfora a la interacción humano-computadora. Expuso una lista de las características que debería poseer una interfaz. Esas características entre otras son:

1. Poseer elementos de comunicación no verbal
2. Ser capaz de tomar la iniciativa, dar sensación de presencia e Incluir reglas de transferencia del control
3. Bajo estas premisas el diseño de la interfaz se puede lograr interfaces capaces de sostener una conversación. Estas interfaces transmiten emociones y se comportan en función de la demanda del diálogo, de su propia personalidad y de convencionalismos sociales. Las conductas que deben mostrar los agentes para conseguir desarrollar una conversación que resulte natural son muy extensas, pero se han de vertebrar alrededor de las siguientes pautas:
4. Emoción: El personaje puede exhibir expresiones faciales que denoten emoción y

gestos expresivos, por ejemplo, para aconsejar, animar o enfatizar algo al usuario (Laureano-Cruces, Mora-Torres, Ramírez-Rodríguez, Gamboa-Rodríguez, 2009)

5. Personalidad: La personalidad de un agente ha de ser elegida teniendo en cuenta sus tareas específicas en un contexto dado, pues la personalidad presenta al personaje con sus propias parcelas de conocimiento y perfiles de interés (Velasco-Santos, Laureano-Cruces, Mora-Torres, Herrera-Bautista, 2010)
6. Se deben considerar los objetivos de la conversación: La razón por la que el hablante está comunicando una cosa (es decir, el objetivo que tiene en mente el hablante) y la forma de esta comunicación están muy relacionadas. Las contribuciones a una conversación pueden ser de propuesta y de interacción (Montoro M. 2000)
7. A partir de este análisis se pueden inferir ciertos modelos que tienen una aplicación directa en los agentes. Es importante tener en cuenta lo que un personaje quiere expresar, se puede acompañar su discurso con cierta expresión facial o gesto corporal.

ANEXO: ETAPAS DE LA HISTORIA DE LA INTERFAZ GRÁFICA Y EJEMPLOS DE **INTERFACES** MÁS REPRESENTATIVAS DE CADA PERIODO**.**

PRIMER PERIODO : NACIMIENTO DE LA IGU

Primera Interfaz gráfica de usuario: Xerox Alto (1973)

En el año 1973, en el centro de investigación Xerox Parc, nacería la que es considerara la primera computadora que incluiría la primera interfaz gráfica de la historia. El **Xerox Alto** sería diseñado por un equipo formado por Ed McCreight, Chuck Thacker, Butler Lampson, Bob Sproull, y Dave Boggs.

La intención por parte de Xerox fue la de desarrollar un sistema informático lo suficientemente pequeño y transportable con capacidad para ser ubicado en una oficina. Debería tener capacidad para poder manejar un sistema operativo con interfaz gráfica y poder compartir información de forma sencilla. Con la materialización del **Xerox Alto** se había conseguido la primera aproximación al concepto de computador moderno.

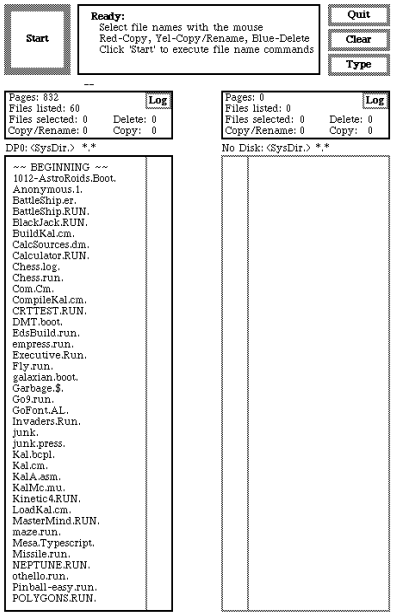


Figura 1: Interfaz Gráfica de Xerox Alto

Xerox Alto poseía una interfaz gráfica rudimentaria en blanco y negro, con la que se podía interaccionar mediante un ratón. Los botones estaban representados a través de formas textuales, de un modo muy simple. No fue implantado ningún sistema de ventanas en este modelo. La interfaz gráfica no presentaba elementos icónicos, ni pestañas, ni barras de desplazamiento en la navegación de la información. Este equipo fue la primera aproximación realizada al paradigma de interacción WIMP[1], sin llegar a una verdadera materialización de "la metáfora de escritorio".

Sin lugar a dudas sería un gran avance que sentaría las bases para los proyectos posteriores, los cuales evolucionarían hasta el modelo de interacción actual.

Intefaz gráfica de Gypsy

En 1974 se comenzó en PARC a trabajar en Gypsy, el primer editor de texto gráfico WYSIWYG (*What-You-See-Is-What-You-Get*, "lo que ves es lo que consigues"). Este principio se aplicaría con mucho éxito posteriormente a muchos otros procesadores de texto y otros editores de texto con formato permitiendo escribir un documento viendo directamente el resultado final, sin necesidad de esperarse hasta poder visualizarlo a través del resultado impreso. En 1975 los ingenieros de Xerox presentaron una demostración de un IGU "incluyendo íconos y el primer uso de menús emergentes".

Interfaz gráfica de Xerox Star 8010

En el año 1981, fue concluido el sucesor del Xerox Alto, el equipo **Xerox Star 8010**. Un grupo de doscientos desarrolladores, dirigidos por Don Massaro, serían los encargados de diseñarlo con el principal objetivo de incorporar las mejores características del Xerox Alto y que además fuese fácil de usar, con capacidad para automatizar y facilitar tareas de oficina.

Este producto, fue etiquetado como "la oficina del futuro", y entre los principales objetivos del proyecto, se encontraba, copiar e implementar el concepto de oficina virtual, buscando

1 Siglas utilizadas para abreviar el paradigma "*Windows, Icons, Menus and Point devices*", ventanas, íconos, menús e interfaces humanos.

además la facilidad de uso por parte del usuario.

Podríamos considerar a Xerox 8010 como la primera computadora que introduce una interfaz gráfica de usuario incluyendo y aplicando la metáfora del escritorio. Aunque no fue un éxito comercial, el equipo Star influenció de manera importante los futuros desarrollos, por ejemplo en Apple, Microsoft y Sun Microsystems.

El requisito más importante, que se pusieron como reto los integrantes del equipo, para el desarrollo de este proyecto realizado con éxito, fue el desarrollo e implementación del concepto WYSIWYG de forma muy acertada y adecuada en la interfaz. Éste consistía en buscar una analogía lo más fielmente posible entre la representación de la información que se ve en la pantalla y el resultado final que pueda tener un documento escrito tras ser impreso.

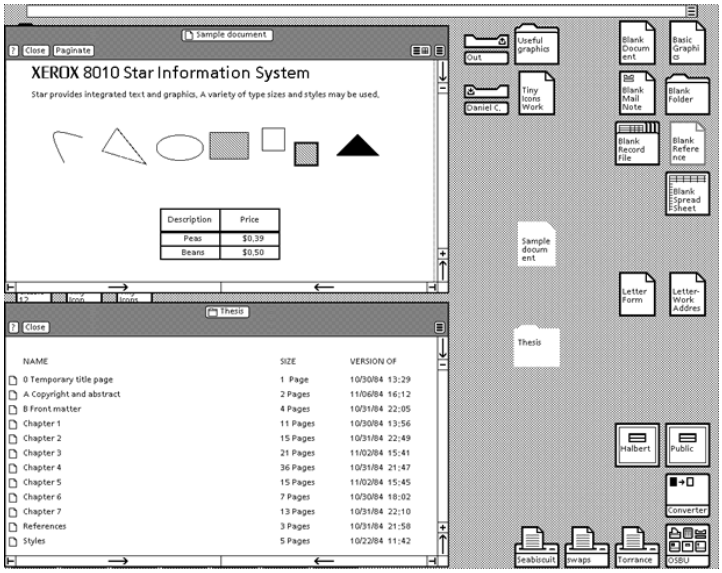


Figura 2: Interfaz de Xerox 8010 Star

De aquí partiría la idea de convertir a la computadora en una máquina apta para ser usada como medio de edición y publicación de contenidos, interés primordial de la empresa Xerox, especializada en sistemas de impresión.

A través de las investigaciones realizadas en el PARC, queda constancia de las intenciones de este centro de desarrollar una herramienta informática óptima que sustituyese el tradicional sistema de trabajo realizado en las oficinas. El Xerox Star, representaba la materialización de esas intenciones, sólo falló en algo, y es que su costo lo hizo

prácticamente inalcanzable e inviable para ser introducido en el mercado, por lo que su distribución estuvo limitada a centros de investigaciones y grandes instituciones, sin tener una verdadera repercusión comercial.

Lo que más podemos resaltar de este sistema son las características de la interfaz. El sistema Xerox Star disponía de los dispositivos de entrada: ratón y teclado, que ya habían sido incluidos en el Xerox Alto. Fue el primero en incluir un sistema de ventanas totalmente contemporáneo prácticamente como las podemos encontrar incorporadas en las interfases actuales, incluso con la posibilidad de sobreponer unas ventanas a otras y manejar múltiples elementos en el escritorio de trabajo.

La primera y más popular consideración, fue el uso de una metáfora del escritorio que representaba cada tipo de fichero a través de un ícono característico. Con esto, se aumentó la facilidad para relacionar y agrupar los tipos de ficheros de datos a través del desarrollo de una gramática visual coherente, que relacionó el tipo de fichero y la aplicación a la que pertenecían cada fichero de datos. De este modo se facilitó la visualización y posibilitó una mejor asociación y uso de los recursos sobre la interfaz por parte del usuario .

Otro de los hallazgos del sistema Star fue la de estandarizar, una serie de funciones generales que pudiesen ser asignadas a todo tipo de archivos, de modo que éstas sirviesen para facilitar la interacción con la computadora y de este modo, mejorar la usabilidad del sistema. Los comandos básicos estandarizados serían los de mover, copiar, abrir, borrar, mostrar propiedades y copiar propiedades.

Sin haber tenido una repercusión comercial, el sistema Star maduró y desarrolló el concepto de la metáfora del escritorio, desarrollando la mayor parte de los recursos interactivos usados posteriormente de forma masiva y que después serían los estándares implementados tanto por Apple como por Microsoft en sus respectivos sistemas operativos.

Segundo Periodo: Desarrollo de las IGU

Apple Lisa (1983) y Apple Macintosch (1984)

Después de sobrevivir a la década de los 70s habiendo completado su primera computadora la **Apple I (1976)**, y luego de haber invertido los beneficios en el diseño y

producción del **Apple II (1977)**, la empresa Apple Computers, formada en sus comienzos por dos aficionados a la electrónica llamados Steve Jobs y Steve Wozniak, lanzarían al mercado en el año 1983 el ordenador **Apple Lisa**, la primera computadora personal de Apple con interfaz gráfica de usuario integrada.

La historia de Apple es reciente y está directamente relacionada con la historia de la informática de consumo, ya que, fue la primera compañía que consiguió introducir en el mercado una computadora con un sistema operativo totalmente integrado con una interfaz gráfica de usuario. La novedad no está por lo tanto en el hecho de haber desarrollado la primera interfaz gráfica, como ya hemos visto, sino en realizar el primer prototipo de computadora personal con interfaz gráfica preparada para ser introducida en el mercado y poder ser vendida de forma masiva.

Apple diseñaría en primer lugar a **Apple Lisa**, orientada al mercado del trabajo de oficina en grandes empresas, y tras su fracaso, apostaría posteriormente por el modelo **Apple Macistosch**, orientado al mercado de la informática personal.

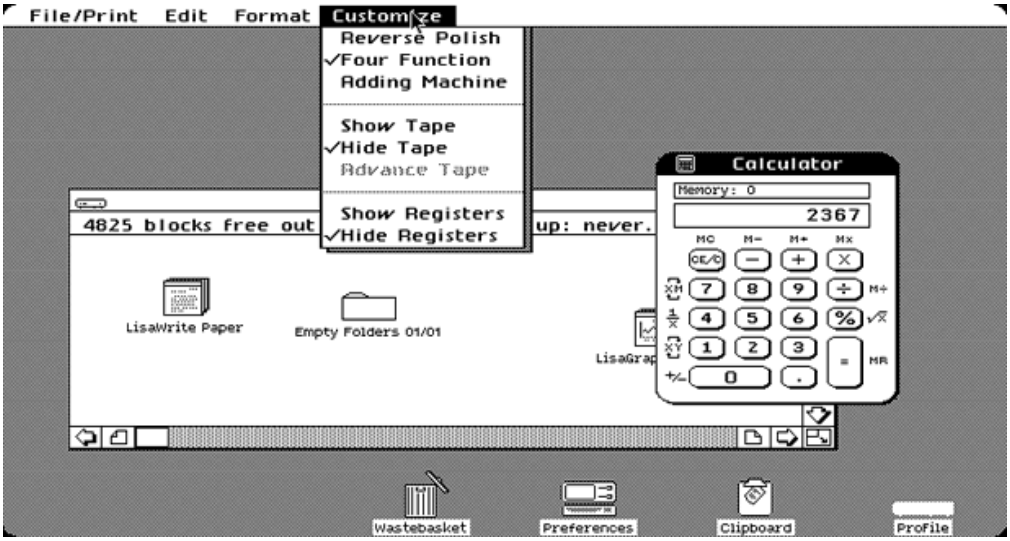


Figura 3: Interfaz de Lisa

El **Apple Lisa** fue una computadora adelantada a su época: introducía además de una interfaz gráfica, una serie de requisitos de hardware muy superiores a la tecnología de ese momento. La intención de la compañía fue diseñar y vender estos modelos a empresas, pero los costos eran elevados, y su reciente competencia con las computadoras introducidos por IBM, truncó sus intenciones. El resultado es que el **Apple Lisa**,

supuso un gran fracaso comercial para la compañía.

Paralelamente al desarrollo de Lisa, Apple diseñó otra gama de productos de bajo-costo, fáciles de usar y especialmente orientados a un consumidor medio: **Apple Macintosch**. La primera **Apple Macintosch** entraría en el mercado en el año 1984 y sería un éxito comercial.

Apple Macintosch tendría el acierto de convertirse en el primer sistema de publicación digital, todo ello gracias a su orientación gráfica, la integración y desarrollo con sistemas de impresión y la integración de software especialmente orientado a la edición.

Lisa introduciría la primera interfaz gráfica diseñada por Apple, perfeccionada posteriormente con la introducción de **Apple Macintosch**. Tanto Lisa como Macintosh, utilizaron el paradigma WIMP a través de la integración de una interfaz humana ratón-teclado. Usaron íconos para identificar aplicaciones en el escritorio, y trabajaban con el sistema de ventanas para representar aplicaciones y documentos en el escritorio, ambas computadoras fueron diseñadas teniendo el concepto de WYSIWYG como prioridad en el diseño.

Otra de la cosas que siempre ha preocupado a Apple y que incluyó Lisa y Macintosch fue la consistencia en la interfaz. Este concepto es integrado a la hora de organizar los comandos en los menús, donde se mantienen un orden que es aplicado de forma consistente, al grueso de las aplicaciones. Organizaciones y estructuras de este tipo en el diseño ayudan a recordar donde están los comandos, y favorecen los procesos de memorización, localización y uso de los mismos, reduciendo la curva de aprendizaje.

Otro aspecto con el diseño de la interfaz contemplado y usado por los ingenieros de Apple es la simplicidad (Powell, 1990; Schneiderman, 1992). Este principio, que ha regido las gramáticas del diseño visual desde sus orígenes, lo usa Apple eliminando información redundante sobre el interfaz. Un ejemplo de ello es la síntesis y estructuración de las posibles respuestas implementado en los menús de confirmación, en un momento en el que el usuario tenga que contestar sobre algún proceso del sistema. De este modo se consigue eliminar la información redundante y ayudar al usuario a visualizar mejor las posibles respuesta y a tomar una decisión de forma clara y concisa.

Next GEM (1985)

En el año 1985 fue desarrollado el entorno gráfico GEM, por la empresa Digital Researchs. La interfaz de GEM se integraba con el sistema operativo DOS bajo plataforma IBM. Más tarde esta interfaz sería adaptada en las consolas o computadoras Atari St igualmente.

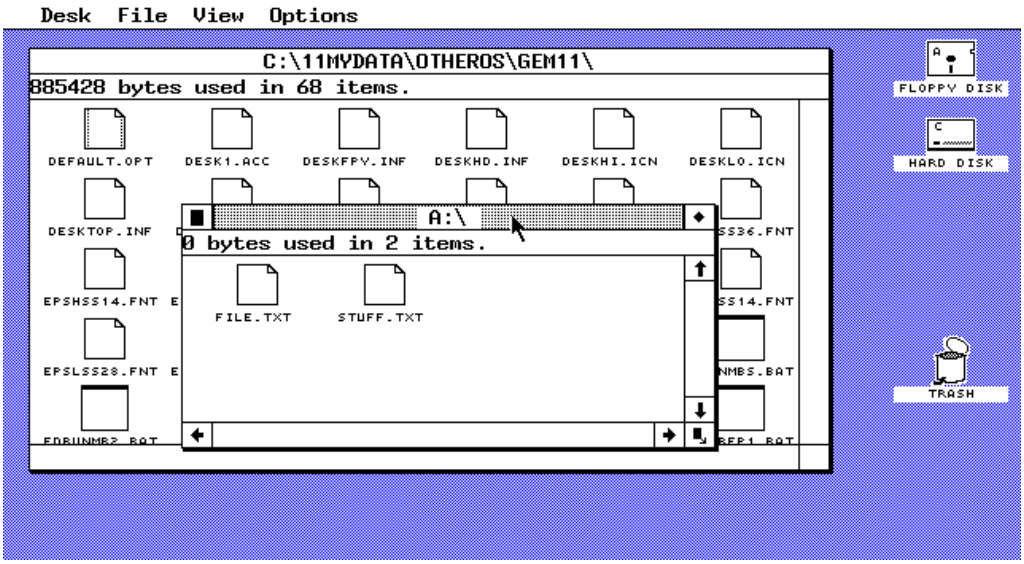


Figura 4: Intefaz de Next GEM

La interfaz gráfica de **GEM** es prácticamente una copia de la de Mac de esos años. Desarrolla la metáfora de escritorio manteniendo una serie de íconos, como son la papelera de reciclaje y para acceder a los dispositivos de almacenamiento de datos como los discos flexibles y el disco duro.

Usa el sistema de ventanas para navegar por la información similar al gestor de archivos *finder*[2] desarrollado por Apple. El sistema de ventana incluía barras de arrastre[3], posibilitando el solapamiento de ventanas en la interfaz, y acciones de manipulación directa de archivos entre ellas. También poseía un menú fijo en la parte superior donde se organizan diferentes comandos para la interacción con opciones de visualización al modo

2 *Finder* es el nombre que recibe el gestor de archivos que incluye Apple en sus sistemas operativos que permite navegar y manipular los archivos del sistema.

3 Las barras de arrastre son elementos interactivos que posiblitan la navegación de la información. En la tercera parte de este trabajo está debidamente desarrollado.

de **Apple Macintosh** y como más tarde también lo haría el sistemas operativo **Windows**.

GEM incluía también algunas aplicaciones especialmente diseñadas, como son una calculadora y un reloj, aunque éste, al contrario de lo que ocurre actualmente, no aparecía visible en el escritorio de trabajo.

Amiga WorkBench (1985)

En este mismo año, sería lanzada al mercado la famosa **Amiga** por la empresa Commodore, integrada con una interfaz gráfica llamada **WorkBench**. Esta interfaz fue diseñado en cuatro colores: Blanco, negro, azul y naranja y especialmente adaptada para ser visualizada en una gama ámplia de monitores y televisores a través de un diseño en alto contraste.

Figura 5: Interfaz de Workbench

En su interfaz se observaba una metáfora de escritorio completa de aspecto rudimentario. Usaba el sistema de ventanas con posibilidad de solapamiento, incluyendo barras de

arrastre, y un botón de cierre en la esquina superior izquierda. Adicionalmente también contaba con íconos de acceso directo en el escritorio y tenía la peculiaridad de usar íconos (en forma de gavetas) para representar las carpetas en el escritorio. En la parte superior mentenía igualmente una barra con informaciones técnicas críticas del sistema y algunos íconos para abrir y cerrar aplicaciones.

En versiones posteriores se mejoraría mucho esta interfaz, adaptada a las posibilidades gráficas de las computadoras Amiga.

Microsoft Windows 1.0 (1985)

La ampliación de cuotas de mercado de mano de IBM, hizo posible la necesaria colaboración entre ésta, y la empresa Microsoft, la cual adaptaría su sistema operativo **MS-DOS** a una interfaz gráfica de usuario para operar sobre ordenadores IBM en el año 1985. Este sistema operativo sería **Windows 1.0**.

Su interfaz gráfica traía incluido un administrador de archivos, una calculadora, un calendario, un reloj, un block de notas, y un emulador de terminal (antigua Interfaz de línea de comandos).

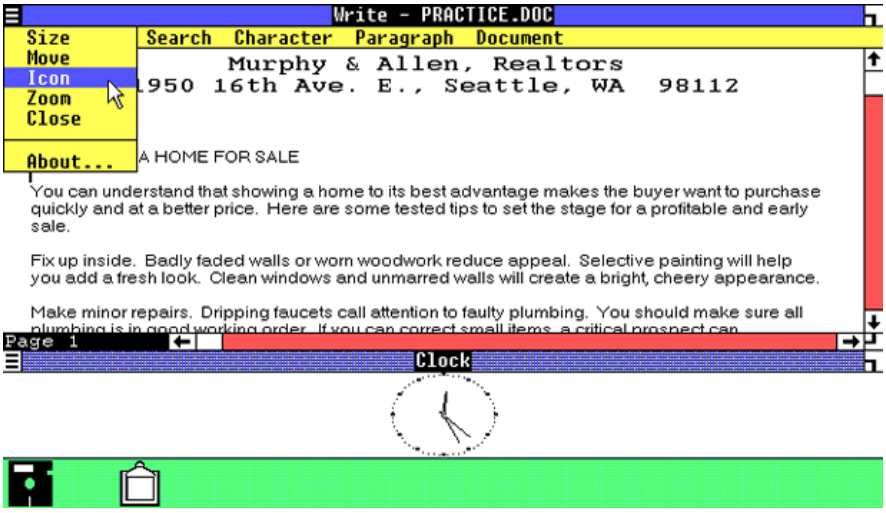


Figura 4: Interfaz Window 1.0

Las interfases de estas primeras versiones de **Windows**, presentan un aspecto rudimentario, muy alejado del aspecto gráfico actual. Se hace uso muy limitado de

íconografía, estando caracterizada por la ausencia de representación de íconos en los archivos, haciendo imposible asociar archivos y aplicaciones.

Una peculiaridad de su sistema de ventanas es que establece su estado por defecto de forma maximizada[4], al contrario que otras interfases donde lo habitual es traer al frente ventanas de tamaños diversos, que permanecen solapadas en la pantalla.

Las ventanas incluían un ícono de redimensión en la parte superior derecha, y en la izquierda ofrecían la posibilidad de acceder a *menús contextuales*[5]. Cada ventana, incluye un menú de comandos en la parte superior, al contrario de los ejemplos comentados con anterioridad, en el interior de la ventana, no en el escritorio de modo fijo y general.

Una de las novedades incluidas es una serie de íconos alienados en la parte inferior de la pantalla que van representando las aplicaciones activas en el sistema. Sin una estructura visual clara ni una asociación definida entre ellos, esta ligera alienación de íconos parece ser el origen de la barra de tareas, que sería introducirían con tanto éxito en la versión 95 de su sistema operativo **Windows.**

Commodore GEOS (1986)

En el año 1986 nació GEOS, una interfaz gráfica primeriza desarrollada especialmente para la computadora personal Commodore 64, la cual tendría gran repercusión de ventas en el mercado de las computadoras de escritorio, siendo adaptada posteriormente para poder funcionar bajo los modelos de IBM.

GEOS incluye varias aplicaciones, como un calendario y un procesador de textos. Al contrario que con el resto de interfaaes gráficas, los usuarios de esta interfaz no emplean de forma habitual el ratón como interfaz humana, sino teclado y joystick[6] como dispositivos

4 La maximización en el contexto de las interfases gráficas, es uno de los estados posibles del elemento ventana, en el cual se extiende al total de la interfaz, solapando y sobreponiéndose a otras ventanas.

5 Menú contextual o flotante, es aquel menú que surge en el contexto de una aplicación, normalmente asociados a íconos concretos flotando en la pantalla. Este menú es un elemento vivo del sistema operativo ya que se va modificando añadiendo nuevos elemento al menú contextual a medida que instalamos nuevos programas.

6 Es un dispositivo de interfaz humana que posibilita la interacción. También es conocido como una palanca

de interacción.

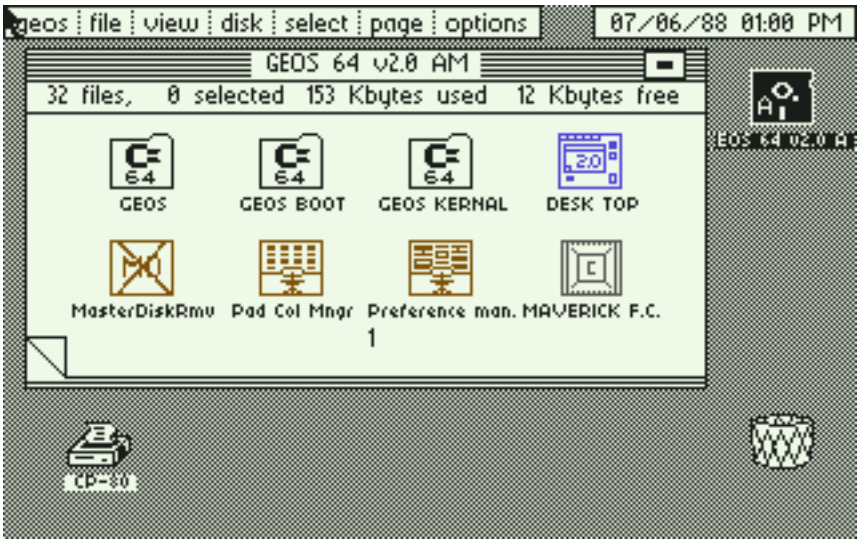


Figura 5: Interfaz de GEOS

GEOS permite posicionar en el escritorio íconos de las aplicaciones y funciones más importantes como son la de impresión, el acceso a dispositivos de almacenamiento y la papelera de reciclaje. No permite posicionar documentos de usuario en el escritorio.

El sistema de ventanas es básico, no permite redimensionar y permanece a un tamaño fijo en la pantalla. Usa un sistema de scroll, para la navegación por ventanas muy básico, el cual consiste en dos botones con flechas indicadoras en posición vertical. Usa la representación simbólica de los documentos a través de íconos, permitiendo diversos sistemas de visualización en el interior de las ventanas. También dispone un menú fijo de acceso a comandos globales del sistema en la parte superior al igual que hemos visto en otros interfaces precedentes.

Risc OS (1987)

En el año 1987 Arcon Computers, desarrollaría la primera versión del **RISC OS**, con una interfaz gráfica de usuario llamada **Arthur.**

Arthur estaba diseñada a color y usaba como medio de interacción un ratón de tres botones. A la hora de conceptualizar los menús de comandos y acciones, Arthur elegía la estrategia de ubicar las acciones no en menús fijos en el escritorio, sino en menús

de control que permite desplazar manualmente, y con gran rapidez, el cursor en una pantalla de computadora o video-juego; se usa especialmente en programas informáticos de juego.

contextuales, activados con el botón central del ratón. Esto es lo que da a esta interfaz un aspecto más moderno y acorde con los paradigmas interactivos que se usan actualmente.

Figura 6: Interfaz de Arthur

Otra de las novedades que incluyó **Arthur** fue una barra de tareas, hoy día elemento imprescindible de la interfaz gráfica, donde se posicionaban los íconos que resumen las tareas que se están ejecutando en el sistema. Es una barra de navegación global, que permite acceso directo a las diversas aplicaciones que están activas, y menús de posibles tareas. Los sistemas windows la harían popular, y actualmente elemento fundamental de cualquier interfaz gráfica de usuario.

NEXT STEP (1989)

La empresa Next Computer Inc. desarrolló en el año 1989 una interfaz gráfica para sus propias computadoras personales llamadas Next. La interfaz gráfica funcionaba gracias a un visualizador PostScript[7], que la empresa Next diseñaría en base a los desarrollos realizados por la empresa Adobe.

---

7 Con el Visualizador de imágenes puede ver, editar, imprimir y convertir los formatos de archivo de imágenes monocromas y en color, así como archivos PostScript.

Su sistema de ventana dispone el menú de navegación(scroll) *al lado izquierdo*. Incluye un original sistema para navegar por el sistema de archivos que consiste en ir añandiendo columnas para representar la jerárquía además de un *scroll* horizontal para la navegación.

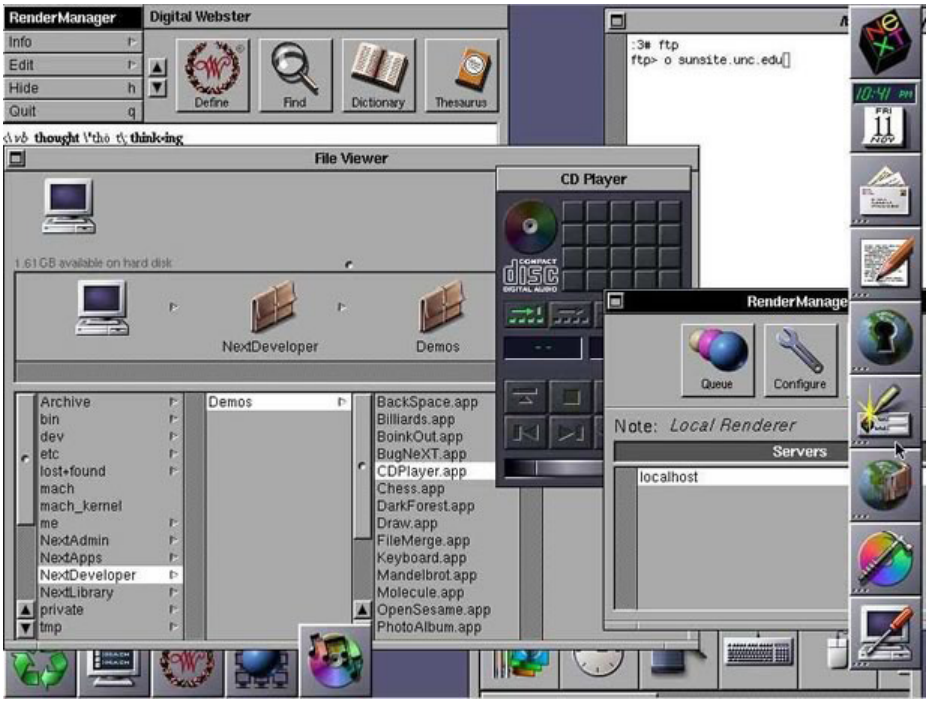


Figura 7: Interfaz de Next

También aparece un menú flotante en la esquina superior izquierda, al contrario que los ordenadores Macintosch no permanece visible de forma fija dispuesto horizontalmente, sólo aparece una vez activada alguna aplicación ocupando una franja vertical.

El elemento más importante introducido por la interfaz NEXTSTEP sería el ***Dock***. *Dock* significa literalmente, resorte, y es una barra especial que muestra en forma de íconos los programas más usados. Incluye siempre el ícono para mostrar el escritorio y la papelera de reciclaje. Esta barra muestra siempre el estado de los programas a través de algunos signos gráficos, indicando si están activos o inactivos en un momento dado. La interfaz gráfica del sistema Mac Os X incluyó posteriormente este elemento.

Windows 3.0

Esta versión apareció en 1990 y demostró el interés que Microsoft tenía en los entornos visuales, y de hecho aprovechó las prestaciones de los procesadores 386, que podían acceder a más de 640 KB y con las que lograron por ejemplo resoluciones de hasta 1024×768 píxeles. La interfaz también cambió de diseño e íconos, con un aspecto notablemente mejor al de las ediciones previas.

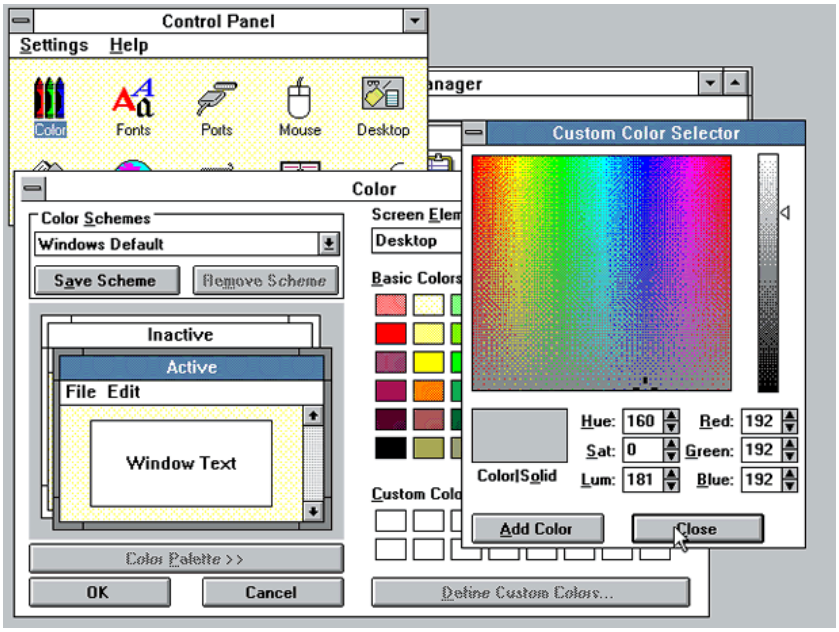


Figura 8: Interfaz de Windows 3.0

Tercer periodo: Automatización y personalización.

Windows 95 (1995)

El sistema operativo windows 95 fue lanzado por Microsoft en octubre del año 1995. Este sistema operativo significa el inicio del imperio Microsoft en el mercado del software informático. Microsoft consigue integrar en Windows 95, el sistema operativo MS-DOS con una interfaz gráfica de forma coherente. Windows 95 tiene una clara orientación a redes, por lo que vendrá integrado con el software Internet Explorer, que sustituirá al gestor de archivos dispuesto anteriormente.

Un cambio importante que introduce este sistema operativo es convertir la interfaz inicial

orientada a aplicaciones, en una interfaz *orientada a objetos* (Lineback, 1995). Este es un cambio importante en la interfaz de Microsoft que incide directamente sobre los procesos de interacción, ya que ahora, para abrir un documento, no es absolutamente necesario seguir el orden de abrir la aplicación y posteriormente el documento. El usuario puede interactuar directamente con el documento-objeto, representado a través de un ícono, en el escritorio y por lo tanto, abrirlo o ejecutarlo desde allí.

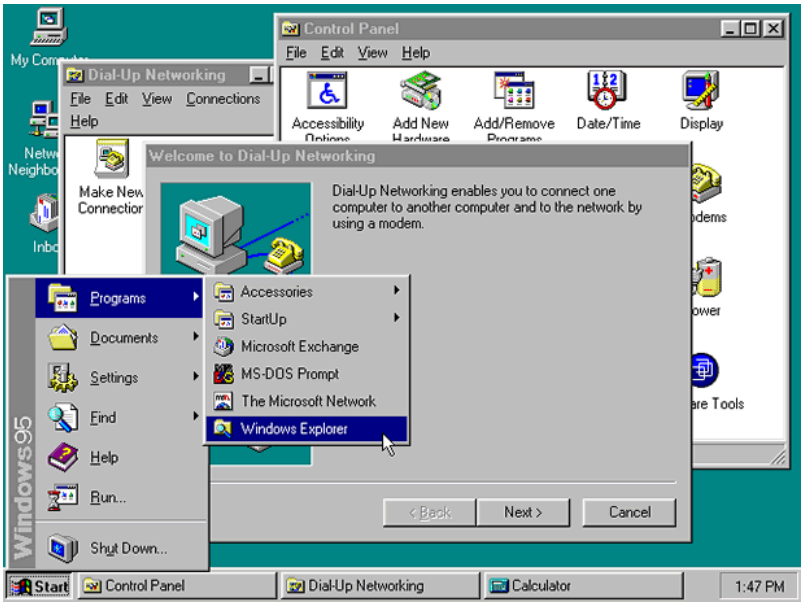


Figura 9: Interfaz Windows 95

Ésto tiene implicaciones sobre la ampliación del paradigma WIMP, provocando cambios importantes. Ahora el escritorio por ejemplo, respecto a versiones anteriores, no sólo sirve para representar aplicaciones minimizadas o activas, sino además íconos que pueden ser manipulados, agrupados, asociados y activados desde allí.

Otro mérito de Windows 95 es haber adaptado a la metáfora del escritorio la manipulación de mayor número de variables del sistema. La interfaz representa e implementa a través de sistemas de ventanas e íconos, la posibilidad de realizar cambios sobre la configuración de partes importantes del sistema, como configuración de redes, o cuestiones relacionadas con la administración del equipo, normalmente ajenas al usuario medio, y normalmente manipuladas a través de una interfaz de línea de comandos.

Otra novedad que introdujo la interfaz gráfica de Windows 95 fue el ***menú de inicio***, al que han sido asociados, en forma de árbol, el grueso de las aplicaciones, archivos y

funciones del sistema. El *botón de inicio* será uno de los grandes hallazgos de Microsoft que mantiene actualmente en todas las interfaces introducidas con sus sistemas operativos.

Otro elemento importante para la usabilidad del sistema tiene que ver con la posibilidad de incluir nombres largos en los archivos. Este elemento desde un punto de vista ergonómico, supone un acercamiento al lenguaje verbal del usuario, y por lo tanto una mejora a la hora de identificar elementos de forma textual.

Los dos elementos con los que identificamos este período son desarrollados por Microsoft de la siguiente manera:

Por un lado Windows 95 introdujo el ***asistente***. Un asistente es una pequeña aplicación que a través de un sistema de ventanas y menús, realiza un tipo de trabajo sobre el sistema, ayudando al usuario con parte de la tarea de configuración. Los asistentes, son intermediarios cuyo objetivo es recoger información por parte del usuario, y tomar decisiones adecuadas respecto a la ejecución de alguna tarea específica. Podríamos tomar al asistente como la aproximación a un autómata inteligente.

Por otro lado, Windows 95 posibilitó personalizar el escritorio, y algunas variables más de la apariencia del sistema operativo, entrando de lleno en los procesos de customización ya comentados.

BeOS (1995)

El sistema Operativo **BeOS**, fue desarrollado por la empresa de origen francés Be Incorporated, en el año 1995, diseñado principalmente para trabajar de forma eficiente sobre aplicaciones multimedia en la plataforma BeBox, con una clara orientación gráfica que haría buen uso de su entorno gráfico. A pesar de constituir un producto excepcional, la empresa Be tuvo que declararse en bancarrota debido a las estrategias de mercado realizadas por sus competidores.

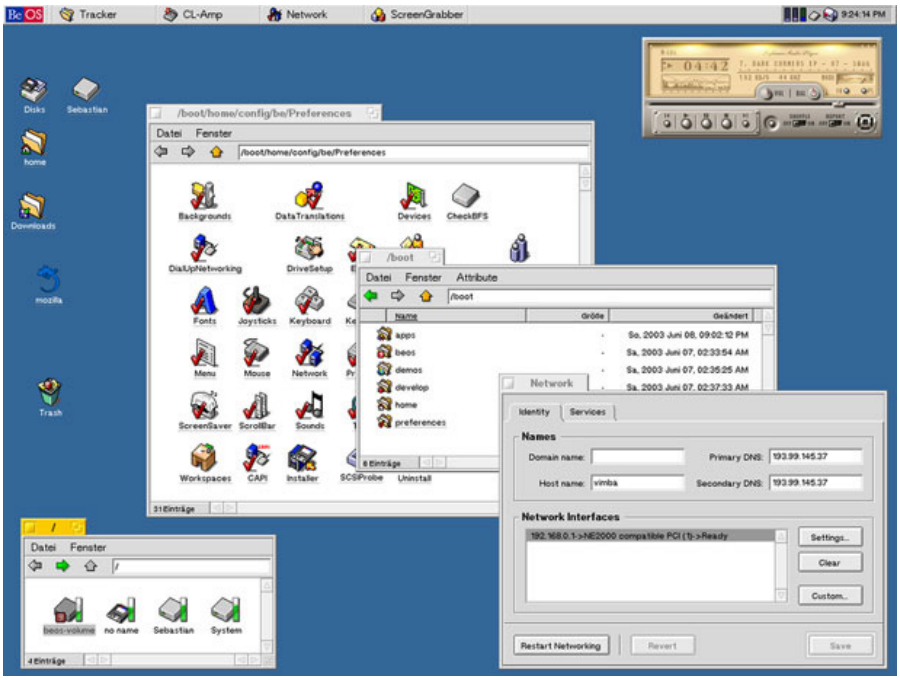


Figura 10: Interfaz de BeOS

La interfaz gráfica tiene un aspecto gráfico acabado y bien desarrollado. BeOS introduce de forma elegante todos aquellos hallazgos realizados hasta el momento:

- Incluye una ***barra de tareas*** en forma de menú, situada en la parte superior derecha, la cual incluye el logo del sistema operativo, botones especiales representando las aplicaciones activas minimizadas, y un área especial para mostrar la hora, y el uso del proceso por parte del sistema.

- Los íconos están representados en perspectiva isométrica y mantienen una gramática concisa respecto a las aplicaciones y tipo de documentos.

- BeOS permite posicionar documentos y archivos en el escritorio, tal cual introdujo Windows 95. Para navegar por los archivos del sistema, tiene su propio gestor de archivos, similar a *Internet Explorer* o el *Finder* introducidos por Microsoft y Apple respectivamente.

Al igual que el resto de interfases de este periodo, BeOS permite personalizar el aspecto de la interfaz, cambiando la apariencia de las barras de scroll, los comportamientos de los

menús, insertando imágenes de fondo, etc.

KDE (1998)

El proyecto KDE (K - Desktop Environment) nació en el año 1996, dentro del contexto del software libre, de mano del desarrollador alemán *Matthias Ettrich*. El sistema de ventanas KDE (K desktop Environment) fue lanzado en su primera versión dos años más tarde. El objetivo del proyecto era desarrollar una interfaz gráfica que operara sobre sistemas operativos Unix, especialmente GNU/LINUX y que posibilitara un método de interacción amigable con la computadora similar a los que ofrecen Windows o Mac OS en otras plataformas.

El escritorio K inicial contenía un panel (una barra de tareas y lanzador de aplicaciones) un escritorio que permitía organizar y posicionar íconos en él. También incluyó un gestor de archivos y un gran número de utilidades.

Al contrario de lo que ocurría en la interfaz de Windows, KDE activaba en el escritorio las aplicaciones bajo *una sola pulsación de ratón*, y no dos, como es habitual en otras plataformas. Esto la hace más coherente con los modelos de interacción provenientes de la Web, los cuales están basados en hipervínculos, normalmente activados con una sóla pulsación.

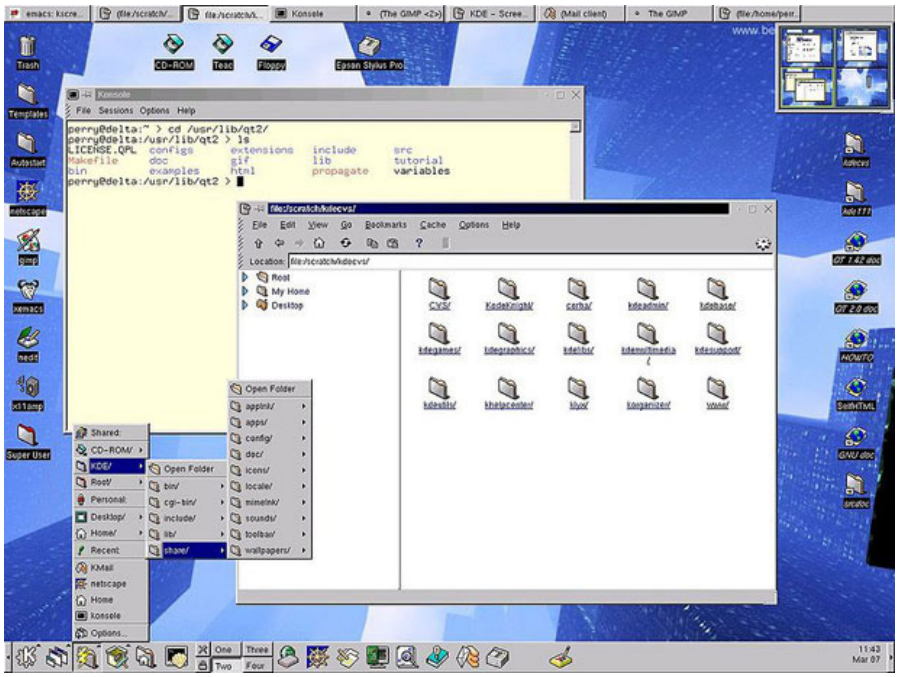


Figura 11: Interfaz KDE 1998.

Como ya se comentó, KDE disponía de una barra de tareas, más próxima a la propia de Windows que al Dock de Macintosch. Una compleja barra, que dispone de un botón de inicio, un área de íconos de aplicación, un área para ventanas minimizadas, y una última parte para tareas especiales.

El menú de inicio es similar al de Windows posibilitando el acceso al grueso de las aplicaciones Linux, a través de menús flotantes.

El sistema de ventanas es similar al del resto de interfases: permite minimizar, maximizar, cerrar, sobreponer, y la manipulación directa sobre las mismas.

Como interfaz propia inscrita en este tercer período, KDE fue preparada para ser customizable en todos sus aspectos gráficos. Quizás uno de los aspectos que más identifique su interfaz, es su gran flexibilidad para adaptarse a las necesidades del usuario. Permite modificar la imagen del escritorio, cambiar los íconos correspondiente a cada aplicación, cambiar la apariencia general de todos los elementos del mismo modo que Aqua de Apple o Windows XP de Microsoft. Permite introducir imágenes de fondo personalizadas sobre menús y áreas de control que no son habituales en otros sistemas. Actualmente KDE se ha convertido en el interfaz gráfico predilecto de los sistemas UNIX

para acompañar a las versiones más recientes de GNU/LINUX.

GNOME (1999)

GNOME es el nombre de la interfaz gráfica desarrollada originalmente por *Javier de Icazas* y *Federico Mena*, ambos mexicanos, y fundadores de la fundación Genome, la cual se creó al igual que el proyecto KDE, con el objetivo de dotar de un entorno gráfico de escritorio y una plataforma de desarrollo de aplicaciones *totalmente libres* en sistemas operativos GNU/LINUX.

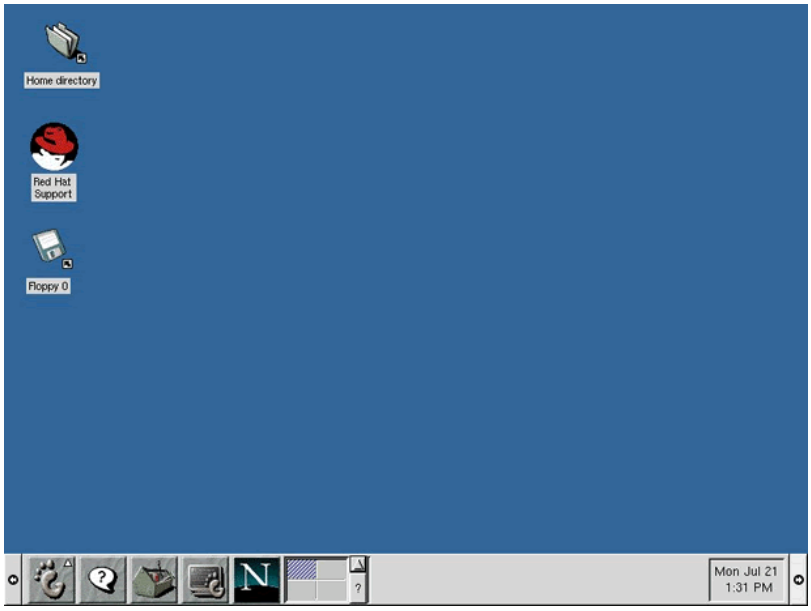


Figura 12: Interfaz de GNOME

El escritorio GNOME es similar en su infraestructura, al de Windows. Dispone de los acceso a las aplicaciones más importantes representadas a través de sus íconos, permitiendo posicionar elementos sobre el mismo.

En sus primeras versiones mantiene un aspecto gráfico relativamente tosco, botones de aspecto angular, con efecto “3d” como el incorporado en las primeros windows y

Macintosh, alejado del look que alcanzarán MAC OS X o Windows XP posteriormente.

Mantiene una barra de tareas, que incluye botón de inicio, área de aplicaciones, y área para las ventanas minimizadas en la parte inferior. Por defecto tiene un tamaño adecuado para posicionar en la barra dos filas de elementos, al contrario de la de Windows que normalmente permite por defecto la inclusión de elementos en una sola fila.

El sistema de ventanas incluye un cierto estilo contemporáneo, redondeando las esquinas, y añadiendo en la cabecera algunos degradados. Mantiene a la derecha, al igual que el sistema de navegación de Windows, los tres controles básicos de manipulación de ventanas. En las ventanas de confirmación, donde hay que dar respuesta al sistema, dispone los botones, en horizontal, al contrario de lo que ocurre normalmente en sistemas Macintosh.

GENOME está diseñado para ser customizable. Permite al igual que el resto de interfases contemporáneas, modificar las imágenes de fondo del escritorio, cambiar todo el aspecto de la interfaz, como son la apariencia de los botones, ventanas, y barras de navegación (*scroll*).

MAC OS X (2001)

El Sistema Operativo MAC OS X fue lanzado con las computadoras Apple Macintosh en el año 2001, y su arquitectura está basada en tecnología Unix al contrario de sus versiones anteriores.

MAC OS X, no sólo cambia su arquitectura interna, sino además renuncia a toda la íconografía desarrollada hasta el momento para introducir un nuevo entorno gráfico denominado ***Aqua***, el cual se inspira en las formas del agua del mar.

El menú superior clásico de acceso global, donde se posiciona en forma de menú textual las variables más importantes sobre archivos y procesos del sistema, es una de las pocas cosas que mantiene la interfaz de MAC OS X respecto a sus versiones anteriores.

Figura 13: Interfaz de MacOS X

El primer elemento diferenciador que introduce la interfaz de Macintosh es una barra de tareas especial, también llamada *Dock*, ya introducido por la interfaz de NEXTSTEP. Este elemento aparece generalmente posicionado en la parte inferior central del escritorio, y mantiene accesos directo a las aplicaciones más usadas en el sistema a través de íconos. En este caso, Dock, permite alojar las ventanas minimizadas de los documentos que se estén usando en ese momento.

El sistema de ventanas lleva ahora los controles situados en la parte superior de cada ventana, posicionados en el lado izquierdo. Estos representan, al igual que en los sistemas Windows, la función de “cerrar ventana”, “minimizar ventana” y “maximizar ventana”, de izquierda a derecha.

En relación a la customización de la interfaz, MAC OS X ofrece varios tipos de visualización de la información. Dentro de cada ventana existe la posibilidad de visualizar los elementos en forma de íconos, en forma de árbol, y en modo multicolumna, un modo peculiar que proviene igualmente de NEXTSTEP en el que cada nivel de la jerarquía de la estructura de los datos es representado a través de los elementos incluidos en cada columna.

MAC OS X tiene la propiedad de cambiar de modo que toda la interfaz pueda ser adaptada por el usuario a diferentes estilos, entrando de este modo en los procesos de customización

ya comentados. *Aqua* introduce en todo la interfaz de Macintosh transparencias, formas redondeadas en los acabados de las ventanas, íconos con formas sinuosas y degradadas que trasciende de algún modo el aspecto tosco y plano de los acabados con efecto tridimensional* predominantes en la generación anterior de interfases gráficas.

WINDOWS XP (2001)

El Sistema Operativo **Windows XP** fue lanzado por Microsoft en el año 2001 y pertenece a la familia de sistemas operativos NT desarrollados con la intención de disponer de una alternativa capaz de competir en calidad con los sistemas Unix, muy superiores a los primeros Windows en su arquitectura.

Windows XP se preocupó por adaptar la interfaz al perfil del usuario. Un ejemplo de esto es el modo es que va colocando aplicaciones en el menú de inicio, según el uso frecuente que el usuario vaya haciendo de ciertas herramientas. También incluyó procesos automatizados, para actuar con “inteligencia” en algunas áreas de la interfaz gráfica como ocurre en la barra de tareas, aglutinando bajo un mismo ícono varias ventanas abiertas de la misma aplicación, ahorrando espacio, pero volviendo del mismo modo más compleja la interacción con la información. Es apropiado mencionar que este tipo de recursos son eficientes partiendo del supuesto de que el usuario esté familiarizado con los procesos de interacción, y no para una persona que se inicia en el proceso. Ocultar un elemento exige al usuario saber donde se encuentra éste, y esto supone por lo tanto un aprendizaje adicional.

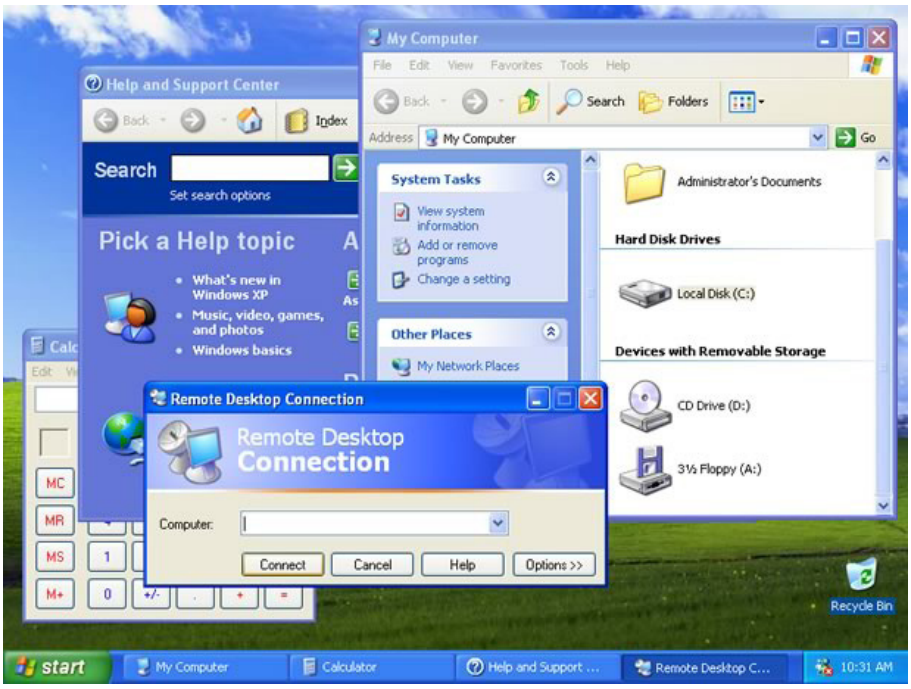


Figura 14: Interfaz de Windows XP

Otro de los elementos novedosos, relacionados con la automatización inteligente del interfaz que introduce **Windows XP** son sus ***mascotas***. A través de diversos personajes, especialmente el perrito, Microsoft introduce la metáfora del ayudante, un autómata que busca orientar al usuario en tareas básicas como buscar archivos, dando consejos al usuario a la hora de realizar alguna acción concreta sobre el sistema.

Windows XP representa el paradigma de interfaz customizable, adaptado para adoptar diversas “pieles” a la hora de representar los elementos comunes que aparecen representados en ella según los gustos del usuario. Viene “vestido” con una “piel” ergonómica, convirtiendo a todos los elementos del interfaces a un estilo orgánico y redondeado muy similar al Aqua del sistema operativo de Apple, sin entrar en tantos detalles de transparencias pero preocupado por trasmitir un “aire” moderno al interfaz alejado del estilo geométrico con aspecto tridimensional desarrollado en la anterior generación de sistemas operativos.

CUARTO PERIODO

KDE 3 (2002)

KDE 3.0 fue publicado en noviembre del año 2002, y es la evolución de KDE 2. El desarrollo de esta serie fue mucho más largo que el de la anterior. Los cambios de API entre KDE 2 y KDE 3 son menores. El aspecto de la interfaz no varió hasta KDE 3.1, en el que consta una importante mejora referente al tema visual: Keramik fue incluido como nuevo tema por omisión junto con el conjunto de íconos Crystal GT y el antialisado[8] de fuentes. En KDE 3.2 Crystal GT fue reemplazado por Crystal SVG. En KDE 3.4 Keramik fue reemplazado por Plastik. Todas las versiones de KDE 3 se basan en Qt 3, que solo fue liberado para GNU/Linux y sistemas operativos tipo Unix, incluyendo Mac OS X.

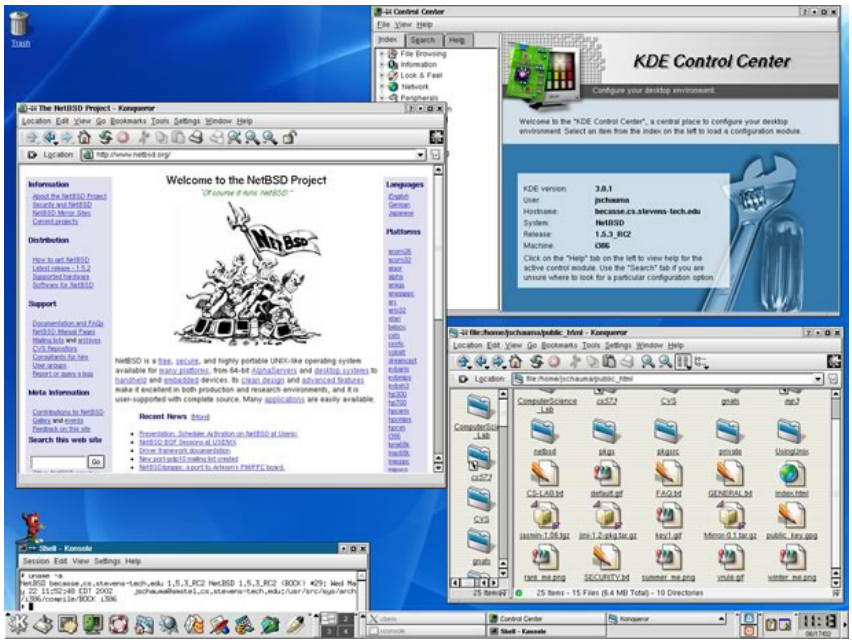


Figura 15: Interfaz de KDE 3

Salvo algunas aplicaciones de terceros, el desarrollo de KDE 3 ha cesado y sus desarrolladores ya no aplican ningún tipo de mantenimiento sobre este.

Windows Vista (2007)

8 El *antialiasing o antialisado* se aplica a menudo en la representación de texto en una pantalla para mostrar contornos suaves que emulan mejor la apariencia del texto producido por métodos convencionales de impresión de tinta sobre papel.

Esta versión se enfocó para ser utilizada en equipos de escritorio en hogares y oficinas, equipos portátiles, tabletas y equipos *media center*.

El proceso de desarrollo terminó el 8 de noviembre de 2006 y en los siguientes tres meses fue entregado a los fabricantes de hardware y software, clientes de negocios y canales de distribución. El 30 de enero de 2007 fue lanzado mundialmente y fue puesto a disposición para ser comprado y descargado desde el sitio web de Microsoft.

Figura 16: Interfaz Windows Vista

La aparición de Windows Vista viene más de cinco años después de la introducción de Windows XP, es decir, el tiempo más largo entre dos versiones consecutivas de Microsoft Windows. Uno de los fracasos mas certeros en la historia de Microsoft, sin embargo un gran avance en la GUI. Bastante animación en 3D, transparencia con Aero, y una barra lateral con Widgets.

Mac OS X Leopard (2007)

Fue anunciado el día 6 de junio de 2005 en la *Worldwide Developers Conference* WWDC (Conferencia mundial de desarrolladores) y lanzado el 26 de octubre de 2007. Leopard se encuentra disponible en 2 formas: una versión de escritorio para uso personal y una

versión para servidores conocida como Mac OS X Server.Este OS de sexta generación, mejoró muchísimo la interfaz de usuario. Básicamente sigue siendo Aqua pero con barras para scroll y el clásico gris platino renovado, ahora con mas azul. La GUI tiene ahora un toque mas 3D, con un dock con íconos interactivos y animados.

Figura 17: Interfaz de MacOS Leopard

Según Apple, Leopard contiene más de 300 cambios y mejoras sobre su predecesor, Mac OS X Tiger, cubriendo componentes del núcleo del sistema operativo, así como también las aplicaciones incluidas y las herramientas de desarrollo. Leopard presenta un escritorio notablemente mejorado, con un Dock rediseñado, Pilas (Stacks), una barra de menú semitransparente y un actualizado Finder que incorpora la interfaz de navegación visual Cover Flow, la cual fue vista por primera vez en iTunes. Entre otras características notables, están el soporte para escribir aplicaciones en 64-bit, una utilidad de manejo de copias de seguridad llamada Time Machine y el soporte de búsquedas de Spotlight en varias máquinas y la inclusión de Front Row y Photo Booth, las cuales antes eran incluidas sólo en algunos modelos de Macs.

GNOME 2.24 (2008)

Los chicos de GNOME pusieron bastante esfuerzo en crear temas y arte en la versión 2.24, ya que su meta es hacer “Que tu computadora luzca bien”. Organizaron una competencia

para encontrar los mejores fondos para escritorio que sus usuarios pudieran crear.

KDE v4.2 (Mar 2009)

La versión 4 de este desktop trajo muchas mejoras, como manejo de ventanas suave pero animado, y siguiendo la tendencia de Microsoft incluyeron Widgets (con relojes analógicos, por supuesto). El tamaño de los íconos es ajustable, y los elementos de diseño son mas fáciles de configurar al gusto. Nuevos sonidos, temas e íconos fotorealistas. Ahora también puede correr bajo plataformas Windows y Mac OS X.

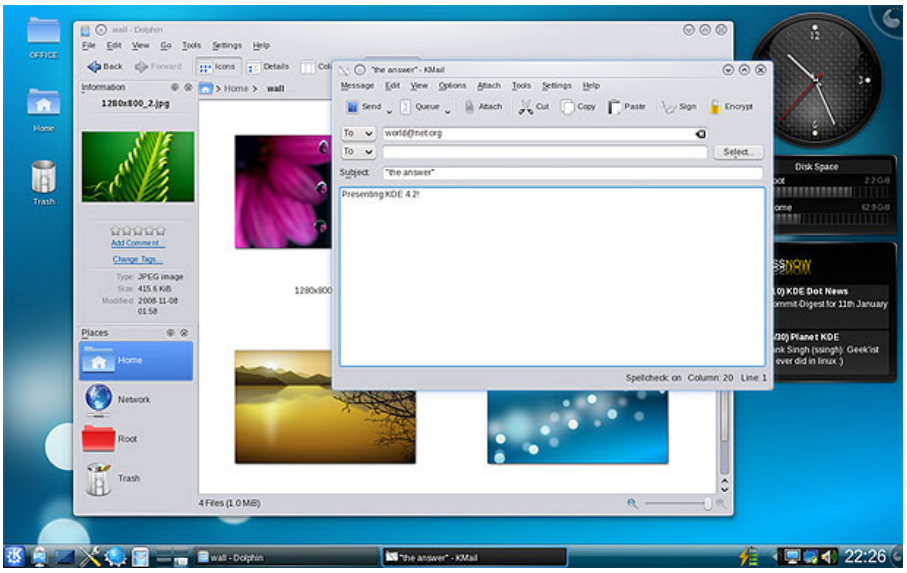


Figura 18: Interfaz de KDE 4.2

Interfaz Gráfica de los Sistema Operativos Móviles.

Un **sistema operativo móvil** o **SO móvil** es un sistema que controla un dispositivo móvil al igual que las PCs utilizan Windows, MacOS o Linux entre otros. Sin embargo, los sistemas operativos móviles son mucho más simples y están más orientados a la conectividad inalámbrica, los formatos multimedia para móviles y las diferentes maneras de introducir información en ellos.

A medida que los teléfonos móviles crecen en popularidad, los sistemas operativos con los que funcionan adquieren mayor importancia y obviamente también aquí surge una guerra no sólo por sel el mejor o el más popular, sino también por presentar una interfaz gráfica que gane más adeptos rápidamente y a la cual los usuarios accedan fácil

e intiuitivamente. De los sistemas móviles, Android ha alcanzado la máxima popularidad con un grueso de fieles seguidores de más o menos el 80% de usuarios. Le siguen en la lista:

- iOS 13,2%
- Windows Phone 3%
- BlackBerry OS 2,9%
- Linux u otros 0,8%
- Symbian OS 0,2%
- Firefox OS Disponible
- Ubuntu Touch que está en desarrollo

Interfaz de ANDROID

Android es un sistema operativo basado en el kernel de Linux, diseñado principalmente para dispositivos móviles con pantalla táctil, como teléfonos inteligentes o tabletas, inicialmente desarrollado por Android, Inc. Google respaldó económicamente y más tarde compró esta empresa en 2005. La estructura del sistema operativo Android se compone de aplicaciones que se ejecutan en un framework Java de aplicaciones orientadas a objetos sobre el núcleo de las bibliotecas de Java en una máquina virtual Dalvik con compilación en tiempo de ejecución.

Android, al contrario que otros sistemas operativos para dispositivos móviles como iOS o Windows Phone, se desarrolla de forma abierta y se puede acceder tanto al código fuente como a la lista de incidencias donde se pueden ver problemas aún no resueltos y reportar problemas nuevos.

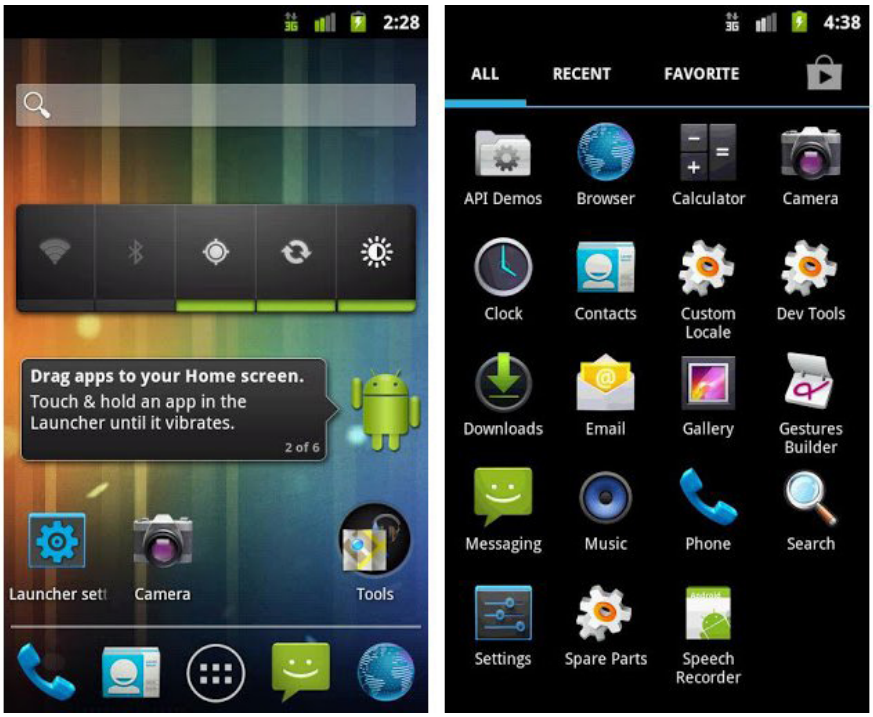


Figura 19: Interfaz de Android

El sistema Android es capaz de hacer funcionar a la vez varias aplicaciones y además se encarga de gestionarlas, dejarlas en modo suspensión si no se utilizan e incluso cerrarlas si llevan un periodo determinado de inactividad.

ANEXO: EVALUACIONES E INSTRUMENTOS

Test de usuario

El siguiente cuestionario tiene como objetivo evaluar el desempeño y eficiencia del sistema DECANO. Agradecemos tú participación, pues los datos que puedas proporcionarnos serán de gran utilidad para mejorarlo.
Te pedimos que leas cuidodosamente cada una de las preguntas antes de emitir las respuestas.

Cuestionario Aplicado

1. ¿Dispone el recurso de una presentación que contextualice y permita optimizar su uso (no en cuanto a comandos técnicos, sino en cuanto a aprovechamiento de los contenidos)?

2. ¿Es posible identificar al autor del recurso (persona, institución, etc.)?

3. ¿Dispone el recurso de un 'mapa del sistema'?

4. ¿Dispone de una herramienta de búsqueda para la localización de información en él contenida (tipo casilla para introducir texto libre)?

5. ¿Son pertinentes las cabeceras de cada ventana o recurso (por así decirlo, los 'titulares' que aparecen en la parte superior de cada ventana y que se pueden encontrar en él)?

6. ¿Resulta atractivo el aspecto general (colores, formas) de los elementos que componen el recurso (imágenes, fondo, texto, etc.)?

7. ¿Presenta elementos que impidan visualizar correctamente los contenidos?

9. ¿Son fácilmente legibles los textos?

10. ¿Dispone de foros o espacios para la de discusión?

11. ¿Facilita información referente a la fechas de elaboración de sus contenidos?

12. ¿Es necesaria la instalación de software adicional para su correcto uso?

13. ¿Resulta su diseño suficientemente claro para utilizarlo y localizar información en él?

14. ¿Tienen los iconos relación directamente con la función de la herramienta?

15. ¿Dispone de botones para deshacer o cancelar una función?

16. ¿Permite monitorear o dar seguimiento a tú trabajo de manera que puedas ver la secuencia de tus movimientos?

17. ¿Te parece clara la organización de los contenidos y herramientas?

18. ¿Te ofrece de forma interactiva diferentes posibilidades de navegación por sus contenidos en función del uso que hagas en cada momento?

19. ¿Responde el recurso adecuadamente a tu ritmo de navegación?

20. ¿Permite prevenir o corregir errores (por ejemplo, avisándote vas a salir del recurso, si has salvado o si se va a abrir una nueva ventana, etc.)?

21. ¿Han funcionado todos los botones o herramientas que has seleccionado?

22. ¿Permite realizar varias tareas al mismo tiempo (escribir, ver imágenes o vídeos y/u oir sonidos, etc.)?

23. ¿Presenta el recurso de forma simultánea información de distintos tipos y niveles de especificidad (texto, imágenes, video, iconos, etc.)?

24. En cuanto a tareas, ¿plantea el recurso actividades de diferente complejidad en función de los resultados obtenidos?

25. ¿Dispone el recurso de mecanismos para facilitar el control de tus acciones (evaluación de resultados, tutoración, etc.)?

26. En general, ¿resulta su manejo fácil e intuitivo (es decir, no requiere mucho esfuerzo y/o tiempo aprender a manejarlo)?

27. ¿Te resulta fácilo o difícil el contenido o tema del recurso?

28. ¿Disponías previamente de conocimientos sobre el tema del recurso?

29. ¿Te ofrece el recurso seguridad en la calidad de sus contenidos?

30. ¿Volverías a utiizar este recurso para aprender más?

31. ¿Recomendarías este recurso a otras personas interesadas en el tema que trata?

ANEXO: FORMATO DE EVALUACIÓN HEURÍSTICA

ANÁLISIS HEURÍSTICO

DATOS GENERALES

FECHA	HORA INICIO	HORA FINAL
ÁREA PROFESIONAL DEL EVALUADOR		
URL SISTEMA	http://www.sistemadecano.com	
PLATAFORMA		
NAVEGADOR/ES		

DINÁMICA

1. Ingresar al sistema y contestar el **Análisis General "A".**

2. Login: Ingresar como usuario registrado (respetar mayúsculas y acentos del usuario y contraseña) y contestar el **Análisis General "A".**

usuario: prueba **contraseña:** decano#1

3. Continuar como usuario registrado y contestar las siguientes áreas del Análisis heurístico.

Nota:
Si considera usted que su área profesional no le permite conocer la respuesta de alguna de las preguntas, favor de dejarla en blanco.

RÚBRICAS

VALOR	DESCRIPCIÓN
5	**Máxima** expresión heurística / Se cumple de forma satisfactoria.
4	**Alta** expresión heurística / Se cumple pero podría mejorar.
3	**Mediana** expresión heurística / Se cumple pero no en su totalidad.
2	**Baja** expresión heurística / Se cumple de manera incorrecta.
1	**Mínima** expresión heurística / Resulta confuso o difícil de localizar.
0	**No se cumple.**
NA	**No aplica** / No es necesario en el caso analizado.

PREGUNTA	VALOR ASIGNADO
¿Se comunica de forma clara e inmediata los objetivos o función del sistema a los usuarios?	
Los contenidos y servicios que se ofrecen ¿Corresponden a los objetivos o función del sistema?	
La URL (dirección del sistema) ¿Es concreta, clara y fácil de recordar?	
¿Existe información sobre las dinámicas que debe seguir un usuario nuevo para registrarse?	
El área destinada para el registro e ingreso de usuarios ¿Se ubica con facilidad?	
TOTAL =	

Análisis General "A"

PREGUNTA	VALOR ASIGNADO
¿El sistema se adapta al mundo de sus usuarios, sus lenguajes y sus conocimientos?	
¿Las herramientas facilitan y optimizan el acceso de los usuarios?	

¿El usuario recibe ayuda cuando la necesita?	
¿El diseño visual / gráfico son coherentes entre sí?	
¿La información que se muestra es necesaria y relevante?	
¿La estructura de la información presentada al usuario guarda una jerarquía?	
¿El usuario puede sentir que posee el control sobre lo que pasa en el sistema?	
TOTAL =	

Identidad e información

PREGUNTA	VALOR ASIGNADO
¿Se muestra claramente la identidad en cualquier lugar del sistema?	
El logotipo ¿Es significativo, identificable y suficientemente visible?	
¿Se proporcionan mecanismos para ponerse en contacto con la institución?	
¿Se proporciona información sobre la protección de datos de carácter personal de los usuarios o los derechos de autor de los contenidos del sistema?	
En los artículos, noticias, informes, etc. ¿Se muestra claramente información sobre el autor, fuentes, fecha de creación y actualización del documento?	
TOTAL =	

Lenguaje y redacción

PREGUNTA	VALOR ASIGNADO
¿El sistema habla el mismo lenguaje que sus usuarios?	
¿Emplea un lenguaje claro y conciso?	
¿La información que presenta es veraz?	
TOTAL =	

Ayuda

PREGUNTA	VALOR ASIGNADO
¿Es necesario destinar un área de ayuda?	
En caso de contar con una	
El enlace a la sección de ayuda ¿Está colocado en una zona visible?	
¿Se ofrece ayuda contextual en tareas complejas?	
Si diferencía el botón o ícono de ayuda tradicional, del botón o ícono para el tutor	
TOTAL =	

Retroalimentación y control

PREGUNTA	VALOR ASIGNADO
¿Se ha controlado el tiempo de respuesta?	
¿Se le informa al usuario lo que ha pasado? Por ejemplo : cuando un usuario se registra, se debe informar que se ha registrado correctamente	
Cuando se produce un error, ¿Se informa de fe forma clara y no alarmista al usuario de lo ocurrido y de cómo solucionar el problema?	
TOTAL =	

Accesibilidad

PREGUNTA	VALOR ASIGNADO
¿El tamaño de fuentes se ha definido de forma relativa, o por lo menos, las fuentes son lo suficientemente grandes como para no dificultar la legibilidad del texto?	
¿El tipo de fuente, efectos y tipografías facilitan la lectura?	
¿Existe un alto contraste favorable entre el color de fuentes y fondo?	
¿Puede el usuario disfrutar de todos los contenidos del sistema Web sin necesidad de tener que descargar e instalar plugins adicionales?	
TOTAL =	

Elementos multimedia

PREGUNTA	VALOR ASIGNADO
¿Las metáforas visuales son reconocibles y comprensibles por cualquier usuario?	
¿El uso de imágenes o animaciones proporciona algún tipo de valor añadido?	
¿La animación y reactividad del tutor facilita la interacción?	
El tutor animado presenta una animación que interfiere con el uso del sistema	
TOTAL=	

Estructura y navegación

PREGUNTA	VALOR ASIGNADO
La estructura de organización y navegación, ¿Es la más adecuada?	
¿Los enlaces son fácilmente reconocibles como tales?¿Su caracterización indica su estado (visitado, activo, etc.)?	
En menús de navegación, ¿Se ha controlado el número de elementos y de términos por elemento para no producir sobrecarga memorística?	

Los rótulos (nombres dados a botones enlaces) han sido asignados de forma coherente con su función y comportamiento?	
¿Es predecible la respuesta del sistema antes de hacer clic sobre el enlace?	
¿Se ha controlado que no haya enlaces que no lleven a ningún sistema?	
¿Existen elementos de navegación que orienten al usuario acerca de dónde está y cómo deshacer su navegación?	
Las imágenes enlace, ¿Se reconocen como 'clicables'? ¿Incluyen un atributo 'title' describiendo la página de destino?	
¿Se ha evitado la redundancia de enlaces?	
¿Se ha controlado que no haya páginas "huérfanas"?	
TOTAL =	

Layout de la página

PREGUNTA	VALOR ASIGNADO
¿Se aprovechan las zonas de alta jerarquía visual e informativa para ubicar los contenidos de mayor relevancia informativa para ubicar los contenidos de mayor relevancia?	
¿Se ha evitado la sobrecarga informativa?	
¿Es una interfaz limpia, sin ruido visual?	
¿Existen zonas en "blanco" entre los objetos informativos de la página para poder descansar la vista?	
¿Se hace un uso correcto del espacio visual de las páginas del sistema?	
¿Se utiliza correctamente la jerarquía visual para expresar las relaciones de pertenencia entre las páginas del sistema?	
¿Se ha controlado la longitud de la página?	
TOTAL =	

DINÁMICA

Sección evaluada	Sugerencia
Análisis Generales	
Identidad e información	
Lenguaje y redacción	
Búsqueda	
Ayuda	
Retroalimentación y Control	
Accesibilidad	

Elementos multimedia	
Estructura y navegación	
Layout de la página	
Opiniones: si considera que hay algún tema importante por observar que no halla sido referido en el Análisis heurístico, favor de abordarlo a continuación.	

4. De acuerdo a lo analizado anteriormente llenar la siguiente tabla.

Sugerencias del evaluador

LISTA DE REFERENCIAS

Lista de Referencias

Aaker, D.A. (1991): Managing Brand Equity Capitalizing on the Value of Brand Name, New York: The Free Press.

Acuña-Garduño E. (2008). *Análisis y diseño de entornos virtuales de aprendizaje colaborativo. Propuesta metodológica. Tesis de Maestría*. México: Universidad Autónoma Metropolitana. Unidad Azcapotzalco.

Acuña-Garduño E. (2012). Análisis y diseño de la interfaz para un sistema de aprendizaje colaborativo, Apoyada por un agente tutor inteligente. Tesis de Doctorado, México: Universidad Autónoma Metropolitana. Unidad Azcapotzalco.

Alem, L. y otros (1997) FILIP and intelligent simulator based framework for skill acquisition a case in air traffic control. En W. van Jooligen, A. Munro y L. Alem (eds.), Proceedings of AI-ED'97 workshop on Architecture for intelligent simulation based learning environment.

Almeida, S.; Febles, J. Y Bolaños, O. (1997) Evolución de la enseñanza asistida por computadoras, Revista Educ. Med. Sup. 11(1).

Asay-Davis, X. S. y otros (2000) Virtual Explorer: Interactive Virtual Environment for Education. Presence: Teleoperators and Virtual Environments, 9(6): pa´gs. 505–523.

Azpiazu, J., Pazos, J. Y Silva, A.(2001) La teleformación mediante Internet, Actas de "El futuro de Internet. Acceso y Teleservicios", Fundación Alfredo Brañas, España, 2001.

Azpiazu, J., Pazos, J. Y Silva, A.(2002) A virtual classroom based on academic memories, Proceedings of International Conference on Informa tio n and Communication Technologies in Education (ICTE2002), España, 2002.

Babacci, M. (2003). Quality Attribute Workshops (QAWs), 3ra. Edición, Carnegie Mello University, Software Engineering Institute.

Ball, G. y otros (1997) Lifelike Computer Characters: the Persona Project at Microsoft. En J. Bradshaw (ed.), Software Agents. AAAI/MIT Press.

Baranauskas, M. C. C. y Borges, M. A. F. (1997) Jonas - The Jonah agent in a computer-based learning system. En Proceedings of the International Conference: Information Technology for Competitiveness - Experiences and Demands for Education and Vocational Training.

Baranauskas, M. C. C. y otros (1997) Learning by creating models: a computer-based environment for industrial application. En Proceedings of the 8th Conference on Artificial Intelligence in Education (AIED97), pa´gs. 426–433.

Barker, P. Y Manji, K. (1991): Designing Electronic Books. *Educational and Learning Technology International (ETTI*). 28 (4).

Barros, B. and Verdejo, M. F. (2001). Entornos para la realización de actividades de aprendizaje colaborativo a distancia. *Revista Iberoamericana De Inteligencia Artificial*. Vol. 3 (12), pp.39-49.

Barrows, H. S. and Tamblyn, R. M. (1980). *Problem-Based Learning: An Approach to Medical Education.* N.Y.: Springer Publishing Company.

Bass, L. y otros (2001) Quality Attribute Design Primitives and the Attribute Driven Design Method. En F. van der Linden (ed.), PFE '01: Revised Papers from the 4th International Workshop on Software Product-

Family Engineering, tomo 2290 de Lecture Notes in Computer Science, pa´gs. 169 – 186. ACM, Springer-Verlag.

Bass, L. y otros (2003) Software Architecture in Practice. Addison-Wesley, Boston, MA, second edition ed.

Baylor, A. L. (2001) Investigating multiple pedagogical perspectives through MIMIC (Multiple intelligent mentors instructing collaboratively). En Proceedings of Artificial Intelligence in Education (AI-ED) International Conference.

Beaird, J. (2007). *The principles of beautiful web design.* Australia: SitePoint.

Bloom, B. S., Engelhart, M. D., Murst, E. J., Hill, W. H. and Drathwohl, D. R. (1956). *Taxonomy of Educational Objetives: The Cognitive Domain.* Longmans.

Bloom, B. S., Hastings, J. TH. Y Madaus, F. (1971). Handbook on formative and summative evaluation of student learning. New York: McGraw-Hill.

Bloom, B.S., Hastings, T. y Madaus G. (1975). Evaluación del aprendizaje. Buenos Aires: Troquel.

Bonsiepe, G. (1998). Del objeto a la interface: Mutaciones del diseño. Buenos Aires: Ediciones Infinito Buenos Aires.

Borges, M. A. F. y Baranauskas, M. C. C. (1997) A User-centered Approach to the Design of an Expert System for Training. En Proceedings of the 8th International PEG Conference, p´ags. 74–81.

Boucké, N. y Holvoet, T. (2008) View composition in multiagent architec- tures. International Journal of Agent-Oriented Software Engineering, 2(1): p´ags. 3–33.

Boucké, N. y otros (2006) Applying the ATAM to an Architecture for Decentralized Control of a Transportation System. En Second International Conference on Quality of Software Architectures, tomo 4214 de Lecture Notes in Computer Science, pa´gs. 180–198. Springer- Verlag.

Bricken, W. (1990a) Learning in Virtual Reality. Inf. t´ec., University of Washington, Human Interface Technology Laboratory.

Bricken, W. (1990b) Training in Virtual Reality. Inf. t´ec., University of Washington, Human Interface Technology Laboratory.

Brown, J. S. and Burton, R. (1978). Diagnostic models for procedural bugs in basic mathematical skills. *Cognitive Science*. Vol. 2, pp.155-192.

Brown, J. S., Burton, R. and Bell, A. G. (1975). SOPHIE. A step towards a reactive learning environment. *International Journal of Man-Machine Studies*. Vol. 7, pp.675-696.

Brusilovsky, P. and Nijhavan, H. (2002) A Framework for Adaptive E-Learning Based on Distributed Re-usable Learning Activities. In: Druscoll, M. and Reeves, T. C., Proceedings of World Conference on E-Learning

Brusilovsky, P., Schwarz, E. and Weber, G. (1996) ELM-ART: An Intelligent Tutoring System on World Wide Web, Intelligent Tutoring System Springer VerlagLecture Notes in Computer Science.

Buitrón, M. (2003): "Consideraciones para el diseño de interfaces gráficas de usuario en ambientes virtuales educativos". Tesis para optar por el grado de Maestra en Diseño en la Línea de Nuevas Tecnologías, Universidad Autónoma Metropolitana, Azcapotzalco, México.

Buche, C. y otros (2004) MASCARET: A Pedagogical Multi-Agent System for Virtual Environments for Training. International Journal of Distance Education Technologies, 2(4): pa´gs. 41–61.

Burton, R. R.; Brown, J. S. (1981). *An investigation of computer coaching for informal learning activities*. In: Sleeman, D., Brown, J. (eds.): ITS, Cap. 4, 79-98, London: Ac. Press.

Burton, R. (1982). Diagnosing bugs in a simple procedural skill, in D.H. Sleeman et. al., Intelligent Tutoring Systems, Academic Press, London.

Buschmann, F. et al. (1996) Pattern-Oriented Soaftware Architecture: A System of Patterns, tomo 1 de Software Design Patterns. John Wiley & Sons.

Byrne, C. (1993) Virtual Reality and Education. En Proceedings of IFIP WG3.5 International Workshop Conference.

Cabero J. (2003), Principios tutor, psicológicos y sociológicos del trabajo colaborativo: su proyección en la tele enseñanza, en Martínez Sánchez, F. (comp.) (2003). Redes de comunicación en la enseñanza, (pp.131 – 156). 1a edición. Barcelona: Paidós.

Cabero, J. (editor) (2000), Nuevas Tecnologías aplicadas a la educación. 2a edición, Madrid: Didáctica y organización Escolar-Síntesis Educación.

Cañas Delgado, Jose Juan, El diseño de su interacción desde la ergonomía cognitiva, Ediciones Pirámide, 2004. Canut, M. F., Gouardères, G. and Sanchis, E. (1999). The Systemion: A New Agent Model to Design Intelligent Tutoring System. In *Artificial Intelligence in Education.* Frontiers in Artificial Intelligence and Applications . v. 50) pp. 54-63.: IOS Press.

Capuano, N. y otros (2000) ABITS: An Agent Based Intelligent Tutoring System for Distance Learning. En Proceedings of the International Workshop on Adaptive and Intelligent Web-Based Education Systems. ITS 2000.

Carbonell, J. R. (1970). *AI in CAI: An artificial intelligence approach to computer assisted instruction*. IEEE transaction on Man Machine System. Vol.11, No. 4, p. 190-202.

Carro, Rosa. (2001) Entornos de enseñanza adaptativa, Tesis Doctoral, Universidad Disponible en: www.ii.uam.es/~rcarro/r_invest_s.html

Cassell, J. (2000) Nudge Nudge Wink Wink: Elements of Face-to-Face Conversation for Embodied Conversational Agents. En J. Cassell, J. Sullivan, S. Prevost y E. Churchill (eds.), Embodied Conversational Agents, p´ags. 1–27. MIT Press, Cambridge, MA.

Cassell, J. y otros (eds.) (2000) Embodied Conversational Agents. MIT Press.

Castañeda, S., Martínez, R. (1999). Enseñanza y aprendizaje estratégicos: Modelo integral de evaluación e instrucción. En *Revista Latina de Pensamiento y Lenguaje*. 4:2B, 251-278.

Cataldi. Z. (2004) Metodología para el diseño de sistemas tutores Inteligentes. Tesis doctoral. Facultad de Informática. UNLP.

Cataldi, Zulma; Lage, Fernando J. (2009) Sistemas tutores inteligentes orientados a la enseñanza para la comprensión (artículo en línea). EDUTEC, Revista Electrónica de Tecnología Educativa. Núm. 28/ Marzo 2009. (Fecha de consulta: 23/sep/2011).

Cataldi, Z.: Lage, Fernando; Salgueiro, F.(2005). Sistemas Tutores Inteligentes. Los Estilos del Estudiante

para Selección del Tutorizado. Proceedings del VII Workshopo de Investigadores en Ciencias de la Computación. WICC 2005. Pp 66-70.

Cataldi, Z. Y Lage, F. (2007). El problema del modelado del estudiante en Sistemas Tutores Inteligentes. II Congreso de Tecnología en Educación y Educación en Tecnología. TE&ET'07. 12-15de junio. Facultad de Informática, Universidad Nacional de La Plata.

Cataldi, Z., Salgueiro, F., Britos, P., Sierra, E. y García Martínez, R. (2006). *Selecting Pedagogical Protocols using SOM*. Research in Computing Science Journal, 21: 205-214.

Cataldi, Z; Salgueiro, F. y Lage, F. (2007b). Fundamentos para el Submódulo Evaluador en Sistemas Tutores Inteligentes: Diagnóstico, predicción y autoevaluación. CACIC 2007. 1-5 de octubre. Universidad Nacional del Nordeste Facultad de Ciencias Exactas y Naturales y Agrimensura. Corrientes y Universidad Tecnológica Nacional, Facultad Regional Resistencia.

Chan, T. W. (1996) Learning companion systems, social learning systems, and the global social learning club. Journal of Artificial Intelligence in Education, 7: pa´gs. 125–159.

Chan, T. W. y Baskin, A. B. (1990) Learning companion systems. En C. Frasson y G. Gauthier (eds.), Intelligent tutoring systems: At the crossroads of artificial intelligence in education.

Chang, D., Dooley L., and Tuovinen E. J., 2001. "Gestalt Theory in Visual Screen Design–A New Look at an Old Subject", the Seventh World Conference on Computers in Education, Copenhagen, July 29-August 3.

Choi, S. y Clark, R. E. (2006) Cognitive and affective benefits of animated pedagogical agents in learning English as a second language. Journal of Educational Computing Research, 33(2): pa´gs. 455–480.

Choua, C., Chanb T. & Linc C. (2002). *Redefining the learning companion: the past, present, future of educational agents, Computer & Education,* 40(3), 255- 269.

Chou, C. y otros (2003) Redefining the learning companion: the past, present, and future of educational agents. Computers & Education, 40(3): p´ags. 255–269.

Clancey, W. J. (1987). *Transfer of rule-based expertise through a tutorial dialogue.* PhD thesis, Stanford University.

Clements, P. y otros (2002a) Documenting Software Architectures: Views and Beyond. The SEI Series in Software Engineering. Addison Wesley Professional, 1 ed on.

Clements, P. y otros (2002b) Evaluating Software Architectures: Methods and Case Studies. The SEI Series in Software Engineering. Addison Wesley Professional, 1 ed on.

Clements, P. C. (2000) Active Reviews for Intermediate Designs. Inf. T´ec. CMU/SEI-2000-TN-009, Software Engineering Institute - Carnegie Mellon University, Pittsburg, PA, USA.

Costa, Joan (2011). Los tres fundamentos del lenguaje gráfico, Revista electrónica I+Diseño, Vol. 4, Año IV, disponible en http://www.diseno.uma.es/i_diseno/i_diseno_4/articulos.htm.

Costa, G.; Salgueiro, F. A., Cataldi, Z., García-Martínez, R. Y Lage, (2005) F. J. *Sistemas inteligentes para el modelado del estudiante* Proc. GCETE'2005 CD**,** Global Congress on Engineering and Technology Education.Brasil. marzo 13-15 2005.

Cugini, J.; Laskowski, S.; Sebrechts, M. (2000). "Design of 3-D visualization of search results: evolution

and evaluation". Proceedings of IST/SPIE's 12th Annual International Symposium: Electronic Imaging 2000: Visual Data Exploration and Analysis (SPIE 2000) , Disponible en http://www.itl.nist.gov/iaui/vvrg/cugini/uicd/nirve-paper.html

Del Moral, M. E y otros. (2001): Diseño de interfaces de navegación y pruebas de usabilidad en aplicaciones interactivas. En las III Jornadas Multimedia Educativa. Barcelona, ICE Universidad de Barcelona.

Díaz,P. Catenazzi, N. & Aedo, I. (1996). De la multimedia a la hipermedia, Editorial Rama:Madrid,

Dondis, D. (2000). La sintaxis de la imagen: introducción al alfabeto visual. México: Gustavo Gili.

Dix, A., Finlay, J., Abowd, G., Beale, R. (1998). *Human-Computer Interaction*. Prentice-Hall.

Dorin, H., Demmin, P. E., Gabel, D. (1990). *Chemistry: The study of matter.* (3rd ed.). Englewood Cliffs, NJ: Prentice Hall, Inc.

Drettakis, G. y otros (2007) Design and Evaluation of a Real-World Virtual Environment for Architecture and Urban Planning. Presence: Teleoperators and Virtual Environments, 16(3): pa´gs. 318–332.

Elliott, C. D. y otros (1999) Lifelike Pedagogical Agents and Affective Computing: An Exploratory Synthesis. Artificial Intelligence Today, pa´gs. 195–212.

Etzioni, O. (1996) Moving Up the Information Food Chain: Deploying Soft- bots on the World Wide Web. En T. A. P. . T. M. Press (ed.), Pro- ceedings of the Thirteenth National Conference on Artificial Intelligence and the Eighth Innovative Applications of Artificial Intelligence Confe- rence, p´ags. 1322–1326.

Falgueras, J. (2000). Usabilidad informática; Diseño actual de la interfaz de usuario. Tesis doctoral. Departamento de Lenguajes y Ciencias de la Computación. Universidad de Málaga.

Felder, R. M. and Silverman, L. K. (1988). Learning and Teaching Styles. *Engineering Education*. Vol. 78 (7), pp.674-681.

Felder, Richard (1988) Learning and Teaching Styles in Engine ering Education, Journal of Engineering Education.

Felder, Richard (1996)Matters of sytle. ASSE Prism, 6(4), Diciembre 1996.

Felder, Richard (2003) Learning and teaching styles in engineering education. Author's preface. Disponible en: www.ncsu.edu/felder-public/Papers/LS-1988.pdf.

Férey, N. y otros (2004) A Distributed Multimedia Database Visualization within an Immersive Environment for Bioinformatics. En IEEE Sixth International Symposium on Multimedia Software Engineering.

Fernández – Guinea, S. (2001). Estrategias a seguir en el diseño de los programas de rehabilitación neuropsicológica para personas con daño cerebral. Revista de Neurología, 33, 4, 373 – 377.

Ferreira, Anita y Kotz, Gabriela. ELE-Tutor Inteligente: Un analizador computacional para el tratamiento de errores gramaticales en Español como Lengua Extranjera. Rev. Signos. (En línea). 2010, vol.43, n.73, pp. 211-236. ISSN 0718-0934. doi: 10.4067/S0718-09342010000200002.

Finin, T., Fritzon, R., McKay, D. & McEntire, R. (1993). KMQL -A Language and Protocol for Knowledge and Information Exchange, Tech. Report, University of Maryland, Baltimore, Maryland.

Fló, Juan. (2010) Imagen, Ícono e Ilusión. Investigación sobre algunos problemas de la representación visual. Siglo XXI: México.

Follows, Scott B.(1999) Virtual learning environments, THE Journal, Nov 99, Vol 27, Issue 4, p. 100.

Freund, J. E y Walpone, R.E. (1990). Estadística matemática con aplicaciones. Cuarta edición, México: Prentice Hall Hispanoamericana, S.A.

Furness, T. A. (1986) The super cockpit and its human factors challenges. En Proceedings of the Human Factors Society, tomo 30, pa´gs. 48–52.

Gagné, R. M. and Briggs, I. J. (1979). *Principles of instructional design*. Holt, Rinehart and Wiston.

Galvin, T. (1994). Tesis Doctoral: *Mebuilder: An Object-Oriented Lesson Authoring System for Procedural Skills* Master's Thesis, Naval Postgraduate School. Monterey.

Gamma, E. y otros (1995) Design Patterns. Elements of Reusable Object-Oriented Software. Addissoon-Wesley.

Gandara, M. (2004). Notas y apuntes del *Diplomado de software educativo y de capacitación:* Evaluación y diseño". Ed. Instituto Latinoamericano de la Comunicación Educativa, México.

Gandara, M. (2004). Notas y apuntes del curso *Introducción a los multimedios en la educación.* Ed. Centro de Entrenamiento de Televisión Educativa, México.

Gardner, H. (1983). *Frames of mind: the theory of multiple intelligences.* New York: Basic Books.

Gardner, H. (1993). Inteligencias Múltiples: La teoría en la práctica. Paidós. Barcelona, Buenos Aires, México.

Gardner, H. (2000) La educación de la mente y el conocimiento de las disciplinas, Paidós: Barcelona.

Gardner, H. (2001) La inteligencia reformulada: las inteligencias múltiples en el siglo XXI. Barcelona. Paidós.

Garrido, J.L., Gea, M., Rodriguez, M.L. (2005). *Requirements enginnering in cooperative systems, in: Requirements Enginnering for Sociotechnical Systems,* Idea Group, Inc., USA, 2005 (Chapter XIV).

Gary, G. & Mazur, J. (1991): "Navigating Hypermedia". Eds. Berk, E. & Devlin, J. Hypertext/Hypermedia Handbook. Intertext Publications, New York: McGraw-Hill.

Gherbi, R. y H´erisson, J. (2001) 3d Modelling tools for spatial-based in silico analysis of DNA. In International Electronic Journal on Computer Graphics and Geometry.

Gigante, M. (1993) Virtual Reality: Definitions, History and Applications. En Virtual Reality Systems, pags. 3–14. Academic-Press.

Gil, José.(2002) Ideas para un modelo de web docente. Recuperado el 11 de abril de 2010 en http://www.unizar.es/ice/web- docente/Modelo%20de%20web%20docente.htm.

Giraffa, L.M.M.; Nunes, M. A.; Viccari, R.M. (1997) *Multi-Ecological: an Learning Environment using Multi-Agent architecture.* Proc. MASTA'97: Coimbra: DE- Universidad de Coimbra.

Gómez, A. Juristo, N. Montes, C. Y Pazos, J.(1997) Ingeniería del Conocimiento, España: Editorial Centro de Estudios Ramón Areces S.A.

Gómez-Pérez, A. and Corcho, O. (2002). Ontology Languages for the Semantic Web. *IEEE Intelligent Systems*. Vol. 17 (1), pp.54-60.

Graesser, A.C., Chipman, P., Haynes, B.C. y Olney, A. (2005). *AutoTutor: An intelligent tutoring system with mixed-initiative dialogue.* IEEE Transactions in Education, 48, 612-618.

Graesser, A. C. y otros (2005b) AutoTutor: An intelligent tutoring system with mixed-initiative dialogue. IEEE Transactions in Education, 48: pa´gs. 612–618.

Graesser, A. C. y otros (2008) The relationship between affect states and dialogue patterns during interactions with AutoTutor. Journal of Interactive Learning Research, 19: pa´gs. 293–312.

Guiudici E., R., & Bris Lluch, Á. (1997). *Introduccion a la teoria de Grafos.* Venezuela: Impresos Gráficas Tao.

Guzmán, E. (2005). *Un modelo de evaluación cognitiva basado en tests adaptativos informatizados para el diagnóstico en Sistemas Tutores Inteligentes*. Universidad de Málaga.

Ha, T. y Woo, W. (2006) Garden Alive: an Emotionally Intelligent Interactive Garden. The International Journal of Virtual Reality, 5(4): pa´gs. 21–30.

Halff, H. M. (1988). Curriculum and Instruction in Automated Tutors. In Polson, M. C., Richardson, J.,*Foundations of Intelligent Tutoring Sytems*. Hillsdale, New Jersey: Lawrence Erlbaum.

Hassan Montero, Y. (30 de Marzo de 2003). *Guía de Evaluación Heurística de Sitios Web*. Obtenido de no solo usabilidad: http://www.nosolousabilidad.com/articulos/heuristica.htm

Hassan Montero, Y.; Martín Fernández, F.J. (2003). Guía de Evaluación Heurística de Sitios Web En: No Solo Usabilidad, nº 4, 30 de Marzo de 2003.

Hassan Montero, Y.; Martín Fernández, F.J. (2005). La Experiencia del Usuario. En: No Solo Usabilidad, nº 4, Septiembre 2005.

Hayes-Roth, R. (1995) An Architecture for Adaptive Intelligent Systems. Artificial Intelligence: Special Issue on Agents and Interactivity, 72: pa´gs. 329–365.

Heng, P. A. y otros (2006) A Haptic Needle Manipulation Simulator for Chinese Acupuncture Learning and Training. International Journal of Image and Graphics, 6(2): pa´gs. 205–230.

Herrera B., Miguel Á.(2012). Las fuentes del aprendizaje en ambientes virtuales educativos, Revista Iberoamericana de Educación, ISSN: 1681-5653, sep. Versión disponible en PDF en http://www.campusoei.org/revista/index/frame_novedades.htm , recuperado en marzo de 2012.

Herrero, P. y otros (2005) Modelling the Sensory Abilities of Intelligent Virtual Agents. Autonomous Agents and Multi-Agent Systems, 11(3): pa´gs. 361–385.

Hirose, M. (2006) Virtual Reality Technology and Museum Exhibit. The International Journal of Virtual Reality, 5(2): pa´gs. 31–36.

Hofmeister, C. et al. (2000) Applied Software Architecture. Addison Wesley.

Hospers, M. et al. (2004) An Agent-based Intelligent Tutoring System for Nurse Education, cap. 9, pa´gs. 143–159. Whitestein Series in Software Agent Technologies. Birkhauser Publishing Ltd.

Hosseini, M. y otros (2002) A haptic virtual environment for industrial training. IEEE International Workshop on Haptic Virtual Environments and Their Applications (HAVE), p´ags. 25–30.

Hughes, C. E. y Moshell, J. M. (1997) Shared Virtual Worlds for Education: the ExploreNet Experiment.

Multimedia Systems, 5(2): pa´gs. 145–154.

IMS Global Learning Consortium, Inc. (2005 Jan). Learner Information Package Specification [Web Page]. Disponible en: http://www.imsglobal.org/profiles/.

ISO/IEC 11581 (2000) Information technology — User system interfaces and symbols — Icon symbols and functions. Disponible en: http://web.zpr.fer.hr/ergonomija/2002/sipka/ISO-1.pdf

ISO 11064-1 (2000) Diseño ergonómico de los centros de control.

ISO 9241 (1996) Requisitos ergonómicos para trabajos de oficina con pantallas de visualización de datos (PVD)

James Garret, J. (6 de marzo de 2002). *Un vocabulario visual para describir arquitectura de información y diseño de interacción*. Disponible en: jjg.net: http://www.jjg.net/ia/visvocab/spanish.html.

James Garret, J. (2009) The Elements of Users Experience, User Centered Design for the Web, Aiga, New Riders.

Jiménez J. (2001) "Un Modelo de Integración de Sistemas Tutoriales Inteligentes y Ambientes Colaborativos de Aprendizaje bajo el esquema de Universidad Virtual" Tesis de Maestría en Ingeniería de Sistemas. Escuela de Sistemas, Universidad Nacional de Colombia sede Medellín, Colombia.

Johnson, A. y otros (1998a) The NICE Project: Learning Together in a Virtual World. En Proceedings of VRAIS'98, p´ags. 176–183. IEEE.

Johnson, A. y otros (1999a) Learning and Building Together in an Immersive Virtual World. Presence: Teleoperators and Virtual Environments, 8(3): pa´gs. 247–263. Special Issue on Virtual Environments and Learning.

Johnson, A. y otros (1999b) The Round Earth Project: Deep Learning in a Collaborative Virtual World. En Proceedings of Virtual Reality, pa´gs. 164–171. IEEE.

Johnson, W. L. y Valente, A. (2008) Tactical Language and Culture Training Systems: Using Artificial Intelligence to Teach Foreign Languages and Cultures. En Proceedings of IAAI 2008.

Jung, C. G. (1994). *Tipos psicológicos*. Barcelona: Edhasa.

Kare, S. (01 de Marzo de 2011). El nacimiento del diseño gráfico para la interfaz digital. Experimenta Magazine. (A. Vit, Entrevistador) España.

Karpich Zardalevich, A. (n.d.). Disponible en: www.catfish-project.com.ar.

Kazman, R. y otros (2000) ATAM: Method for Architecture Evaluation. Inf. T´ec. CMU/SEI-2000-TR-004, Software Engineering Institute - Carnegie Mellon University, Pittsburg, PA, USA.

Kazman, R. y otros (2002) Making Architecture Design Decisions: An Economic Approach. Inf. T´ec. CMU/SEI-2002-TR-035, Software Engineering Institute - Carnegie Mellon University, Pittsburg, PA, USA.

Kerlinger, F. (1983). *Investigación del Comportamiento. Técnicas y Metodología* (2a Edición). México: Editorial Interamericana.

Kerlinger, F. (1994). Investigación del comportamiento: Técnicas y metodología. México: Interamericana.

Kirner, T. G. y otros (2001) Development of a Collaborative Virtual Environment for Educational Applications.

En Proceedings of the 2001 Conference on 3D Technologies for the World Wide Web, pa´gs. 61–68.

Klein, M. y Kazman, R. (1999) Attribute-Based Architectural Styles. Inf. T´ec. CMU/SEI-99-TR-022, Software Engineering Institute - Carnegie Mellon University, Pittsburg, PA, USA.

Klein, M. H. y otros (1999) Attribute-Based Architecture Styles. En Software Architecture, Proceedings of the First Working IFIP Conference on Software Architecture (WICSA1), pa´gs. 225–243. Kluwer Academic Publishers, San Antonio, Texas, USA.

Kolb, D. A. (1984). *Experiential Learning: Experience as the Source of Learning and Development*. N.J.: Prentice-Hall.

Korze, D. y otros (2001) Virtual delivery room as a teaching tool. En Proceedings of the 9th Mediterranean Conference on Medical and Biological Engineering and Computing (MEDICON 2001), tomo 1, p´ags. 476–79. Pula, Croacia.

Kroll, P. y Kruchten, P. (2003) The Rational Unified Process Made Easy. A Practitioner's Guide to the RUP. Object Technology. Addison Wesley.

Kuljis, J. (2002) A distributed virtual collaborative learning environment. En Proceedings of AI 2002. 20th IASTED International Multiconference on Applied Informatics, p´ags. 464–69. Innsbruck, Austria.

Krug, S. (2001). *No me hagas pensar: Una aproximación a la usabilidad en la Web* (2 ed.). Berkeley, California: New Riders.

Kruchten, P.(2004). An ontology of architectural design decisions in software intensive systems, In 2nd Groningen Workshop on Software Variability, pp. 54–61

Labrou,Y. y Finin, T. (1997) A Proposal for a new KQML Specification. Reporte Técnico, Universidad de Maryland, Baltimore.

Larijani, L. C. (1994) The Virtual Reality Primer. McGraw Hill.

Laureano-Cruces, A., de Arriaga-Gómez, F. (1998). Multi-Agent Architecture for Intelligent Tutoring Systems. *Interactive Learning Environments*, Vol. 6, No. 3, pp. 225–250.

Laureano-Cruces, A., F. de Arriaga. *Reactive Agent Design for Intelligent Tutoring* Systems. En *Cybernetics and Systems (an International Journal)*. Vol. 31, pp. 1- 47. ISSN: 0196-9722. (Ed). TAYLOR & FRANCIS.

Laureano-Cruces, A., de Arriaga, F., García-Alegre, M. (2001). *Cognitive task analisys: a proposal to model reactive behaviours*. En Journal of Experimental & Theoretical Artificial Intelliegence.

Laureano-Cruces, A., Terán-Gilmore, A., de Arriaga, F, El Alami M. (2003).La Importancia de las Estrategias Cognitivas en el Diseño del Currícula Didáctico. Vol. I, pp. 35 – 41. En el *XVI Congreso Nacional y II Congreso Internacional de Informática y Computación de la ANIEI*. Zacatecas, 22-24 de octubre del 2003.

Laureano-Cruces, A., Ramírez-Rodríguez, J., Terán-Gilmore, (2004). *A. Evaluation of theTeaching-Learning Process With Fuzzy Cognitive Maps*. En Memorias de Ibero-American Conference on Artificial Intelligence. IBERAMIA 2004.

Laureano-Cruces, A.L., Ramírez-Rodríguez, J., Mora-Torres, M., de Arriaga, F., Escarela-Pérez, R. (2010). *Cognitive-Operative Model of Intelligent Learning Systems Behavior.* En la Revista *Interactive Learning Environments*. Routledge. ISSN: 1049-4820. UK. Vol. 18, no.1, pp. 11-38.

Laureano-Cruces, A., Mora-Torres, M., Ramírez-Rodríguez, J., Gamboa- Rodríguez, F. (2010). *Imple-*

mentation of an affective-motivational architecture tied to a teaching-learning process. En Proceedings de E-Learn 2010 World Conference on E-Learning in Corporate Govermment, Healthcare, & Higher Education, pp. 1930-1938. ISBN: 1-880094-53-5. Orlando, Florida, October 18- 22.

Lester, J. y Stone, B. (1997) Increasing believability in animated pedagogical agents. En Proceedings of the First International Conference on Autonomous Agents, p´ags. 16–21.

Lester, J. y otros (1997a) Animated pedagogical agents and problem solving efectiveness: A large-scale empirical evaluation. En Proceedings of Eighth World Conference on Artificial Intelligence in Education, p´ags. 23–30.

Lester, J. y otros (1997b) Cosmo: A life-like animated pedagogical agent with deictic believability. En IJCAI97 Workshop on Animated Interface Agents: Making them Intelligent. Nagoya, Japan.

Lester, J. y otros (1997c) The persona effect: Affective impact of animated pedagogical agents. En Proceedings of CHI 97, p´ags. 359–366. ACM.

Liaw, Shu-Sheng (2001) Designing the hypermedia -based learning environment, International Journal of Instructional Media. Volume 28, Issue 1, p. 43.

Loftin, R. B. y otros (2004) Training in Peacekeeping Operations Using Virtual Environments. IEEE Computer Graphics and Applications, 24(4): p´ags. 18–21.

LOM . Learning Object Metadata [Web Page]. Accessed 2004 Mar. Disponible en: *ltsc.ieee.org/doc/wg12/LOMv4.1.htm*.

López, Antonia (2000) Herramientas de desarrollo de software educativo, España: Universidad Politécnica de Madrid.

Lozano, A. y otros (2003) Virtool-D: Entorno virtual educativo para aprendizaje en dominios procedimentales. En Proceeings of SIIE 2003, International Symposium in Education Computing.

LTSC . Learning Technology Standards Committee [Web Page]. Accessed 2004 Apr. Disponible en: http://ltsc.ieee.org/.

Lucas, S. y otros (2008) Virtual Reality Training Improves Simulated Laparoscopic Surgery Performance in Laparoscopy Na¨ive Medical Students. Journal of Endourology, 22(5): pa´gs. 1047–1052.

Lynch, P. J., & Horton, S. (2000). *Principios de diseño Básicos para la creación de sitios web* (2ª ed.). Naulcalpan: Ediciones G. Gili SA de CV.

Manzini & Ezio (1996) Artefactos: Hacia una nueva ecología del ambiente artificial, Celeste Ediciones, S.A.

Marqués, P. (1998) Clasificación de software educativo. Recuperado el 15 de enero de 2011 en http://www.xtec.es/~pmarques/edusoft.htm.

Marrero, Carlos (2006) Interfaz Gráfica de Usuario, Aproximación semiótica y cognitiva, Informe de Proyecto de Investigación, Universidad de la Laguna, Tenerife.

Mason, R. y Lind, D. (1998). Estadística para Administración y Economía. Alfa Omega.

Matsuda, N., & Vanlehn, K. (2005). *Advanced Geometry Tutor: An intelligent tutor that teaches proof-writing with construction*. In Proc. of The 12th International Conference on Artificial Intelligence in Education.

Marsella, S. y Johnson, W. L. (1997) An instructor's assistant for team- teaching in dynamic multi-agent virtual worlds. En Proceedings of the Fourth International Conference on Intelligent Tutoring Systems, LNCS. Springer-Verlag.

Mason, R. & Lind, D. (1998). Estadística para administración y economía, México:Alfaomega Grupo Editor, S.A. de C.V.

McNeil, P. (2010). *The web designer's idea book* (Vol. 2). Cincinnati, Ohio: How Books. Miniwatts Marketing Group (2012). *Internet World Stats.* Disponible en: http://www. internetworldstats.com/stats.htm.

Medinilla, N. y Gutiérrez, I. (2007) La Incertidumbre como Herramienta en la Ingenier´ıa de Software. IEEE Latin America Transactions, 5(4): pa´gs. 265–270.

Méndez, G. y otros (2001) PRVIR: An Integration Between an Intelligent Tutoring System and a Virtual Environment. En SCI2001, tomo VIII, pa´gs. 175–180. IIIS, IEEE Computer Society, Orlando, FL.

Méndez, P. (2008) Una Arquitectura Software Basada en Agentes y Recomendaciones Metodolo´gicas para el Desarrollo de Entornos Virtuales de Entrenamiento con Tutor´ıa Inteligente, Tesis Doctoral, Universidad Politécnica de Madrid.

Mendehall W., y otros (1994) Estadística Matemática con Aplicaciones. México: Grupo Editorial Iberoamérica.

Meneses, B.(1997) La Aplicación de la Estadística no paramétrica en la Administración, Disponible en http://www.uv.mx/iiesca/files/2013/01/estadistica1997.pdf.

Montoro Manrique, G. (2005) Estudio e Integración de un Sistema de Diálogos Dinámico en un Entorno Inteligente, Tesis Doctoral, Universidad Autónoma de Madrid.

Moreno, R. y otros (2001) The case for social agency in computer-based teaching. Do students learn more deeply when they interact with animated pedagogical agents? Cognition and Instruction, 19(2): pa´gs. 177–213.

Müller-Brockmann, J. (1982). *Sistemas de Reticulas.* Barcelona, España: Editorial Gustavo Gili. S.A.

Muller-Wittig, W. K. y otros (2001) LAHYSTOTRAIN - VR-based Inte- lligent Training Environment for Laparoscopy and Hysteroscopy. En S. Richir (ed.), Virtual Reality International Conference (VRIC) 2001. Proceedings, p´ags. 225–233.

Negroponte, N (1995) *El Mundo digital*. Barcelona: Ediciones Blume ISBN:234562.

Nielsen, J. (2002). *Usabilidad-Pruebas de usuario en las intranets.* Madrid: Prentice Hall. (pp.265-294)

Nielsen, J. (1995 01-Enero). *Nielsen Norman Group*. From Disponible en http://www.nngroup.com/articles/ten-usability-heuristics/

Nilsson, N. (1998) *Artificial Intelligence: A New Synthesis.* Morgan Kaufmann Publishers.

Nord, R.; Barbacci, M.; Clements, P.; Kazman, R.; Klein, M.; O'Brien, L.; & Tomayko, J. (2004) Integrating the Architecture Tradeoff Analysis Method with the Cost Benefit Analysis Method (CMU/SEI-2003-TN-038) Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University.

Noma, T. y Badler, N. (1997) A virtual human presenter. En Proceedings of the IJCAI '97 Workshop on Animated Interface Agents: Making them Intelligent, p´ags. 45–51

Odell, J. (2000) Agent Technology. Green Paper. Inf. T´ec. OMG Document ec/2000-08-01, Version 1.0, Object Management Group - Agent Working Group.

Odell, J. (2002) Objects and Agents Compared. Journal of Object Technology, 2(2): pa´gs. 41–53.

Pantelidis, V. S. (1996) Suggestions on When to Use and When Not to Use Virtual Reality in Education. VR in the Schools, 2(1): pa´g. 18.

Park, J. S. y otros (2005) Visible Korean Human: Improved Serially Sectioned images of the entire body. IEEE Transactions on Medical Imaging, 24: pa´gs. 352–360.

Parnas, D. & D. Weiss, D. (1985). The Modular Structure of Complex Systems, en Proceedings of the 7th International Conference on Software Engineering, Orlando, Florida, pp. 408-419.

Pazos Sierra, J. (2002a)Problemas. Espaaña: Universidad Politécnica de Madrid. 2002

Pazos Sierra, J.(2002b) El método experimental: tipos de experimentos. Documento pendiente de publicación.

Person, N. K. y otros (2001) Simulating human tutor dialog moves in AutoTutor. International Journal of Artificial Intelligence in Education, 12: p´ags. 23–29.

Piaget, J. (1954) The Construction of Reality in the Child. New York: Basic Books.

Pohjonen, Juha (1997) New learnings environments. Ver documento en: http://oyt.oulu.fi/~pohjonen/nle/nle/32.htm. Descargado en 14/11/2001.

Popovici, D. M. y otros (2005) VirtualDive - a VR-based educational virtual environment. Seventh International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC 2005).

Preece, J., Rogers, Y., Sharp, H., Benyon, D., Holland, S. & Carey, T. (1994) Human computer Interaction, Wokingham: Addison-Wesley.

Pressman, R. S. (2004) Ingeniería de software. São Paulo: Mc Graw Hill.

PROMETEUS . PROMETEUS: PROmoting Multimedia Access to Education and Training in EUropean Society [Web Page]. Consultado en septiembre de 2007. Disponible en: http://www.prometeus.org/.

Randell, B. y Buxton, J. (eds.) (1970) Software Engineering Techniques: Report of a Conference Sponsored by the NATO Science Committee. Scientific Affairs Division, NATO.

Ramdane-Cherif, A. y otros (2005) The Platform Based-Agents to Test and Evaluate Software Architecture. Journal of Object Technology, 4(1): pa´gs. 67–82.

Redding, R.E. (1992) A standard procedure for conducting cognitive task analysis. ERIC. Documentation Reproduction Service .No. DE 340-847.

Rickel, J. y L. Johnson (1997). *Integrating Pedagogical Capabilities in a Virtual Environment Agent.* En *Memorias Autonomous Agents 97*, pp. 30-38. Marina del Rey California USA. ISBN: 0-89791-877-0/97/02.

Rickel, J. y Johnson, W. L. (2002) Extending Virtual Humans to Support Team Training in Virtual Reality. En G. Lakemayer y B. Nebel (eds.), Exploring Artificial Intelligence in the New Millenium, pa´gs. 217–238. Morgan Kaufmann, San Francisco.

Rickel, J. y otros (2002a) Collaborative Discourse Theory as a Foundation for Tutorial Dialogue. En S.

A. Cerri, G. Gouard`eres y F. Paraguac ¸u (eds.), Proceedings of the Sixth International Conference on Intelligent Tutoring Systems, tomo 2363 de LNCS, pa´gs. 542–551.

Rivas, U. (2012) La semiosis: un modelo dinámico y formal de análisis del signo, Revista Razón y Palabra, No. 21, Universidad de Santiago de Compostela, consultado en http://www.razonypalabra.org.mx/anteriores/n21/21_mrivas.html.

Rodriguez-Toro, C. A., Tate, S. J., Jared, G. E. M., and Swift, K. G. (2003). "Complexity metrics for design (simplicity + simplicity = complexity)." Proceedings- Institution of Mechanical Engineers Part B Journal of Engineering Manufacture, 217(5): pp. 721-726.

Ryder, J. M. y Redding, R. (1993) Integrating Cognitive Task Analysis into Instructional Systems Development. Educational Technology Research & Development. (ET&RD). Vol. 41, No. 2, pp 75-96.

Ryokai, K., Vaucelle, C., & Cassell, J. (2002). Literacy Learning by Storytelling with a Virtual Peer. Documento presentado en In Proceedings of Computer Support for Collaborative Learning 2002. Disponible en http://web.media.mit.edu/~justine/publications.html

Salgueiro, F., Cataldi, F., Lage, F., García-Martínez, R. (2005a) Sistemas Tutores Inteligentes: Redes Neuronales para Selección del Protocolo Tutor. Proceedings del IV Workshop de Tecnología Informática Aplicada en Educación del X Congreso Argentino de Ciencias de la Computación. Pág. 255-266.

Salgueiro, F., Cataldi, Z., García–Martínez, R. (2005b). Los Estilos Tutor en el Modelado del Tutor para Sistemas Tutores Inteligentes. Revista de Informática Educativa y Medios Audiovisuales 2(4):70-79.

Salinas, Jesús (1996) Campus electrónicos y redes de aprendizaje. Recuperado el 8 de agosto de 2010. www.uib.es/depart/gte/salinas.html.

Salinas, Jesús (1997) Nuevos ambientes de aprendizaje para una sociedad de la información. Recuperado el 8 de agosto de 2010www.uib.es/depart/gte/ambientes.html.

Salkind, Neil. (1999) Métodos de investigación, Pearson Educación: México..

Schank, R. C., Berman, T. R. and Macpherson, K. A. (1999). Learning by doing. Reigeluth, C. M., *Instructional-Design Theories and Models. Volume II.* N.J.: Lawrence Erlbaum Associates.

Schank, R. C. and Cleary, C. (1995). *Engines for Education*. Erlbaum, Lawrence Associates.

Scolari, C. (2004). Hacer clic: Hacia una sociosemiótica de las interacciones digitales. Barcelona: Editorial Gedisa.

Shneiderman, B. (1987). Designing the User Interface: Strategies for Effective Human-Computer Interaction. Addison-Wesley, Reading, MA.

Schneiderman Ben, (1998), "Designing the User Interface: Strategies for Effective Human-Computer Interaction". 3ra Edición. Addison-Wesley.

Shneiderman, B. (1992). Designing the User Interface. Strategies for Effective Human-Computer Interaction. 2ª edición. Addison Wesley.

Scholtz & Morse (2002) "The Industry Usability Reporting Project" , A Standard Reporting Format for Summative Usability Evaluations, National Institute of Standards and Technology, Disponible en http://www.nist.gov/iusr.

SCORM . Sharable Content Object Reference Model. [Web Page]. Consultado en enero de 2008.

Disponible en: http://www.adlnet.org/index.cfm?fuseaction=DownFile&libid=648&bc=false.

Shaw, M. y Clements, P. (2006a) The Golden Age of Software Architecture. IEEE Software, 23(2): pa´gs. 31–39.

Shaw, M. y Clements, P. (2006b) The Golden Age of Software Architec- ture: A Comprehensive Survey. Inf. T´ec. CMU-ISRI-06-101, Institute for Software Research International, Carnegie Mellon University, Pittsburgh, PA, USA

Shi-Kuo Chang (1990) A Visual Language Compiler for Information Retrieval by Visual Reasoning. IEEE Trans. Software Eng. 16(10): 1136-1149 .

Sleeman, D. and Brown, J. S. (1982). *Intelligent Tutoring Systems*. Academic Press.

Sommerville Ian (2005). Ingeniería del Software, Prentice Hall. ISBN:8478290745 January 2005. 687 páginas 7a. Edición.

Sowa, John (2001) Glossary. Disponible en: http://users.bestweb.net/~sowa/ontology/glosss.htm.

Spiegel, M. R.(1991) Estadística. España: McGraw-Hill.

Sunar, M. S. y otros (2006) Effective Range Detection Approach for Ancient Malacca Virtual Walkthrough. The International Journal of Virtual Reality, 5(4): pa´gs. 31–38.

Sutherland, I. Greenfield, H. Koff, W. & Reemtsma, K. (1971) Moving Computer Graphic Images Seen from Inside the Vascular System. Transactions of the American Society of Artificial Internal Organs, 17: pa´gs. 381–385.

Swartout, W. R. y otros (2006) Toward Virtual Humans. AI Magazine, 27(2): pa´gs. 96–108.

The HUMAINE Consortium. (2004). Network of Excellence HUMAINE [Web Page].Consultado en enero de 2009 Disponible en: http://emotion-research.net/aboutHUMAINE/.

Thimbleby, H. (1990) User Interface Design. New York City: ACM Press.

Trikic, A. (2001) Evolving open learning environments using hypermedia technology. En: Journal of Computer Assisted Learning, No. 17, pp. 186-199, 2001

Tufte, E.(1990). Envisioning information. Cheshire, CT: Graphics Press.

Tufte, E. (1989). Visual design of the user interface. Armonk, NY: IBM Corporation.

Uhr, L. (1969) Teaching machine programs that generate problems as a function of interaction with students. En Proceedings of the 1969 24th ACM National Conference, pa´gs. 125–134. ACM, New York, NY, USA.

UNE 139801 (2003) Aplicaciones informáticas para personas con discapacidad, Requisitos de accesibilidad al ordenador, Hardware, Disponible en http://www.udc.gal/fcs/es/web-to/terapia/asignaturas/toyafam/08tema/UNE139801-2003.pdf

UNE 139802 (2003) Aplicaciones informáticas para personas con discapacidad, Requisitos de accesibilidad al ordenador, Software, Disponible en http://www.udc.gal/fcs/ga/web-to/terapia/asignaturas/toyafam/08tema/UNE139802-2003.pdf.

Van Heijst, G. Schreiber, A. Th. Y Wielinga, B. J.(1997) Using Explicit Ontologies in KBS Development, International Journal of Human-Computer Studies. Vol 46, 2/3, pp 183- 292.

VanLehn, K. (1988). Student Modeling. In Polson, M. C.,Richardson, J.,*Intelligent Tutoring System,* Lawrence Erlbaum.

Van Liere, R.; W. de Leeuw, J.; Mulder (2002) Virtual reality in biological microscopic imaging.

En Proceedings of IEEE International Symposium on Biomedical Imaging, p´ags. 879–882.

Vassileva, J. y otros (1999) A Multi-agent Approach to the Design of Peer- help Environments. En S. Lajoie y M. Vivet (eds.), Artificial Intelligence and Education, p´ags. 38–45. IOS Press.

Vassileva, J., McCalla, G. I. and Greer, J. (2003). Multi-Agent Multi-User Modelling in I-Help. *User Modelling and User Adapted Interaction. Special Issue on User Modelling and Intelligent Agents.* Vol. 13 (1), pp.179-210.

Viciana-Abad, R. y Reyes-Lecuona, A. (2005) Patient Modelling Using Expert Systems for Medical Training Simulations Based on Virtual Reality. En 7th International Conference on Virtual Reality, VRIC - LAVAL VIRTUAL.

Voronoi, G. (1907) Nouvelles applications des param`etres continus `a la th´eorie des formes quadratiques. Journal fu¨r die Reine und Angewandte Mathematik, 133: pa´gs. 97–178.

Wazlawick, R. S. et al. (1999) Virtual Museum: An Authory Tool for the Creation of Virtual Reality Museums to Support the Collaborative Learning on the Internet. Inf. t´ec., ProTeM Project.

Weber, G. and Brusilovsky, P. (2001). ELM-ART: An adaptive versatile system for Web-based instruction. Special Issue on Adaptive and Intelligent Web-based Educational Systems. *International Journal of Artificial Intelligence in Education*. Vol. 12 (4), pp.351-384.

Weber, G., Kuhl H.C. and Weibelzahl, S. (2001). Developing Adaptive Internet Based Courses with the Authoring System NetCoach. In *Hypermedia: Openness, Structural Awareness and Adaptivity*. (Reich, Siegfried , Tzagarakis, Manolis M., and De Bra, Paul M. E. Eds. Lecture Notes in Computer Science pp. 226-238.Berlin: Springer.

Webber, C. y Pesty, S. (2002) A two-level multi-agent architecture for a distance learning environment. En E. de Barros Costa (ed.), Workshop on Architectures and Methodologies for Building Agent-based Learning Environments (ITS 2002), p´ags. 26–38.

Weiers, R., (1986). Investigación de mercados, México:Prentice Hall Hispanoamericana.

Wenger, E. (1987a). *Artificial Intelligence and Tutoring Systems*. Los Altos: Morgan Kaufmann.

Wiig, K.(1999). Knowledge Management Foundations: thinking about thinking. How people and organizations create, represent, and use knowledge. USA: Schema Press.

Winn, W. (1993) A Conceptual Basis for Educational Applications of Virtual Reality. Tech Report R-93-9, University of Washington, Human Interface Technology Laboratory.

Woods, S. y M. Barbacci (199) Architectural evaluation of collaborative agentbased systems, Technical Report SEI-99-TR- 025.

Wooldridge, M. (1995). This is MyWorld: The Logic of an agent-oriented tested for DAI. Wooldridge, M.,Jennings, N. R.,*Intelligent Agents: Theories, Architectures and Languages.* v. 890) pp. 160-178.Germany: Springer-Verlag.

Wooldridge, M., Jennings, N. R. and Kinny, D. (1998). The GAIA methodology for Agent- Oriented Analysis

and Design. *Journal of Autonomous Agents and Multi-Agent Systems.* Vol. 3 (3), pp.285-312.

Wooldridge, M. y Jennings, N. R. (1995a) Agent Theories, Architectures, and Languages: a Survey. Intelligent Agents, p´ags. 1–22.

Wooldridge, M. y Jennings, N. R. (1995b) Intelligent agents: theory and practice. The Knowledge Engineering Review, 10(2): pa´gs. 115–152.

Yildirim, Z., Ozden, M. Y Aksu, M (2001) Comparison of hypermedia learning and tradicional instruction, The Journal of Educational Research, USA: Bloomington, Marzo Abril 2001.

Youngblut, C. (1998) Educational Uses of Virtual Reality Technology. Inf. t´ec., Institute for Defense Analyses.

Zambonelli, F. & Parunak H. V. D. 2003. Signs of a revolution in computer science and software engineering. In *Proceedings of the 3rd International Workshop on Engineering Societies in the Agents World*. Lecture Notes in Computer Science, vol. 2577. Springer-Verlag, New York.

Martha Gutiérrez Miranda
martha_g_miranda@hotmail.com *Curriculum Vitae*

Formación Académica

• Licenciada en Diseño de la Comunicación Gráfica egresada de la UAM-Azcapotzalco en el año 1997. Maestra en Mercadotecnia en la Universidad ETAC y obtiene grado académico en el año 2007 con excelencia Académica.

Actividad docente

• Es profesor-investigador en la Universidad Autónoma Metropolitana desde 1997. Ha tenido la oportunidad de impartir cursos (alrededor de 50) para Educación Continua de la UAM-Azcapotzalco, Coodinación de Vinculación en la UAM-Azcapotzalco, el Tec de Monterrey campus Estado de México, la Universidad del Valle de México, el Colegio Indoamericano, El Periódico Excélsior, El Gobierno Municipal de Tultitlán, OPDM Tultitlán, etc. Adicionalmente ha participado en diversas comisiones académicas para el desarrollo e implementación de Planes y Programas de Estudio en la UAM-Azcapotzalco, formó también parte del Comité Académico de la Licenciatura en Diseño de la Comunicación Gráfica de 2007 a 2010 y fungió como Coordinadora de Investigación del Grupo GROPUS, de 2007 a 2011.

Desarrollo Profesional

• A nivel profesional ha desarrollado proyectos de diseño gráfico y publicidad desde el año 1997, a empresas como GAcción, CIMCO, Terza, Plásticos Panamericanos, Coca-Cola, Hierro y Maderas San Cristóbal, ICA, Transportadora Especializada San Marcos, TASA, Envios, S.A., Bacardi y Compañía, OHM, Suburbia, Plafusa, IFAD, en otras. Ha participado como asesor para proyectos interdisciplinarios, destacándose el proyecto FORHUM, desarrollado con el Colegio de Postgraduados y OHIO State University y el Programa interdisciplinario de gestión de Diseño para el desarrollo de Marcas y mejora del proceso de comercialización para las Cooperativas que integran a la Red Mexicana del Amaranto, en la comunidad de Tulyehualco, D.F.y actualmente es parte del Proyecto de Rescate del Archivo Histórico de la Ciudad de Querétaro.

Participaciones Académicas y Redes e Investigación

• Forma pate de la REDLoac (Red de Docentes Latinoamericanos y del Caribe) y la Red académica EDMedia Internacional desde 2003, así como de la AACE (Asociation for Advance in Computer Education) a partir de 2001. También ha participado como ponente en eventos nacionales e internacionales como: 3rd International Conference on Education an New Learning Technologies (Barcelona, España), el Congreso Internacional de Eduación a Distancia (Guadalajara, México), el Congreso Virtual Educa 2010 (Santo Domingo, República Dominicana), El COMIE (Veracruz, México), El Congreso Iberoamericano de Materiales Didácticos Innovadores (México, D.F.), Expo-Ciac (México, D.F.), Congreso PT3 (Preparing Tomorrow´s teachers) (Albuquerque, New Mexico y Orlando, Florida, USA), WEBNet (San Antonio, Texas, USA), Seminarios Internet 2 (México,D.F.), entre otros.

Adicionalmente ha tenido la oportunidad de escribir artículos de investigación en publicaciones como la Revista Tiempo de Diseño, la Revista Educanet, Memorias de Congresos como el EDUlearn, VirtualEduca, EDMedia, además de materiales didácticos de apoyo docente como antologías, Notas de curso y cuadernos de apoyo.

Actualización y Formación

Ha tomado cursos tanto de formación profesional como de formación docente, destacando seminarios y congresos, tanto nacionales como internacionales, como: Designfest, Seminario Merca-publi-diseño, Congreso EDMEDIA, Congreso EDULEARN, Diplomado en Práctica Pedagógica interdisciplinaria, COMIE, Cursos de Programación, Animación Digital, Retoque Digital, Diseño de Marca, Microenseñanza, Evaluación Curricular, Creatividad, Competencias docentes, Educación virtual, Objetos de Aprendizaje, SCORMS, etc.