Universidad Carlos III de Madrid
Escuela Politécnica Superior

Ingeniería en Informática

Proyecto Fin de Carrera

DISEÑO DE UN MUNDO VIRTUAL PARA LA ENSEÑANZA DE ARQUITECTURA SOFTWARE

Autor: Verónica Casado Manzanero
Tutor: Anabel Fraga Vázquez

Diciembre, 2009

Agradecimientos

En primer lugar querría dar las gracias a mis padres, por su apoyo, comprensión, generosidad, por darme todo lo que necesito y más, y sobre todo, por enseñarme a ser como soy y servirme de ejemplo para convertirme en mejor persona cada día.

También me gustaría recordar a mi hermana por ayudarme siempre en lo he necesitado con esa gran dosis de paciencia que sé que ha de tener. A mis amigas Carol y Raquel, por permanecer siempre a mi lado a pesar de estar semanas sin vernos.

A mi novio Marcos, por pasarme esa paciencia y tranquilidad suya que tanto aprecio, por apoyarme en todo momento, por creer en mí y por permanecer a mi lado durante estos seis largos años.

Por último, agradecerle a mi tutora Anabel toda la ayuda prestada, tanto en las asignaturas como en este proyecto. Gracias a su inestimable ayuda y entusiasmo he podido completar con éxito el trabajo aquí propuesto. Ha sido un verdadero placer trabajar con ella.

Contenido

Índice de Figuras

Índice de Tablas

1. INTRODUCCIÓN Y MOTIVACIÓN

En una época en la que la formación es imprescindible para crear buenos profesionales, han surgido numerosos puntos de apoyo a la enseñanza tradicional aprovechando las ventajas que ofrecen las tecnologías. La mayor parte de las universidades cuenta con un sistema de gestión de cursos, accesible a través de Internet, en el que los alumnos pueden encontrar los materiales utilizados en las clases, enlaces externos de apoyo, ejercicios, exámenes de años anteriores, etc. Es un sistema virtual que provoca una extensión de las aulas hasta los hogares de los alumnos.

Igualmente, en la red han surgido entornos virtuales en los que escuelas, universidades, o simples cursos, son impartidos. Gracias a ellos, es posible acercar las aulas a gente que, ya sea por causas profesionales o personales, no pueden acudir a clases presenciales. El famoso mundo virtual Second Life acoge infinidad de instituciones entre sus numerosas islas, dejando hueco a toda clase de formaciones académicas: es posible aprender idiomas, acceder a las sucursales de las universidades más conocidas, o realizar cursos con los que especializarse.

Quizá incluso, en un futuro, y mediante instrumentos de realidad virtual, el alumno pueda sentir que está acudiendo verdaderamente a clase desde su hogar, verse inmerso en un ambiente académico tal y como se concibe hoy en día, aunque sin tener que realizar desplazamientos, pudiendo relacionarse con gente de cualquier parte del mundo.

Por otro lado, la Arquitectura de Software, dentro del mundo de la informática, es un proceso importante que permite la realización de proyectos de una manera más óptima, garantizando en gran medida su éxito, y facilitando la labor de un equipo completo de desarrollo y mantenimiento.

El objetivo del presente proyecto es aunar los beneficios que ambas ramas aportan y crear un mundo virtual en el que se enseñe Arquitectura de Software a nivel universitario. Una parte importante para tal fin es la investigación sobre las tecnologías que harán que esta idea sea viable, así como proyectos anteriores que sirvan como referentes de éxito en una empresa similar. De la misma manera, se estudiarán herramientas que puedan ayudar a la consecución del proyecto y a la implantación del mismo, contando con las diferentes posibilidades que esto plantea.

El fin último es definir los pasos que se han de dar para la construcción del sistema que haga una realidad de este mundo virtual. Por ello, tras la investigación inicial, se procederá a realizar un análisis de las necesidades de la aplicación, y se definirá una metodología de enseñanza específica para esa área concreta del desarrollo de software. Finalmente, contando con toda esta información, se realizará el diseño del sistema. Con la conclusión de este proyecto, quedarán establecidas las instrucciones precisas que llevarían a un equipo de desarrollo a realizar con éxito la construcción del sistema.

2. ESTADO DEL ARTE

2.1. Mashup

2.1.1. Introducción

Actualmente, cuando Internet se ha adentrado de lleno en los hogares y empresas, surge una nueva forma de crear las páginas Web. Hace ya algunos años se introdujo el concepto de Web 2.0, que se refiere a una novedosa manera de entender la Web en la que la interacción del usuario es fundamental para la creación de los contenidos. Un perfecto ejemplo son las famosas redes sociales.

En los últimos tiempos, ha crecido la utilización de mashups, considerados como "un paso más allá". Se puede definir el término mashup como una aplicación Web, o sitio Web, que combina a la perfección el contenido de más de una fuente en una experiencia integrada [1]. Es decir, un sitio o aplicación Web que utiliza los servicios que ofrecen otras páginas y los integra, formando un único sitio. Es lo que podría llamarse una aplicación o sitio Web híbrido. Como se puede comprobar, la característica fundamental es que no es una página estática en la que el contenido lo actualiza su propio dueño, sino que se renueva constantemente al ritmo que lo hacen aquellos sitios de los que se extrae la información.

El término mashup no se ha utilizado por primera vez en el mundo de la informática. Apareció en Jamaica, y se utilizaba para describir la ruptura de algo. Más tarde la música se hizo eco del término y lo utilizó para describir aquellas canciones que eran mezcla de otras, normalmente hip-hop y reggae. De esta misma forma, mashup se utiliza para definir una aplicación Web híbrida, la mezcla de dos o más sitios Web diferentes para crear uno nuevo.

La integración de los contenidos se hace mediante programación, utilizando típicamente APIs y librerías que ofrecen los sitios Web originarios, así como diferentes tecnologías que permiten recoger la información de dichos sitios.

Para utilizar mashup en un sitio Web, se siguen normalmente tres acciones básicas [1]:

- Los datos se extraen de los sitios Web de origen.
- Estos datos son traducidos de forma que tengan significado en el sitio Web de destino.
- El nuevo paquete de datos es enviado al sitio Web de destino.

Es importante remarcar que para que un sitio o aplicación Web se considere mashup, debe integrar su contenido utilizando directamente los datos y servicios que ofrece el origen, por ejemplo, mediante APIs.

El ejemplo más común para hablar de mashups son aquellas páginas que utilizan la tecnología aportada por Google Maps para representar información geográfica. Esta original idea surgió en 1991, cuando David Gelernter propuso utilizar un software para crear una simulación por ordenador del mundo físico, haciendo posible mapear todo, desde el flujo del tráfico a ventas de pisos en una pantalla del ordenador [4]. Gelernter vio cómo su proposición se llevaba a cabo cuando Google y Yahoo anunciaron la publicación de su documentación para que los programadores pudieran acceder a sus mapas y recrearlos.

Los mashups, en su mayoría, tienen un uso no comercial; el objetivo de los desarrolladores es presentar la información que desean de esta forma. Sin embargo, Google, Yahoo y Microsoft están creando servicios para unirlos a publicidad mediante anuncios contextuales vinculados a localizaciones específicas. Esto viene a ser una extensión de los ya existentes Google AdWords (también proporcionados por Yahoo y otras compañías), que muestran los anuncios según la consulta hecha en el buscador, o Google AdSense, que añade publicidad al sitio según el contenido de la página en la que se muestre. En este nuevo caso, la publicidad aparecería situada en el mapa resultante de una petición hecha.

La pregunta más importante, sin embargo, es si realmente son necesarios los mashups. Como casi todo, son relativamente necesarios y, por supuesto, algunos son más útiles que otros. Los más cercanos a los usuarios, los que suponen la fusión con mapas, son lo que se pueden ver como más ventajosos: es un gran avance poder encontrar en un único mapa toda la información sobre restaurantes de una zona, hoteles, pisos en venta, etc. Es más, podría decirse que son la evolución de los G.I.S. (sistemas de información geográfica), siendo incluso más accesibles para cualquier programador o diseñador que desee utilizarlos en su página Web, ya sea personal o empresarial. Según Perry Evans, no es la novedad lo que hace interesante este tipo de mashups, “lo que los diferencia es la accesibilidad y el hecho de que el número de participantes en publicidad local con un objetivo está creciendo” [4].

2.1.2. Arquitectura

Todos los mashups siguen una arquitectura única, compuesta de tres partes [2]:

- La fuente de donde provienen los datos.

  Son aquellos sitios de los cuales se recoge la información para crear la nueva página Web. Normalmente viene a través de APIs o librerías, las cuales hacen más sencillo el acceso a los servicios ofrecidos. Para algunas webs, como las de noticias, es más común encontrar que los datos se obtienen a través de otros protocolos Web como RSS, REST y Web Service.

  Sin embargo, no todos los sitios están adaptados para entrar a formar parte de un mashup. En estos casos se hace lo que en inglés se llama “screen scraping. Esta técnica consiste en, mediante una herramienta especializada, extraer información de los

contenidos de las páginas proveedoras, traduciéndolas a un lenguaje que se pueda utilizar. La dificultad de esta técnica viene dada por la limitada capacidad de la traducción de un contenido que está enfocado en primera instancia a los usuarios y que, por ende, se encuentra en lenguaje natural.

- El sitio Web de destino, aquél que va a alojar el mashup.

Este sitio se hará eco de los datos obtenidos de las fuentes del primer punto y, por lo tanto, no es dueña de los mismos. Aquí se debe diferenciar la parte lógica de la física del mashup, puesto que no tienen por qué localizarse en el mismo sitio. Cuando se habla del lugar que aloja el mashup, se refiere al emplazamiento donde la parte lógica del mashup reside, que no tiene por qué ser donde se ejecute.

Esta última aserción viene a distinguir dos situaciones que pueden ocurrir, dependiendo de la implementación del mashup. Por un lado se puede implementar como una Web dinámica del lado del servidor, siendo éste el lugar donde se ejecuta el sitio. Por otro, se puede tener la opción de utilizar tecnologías como *Ajax*, actualmente muy extendido, que permite ejecutar la página en el lado del cliente.

La introducción de Ajax en este tipo de sitios entra dentro de lo que se llama "Aplicaciones Ricas de Internet" (RIAs, Rich Internet Applications). Como se puede deducir, esta inclusión puede traer numerosos beneficios, más aún teniendo en cuenta el dinamismo que caracteriza su construcción. Algunas de estas ventajas son la reducción de la sobrecarga del servidor en el que se aloja el mashup al recoger toda la información y después tener que enviársela al cliente, y una mejor interacción con el cliente, al no recargar la página entera. (Para más información sobre este tema ver el apartado "2.1.8.1 Ajax" del presente documento.)

Lo más común, sin embargo, es encontrarse aplicaciones que utilizan ambas formas de ejecución, recogiendo parte de los datos en el lado del servidor, y otra parte en el lado del cliente, intentando optimizar al máximo la utilización de las técnicas de recuperación de datos, así como la experiencia del cliente.

- El navegador del cliente es la última de las partes de la arquitectura.

Es el lugar en el que el usuario podrá ver el mashup, donde se éste es representado gráficamente. Como se ha comentado, la generación de la página puede llevar código Javascript, siendo necesario para la correcta visualización de la misma que el navegador soporte este tipo de tecnología.

2.1.3. Clasificación de mashups

Como en todas las novedades tecnológicas, aún no hay un consenso sobre los tipos de mashups existentes, si bien la distinción entre unos y otros es bastante homogénea. En este proyecto se ha decidido optar por los géneros propuestos por Duane Merrill [2]:

- Mashups de mapeado. Están basados en la ingente cantidad de información almacenada que está relacionada, de una u otra forma, con las localizaciones. Aprovechan la mayor compresión por parte del usuario de una información cuando la ve representada, más aún si dicha visualización se hace sobre un mapa. Para la realización de este tipo de aplicaciones se utiliza, en mayor medida, el API proporcionado por Google: su famoso *Google Map*. Dicha librería está muy extendida y es relativamente sencilla de usar. Sin embargo, no es la única propuesta en el mercado, puesto que tiene numerosos competidores, como *Virtual Earth* de Microsoft, *Yahoo Maps* de Yahoo y *MapQuest* del AOL.

- Mashups de videos y fotos. Con el nacimiento de la denominada Web 2.0 han florecido en Internet páginas del estilo de *Flickr* o *YouTube*. Estas páginas albergan fotos y videos que, en la mayoría de las ocasiones, pueden ser relacionados con información externa. Esta situación se ha utilizado para crear mashups con dichos recursos e información extraída de otras páginas. Un ejemplo podría ser una página de noticias en las que las fotos se expusieran atendiendo a la noticia y a la información referente al archivo multimedia.

- Mashups de búsqueda y compras. Hay varias páginas en la Web en las que se comparan productos y precios utilizando agentes inteligentes que intentan ofrecer al usuario la mejor oferta posible. Ahora, esta tarea se puede llevar a cabo más fácilmente gracias a los mashups. Algunos portales destacados en la venta de productos, como eBay y Amazon, ofrecen APIs para que dichas herramientas de comparación de productos puedan acceder a su contenido mediante código.

- Mashups de noticias. Otro de los legados de la Web 2.0 es el uso de lo que se llaman feeds, como RSS y Atom, explicados en el apartado de 2.1.8 Tecnologías relacionadas de este mismo documento. Utilizar estas tecnologías en mashups permite agregar contenido de las páginas que ofrecen esta sindicación en otra página distinta, creando una oferta de noticias más personalizada, atendiendo a cuáles de estas páginas se suscriba el usuario.

Otra posible diferenciación entre mashups es la ofrecida en un artículo de Internet [3] realizado por Shaw, una consultora experta en este tema:

- Mashups de consumidores. Son los más conocidos al ser los más extendidos en la red, buscando tener un uso para el usuario normal. Un buen ejemplo son todas aquellas páginas que utilizan el API de Google Maps para integrarlo a diferentes funcionalidades, como podría ser el alquiler de pisos.

- Mashups de datos. Están enfocados directamente a la integración de datos provenientes de varias fuentes. Utiliza para ello, principalmente, fuentes a modo de RSS.

- Mashups empresariales. Este tipo de mashups, como su nombre indica, está orientado al mundo empresarial, e integra información tanto externa, de distintos sitios Web, como información interna de la propia empresa, generando un sitio Web que se adapte especialmente al negocio para el que está diseñado.

La autora destaca por encima de los demás mashups la utilidad de los empresariales. En el mundo Web más superficial los dos primeros podrían ser suficientes para cubrir las necesidades de la mayoría de los usuarios. Sin embargo, hay un mundo más profundo, aquellos sitios o portales controlados por empresas, que ofrecen realmente un servicio. Estos sitios están destinados a múltiples personas implicadas en el proyecto, cruzan límites organizativos, y están basados en procesos, como contratar nuevos empleados, realizar informes, etc.

Los mashups de datos y consumidores no pueden cubrir todas las necesidades que se generan en una empresa, simplemente no son suficientes para la implementación de todas las tareas que se necesitan para el correcto funcionamiento de la misma. Es por esto que surgen los mashups empresariales, que, además de ser capaces de reunir información y servicios de varias fuentes, cuentan con una potente capacidad de procesamiento. Esta capacidad es la que les brinda la oportunidad de satisfacer y aunar todas las necesidades que aparecen en el día a día de la empresa.

2.1.4. Integración de los datos

La integración de los datos usando mashups no requiere complicación en la mayoría de los casos, siendo lo normal superponer una información a otra. De nuevo el ejemplo más común que se puede señalar es el situar ciertos datos geográficamente utilizando Google Maps. Sin embargo, hay ciertos casos en los que los datos son extremadamente abundantes o, simplemente, no es tan sencillo relacionarlos entre sí, haciendo que las consultas sobre los mismos sean más lentas.

Es en estos casos son en los que se han centrado tres autores de la Universidad de Leipzig. En un artículo conjunto [5] hablan sobre un framework para el desarrollo de mashups de integración de datos complejos y dinámicos. Consiste en una serie de componentes para generar consultas y relacionar los datos permitiendo una transformación adicional de éstos, con lo que se sacaría un mayor partido a la Web. El gráfico siguiente pretende mostrar cómo es el diseño inicial del framework:

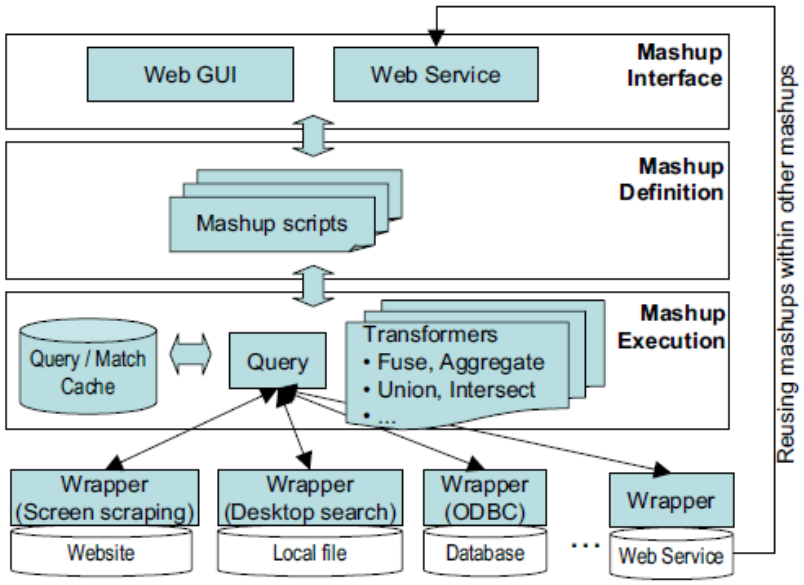

Imagen 1: Arquitectura del framework de integración de datos

La primera capa es la interfaz o servicio Web que utiliza al mashup y a través de la cual se muestran los resultados al usuario.

La segunda capa se refiere a la definición del mashup. Según su diseño, esta definición la hace el desarrollador, especificando su algoritmo con scripts en los que se insertan las instrucciones en lenguajes de programación de alto nivel. Dicho script tiene como finalidad proveer un servicio tal y como lo hace un servicio Web, pudiendo ser invocado por una interfaz u otro servicio dentro de un mashup externo, a través de la ya explicada primera capa.

En la última capa, los wrappers transforman los datos contenidos en diferentes fuentes de información en un archivo XML para que, de esta forma, la entrada al framework tenga siempre la misma estructura, lo que facilita el tratamiento de los resultados.

A continuación se procederá a explicar su funcionamiento global: el mashup, y por tanto el conjunto de consultas necesarias para su visualización, se divide en varios segmentos, cada uno asociado a un script para poder ejecutarlos en orden. Cuando se ha terminado con el primer script, se muestra su solución al usuario, exponiendo así un resultado intermedio, a la vez que el framework puede realizar en background el siguiente script para refinar aún más los primeros resultados. Se podrán ejecutar más scripts para hacer diferentes consultas a petición del usuario, por ejemplo, si éste pulsa un botón determinado, especificando aún más los resultados.

La tercera capa coge los datos recogidos por los wrappers, ya transformados en ficheros XML para homogeneizar su formato, y hace las consultas necesarias para ejecutar los scripts. Adicionalmente, se pueden hacer una serie de transformaciones, siendo algunos ejemplos comparar objetos, hacer agregaciones en las consultas, etc.

En definitiva, se trata de un software que permite acelerar el resultado de mashups, cuyas complejas consultas hagan que el tiempo de respuesta sea demasiado grande como para que se considere aceptable.

2.1.5. Enterprise Mashups o Mashups Empresariales

Si bien mashups se está extendiendo en gran parte a través de la red, siendo usado por los portales para enriquecer su oferta al usuario que navegue por ellas, hay otras formas de aprovechar esta nueva manera de entender la integración.

John Cupri y Chris Warner han escrito una serie de tres artículos, de los cuales aquí se tratarán dos, [6] y [7], en los que proponen el uso de mashups en la empresa, tal y como se conciben hoy en día las aplicaciones en este ámbito: siguiendo una arquitectura SOA (Service Oriented Architecture), comentada en este mismo documento en el apartado 2.1.8 Tecnologías relacionadas.

Lo que ambos autores proponen es el uso de mashups como intermediarios entre sus propias herramientas, que usualmente siguen una arquitectura SOA, y los usuarios de las mismas.

Básicamente un enterprise mashup funciona muy parecido a un Web mashup, aunque debe mantener los requisitos que se imponen por el hecho de estar incorporados a la forma de trabajar de la empresa, como la seguridad, conformidad e integración con otras herramientas ya existentes.

Según estos dos autores, una definición de mashup que coincida con estas restricciones podría ser la siguiente: "*una micro-combinación centrada en el usuario de fuentes externas e internas basadas en estándares.*" Esta complicada definición se entiende mejor si se explica cada una de las características expuestas por separado.

La más importante de dichas características es la que haced referencia a centrarse en el usuario. Todos los mashups están destinados a un usuario final, que será el que vea los resultados de haber aunado las distintas fuentes de datos en una sola. En el caso de los enterprise mashups, además, connota la escasa participación del departamento de TI (Tecnologías de Información), puesto que utiliza datos normalizados fácilmente integrables y que tienen significado por sí mismos.

Cuando se está utilizado mashups para presentar información en la empresa, lo normal es coger información de numerosas fuentes, de la misma forma que la cantidad de información de cada una de ellas es relativamente baja, sobre todo si se compara con los miles de millones de datos que se guardan en las empresas.

En cuanto a los datos internos y externos, suelen estar basados en estándares, lo que no difiere demasiado de lo que ya se ha explicado de mashups, los cuales suelen coger

información que viene en un determinado formato para facilitar la lectura de la misma. Si bien en los Web mashups no siempre este formato pertenece a un estándar, pudiendo recordar como ejemplo la técnica de screen scraping, en el ámbito empresarial sí se hace más importante el uso estricto de los mismos, siendo lo más populares WSDL (utilizado por SOAP), REST y RSS.

Como se puede comprobar, la mayor diferencia entre los mashup de Web y los mashup de empresa es la finalidad y, por tanto, la forma de utilizarlos. Para comprender mejor cómo funcionan dentro de la empresa se van a enumerar las características que cumplen:

- Los mashups deben colaborar con otras herramientas ya instaladas en la empresa, ya sea para buscar en ellas, compartir información, etc.
- La información que manejan normalmente viene en pequeñas cantidades y está relacionada entre sí.
- Puesto que el usuario final al que están destinados suele necesitar la información en el momento en el que se la solicita al mashup, éstos deben responder en tiempo real a dichas peticiones.
- Los mashups han de funcionar en el entorno para el que los han desarrollado, no debería ser necesario implantar una nueva infraestructura para su utilización.
- La cantidad de información que manejen debe ser lo suficientemente pequeña para que el mashup no se sature.
- Los mashups deben tener una interfaz con la que el usuario pueda trabajar, pues éste es el fin de los mismos.

Es importante tener claro que al introducir mashups en la empresa no se pretende eliminar las herramientas ya extendidas en este mundo. La particularidad de mashup, lo que lo hace tan dinámico y útil, es que se puede combinar con todas ellas para aumentar su beneficio, pues son fuentes excepcionales de las que obtener la información que el mashup muestra al usuario.

Es igualmente cierto que, aunque la utilización de mashups empresariales puede conllevar grandes beneficios, no es tan simple su realización como los mashup de Web debido a que la empresa necesita una manera diferente de hacer las cosas. Un Web mashup confía en la autenticidad, veracidad y seguridad de las fuentes de datos que consulta, mientras que en una empresa eso supone riesgos que no deben permitirse. Además, un Web mashup, en la mayoría de los casos, simplemente muestra la información recogida, mientras que en el mundo de los negocios los datos suelen ser modificados, añadidos y borrados continuamente.

Cupri y Warner definen dos tipos de mashups: mashups de navegador y mashups de servidor. Los primeros son los mashups más usuales, y hacen la mayor parte de las operaciones en el lado del navegador. Los otros, por el contrario, centran el peso de las mismas en el

servidor donde se aloja el mashup. Los escritores de estos artículos abogan, en el escenario empresarial, por los mashups de servidor, dando una serie de razones para apoyar su causa:

- Pueden tener contratos pre-negociados con las fuentes de datos más comunes.
- Pueden proveer una interfaz visual sencilla para integrar o juntar los servicios virtuales.
- Son capaces de almacenar de manera segura información de autentificación y autorización de los servicios virtualizados.
- También tienen la capacidad de ayudar al usuario a compartir mashups con las herramientas con las que trabajan cada día.
- Permite combinar datos externos e internos de una manera sencilla.

El uso de mashups dentro de la empresa permite optimizar los recursos y ofrecer una respuesta más rápida, minimizando el tiempo que tiene que estar el cliente buscando datos y relacionándolos entre sí.

En las empresas, además, se está extendiendo el uso de una arquitectura orientada a servicios, la ya mencionada SOA. A pesar de las grandes ventajas que aporta, también conlleva riesgos que dificultan su implantación. Una de las reticencias viene dada cuando no se trabaja dentro de la misma red interna de la empresa, debido a que los servicios contestan a las peticiones surgidas sin la necesidad de comprobar quién ha hecho la solicitud ni si tiene las credenciales adecuadas. Si el SOA ofrece sus servicios al exterior, se expone a que cualquiera pueda intentar acceder a él y hacerle una petición para aquello que ofrece, y éste contestará, pues no comprobará la identidad del consumidor del servicio.

En este punto es donde entran los mashups, como punto intermedio entre el SOA y los usuarios que accedan a los servicios ofrecidos por la empresa, tal y como se muestra en la Figura 1. De manera sirve como enlace directo entre ambos, controlando la seguridad. Incluso pueden desempeñar temporalmente el trabajo del servicio, o al menos parte de éste, si se encuentra, por ejemplo, en estado de actualización por algún cambio en el SOA. Igualmente, la introducción de mashups ayuda a que los usuarios tomen las herramientas ofrecidas por la empresa como parte de su vida cotidiana dentro de su trabajo.

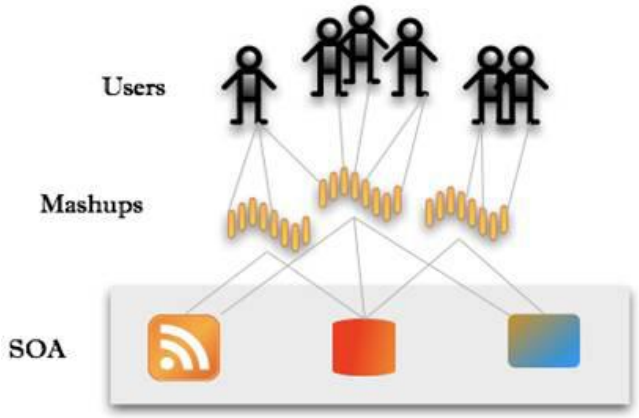

Imagen 2: Mashups como capa entre SOA y usuarios

Hay múltiples escenarios en los que se puede instaurar un mashup en la empresa, tal y como indica la siguiente figura [8].

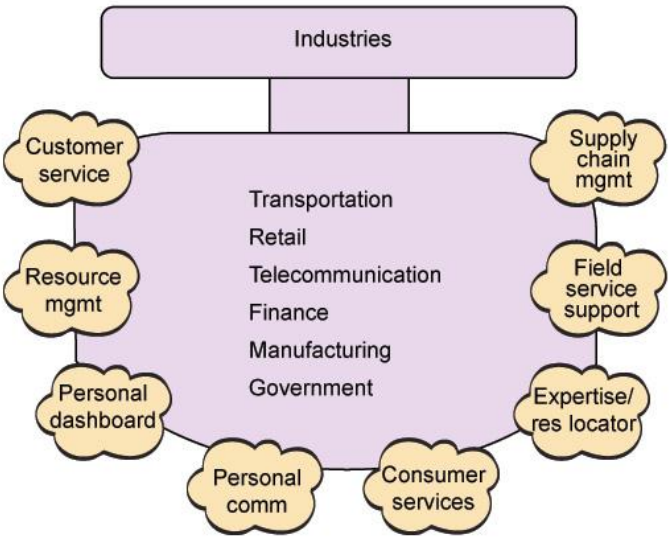

Imagen 3: Escenarios de un mashup empresarial

Dos de estos escenarios son los más comunes, la gestión de recursos (resource management) y las tablas personales en las que se muestre información de interés (personal dashboards). Un ejemplo de la primera podría ser la gestión de los datos sobre las ventas de productos en respuesta a eventos naturales, o las ventas de determinados artículos teniendo en cuenta la situación geográfica en la que se encuentran las distintas tiendas.

En todos los casos, la finalidad es recoger información de distintas fuentes y mostrarlas en una misma pantalla, pudiendo observar las implicaciones de los datos frente a distintas variables. Una buena aplicación de los mashups es en la toma de decisiones de altos cargos de las empresas: contar con información puede mejorar las decisiones asumidas, mejorar la productividad, etc.

2.1.6. Ventajas y desventajas

Es evidente que el uso de mashups proporciona una nueva forma de entender la visualización de los datos. Ahora es posible que un desarrollador, sin mayor esfuerzo que utilizando dos servicios accesibles a través de la red que han sido realizados por otros, cree un sitio Web rico en información y, en la mayoría de los casos, fácilmente leídos en un primer vistazo. No es necesario disponer de grandes recursos, ni siquiera en el lado del servidor, pues todos los datos son recogidos de las fuentes a partir de las cuales se construye el mashup. Únicamente hay que aportar una amplia dosis de creatividad para que el producto final interese a los usuarios de la red.

La utilización de mashups se está viendo tan extendida precisamente por su facilidad de construcción y por lo atrayente que resulta que en una única página se fundan las ventajas de otras dos ya interesantes de por sí. Ni qué decir tiene su amplio repertorio de posibles funcionalidades dentro de una empresa al mezclarse con las herramientas más importantes ya extendidas, entre las cuales destaca la ayuda en la toma de decisiones al poder visualizar distintas situaciones, con el incentivo de estar toda la información visible sin necesidad de acudir a herramientas de data warehouse, sobre todo cuando el tamaño de la información no es eminentemente grande.

Desafortunadamente, no todo son ventajas, existe una dependencia insalvable con los proveedores de las fuentes de datos. Sin ellos, el mashup no existiría; si la información en ellos no es actualizada, el mashup tampoco se actualiza; si no ofrece datos verídicos, el mashup tampoco es fiable; si sus datos no son accesibles por una posible caída del servidor, el mashup tampoco podría mostrar la información pertinente, etc. Eso, sin entrar a comentar la propiedad intelectual al utilizar software desarrollado por otras personas distintas al creador del mashup, o a la Ley de Protección de Datos.

Por trabajar en la red también se crean algunos riesgos, como que ésta esté saturada y el mashup tarde en la recogida de datos y, por tanto, en mostrar al usuario su solución. Esto en un sitio Web es un problema grave, puesto que una respuesta rápida es fundamental para que el usuario se quede en la página y vuelva a visitarla, siendo esto al fin y al cabo para lo que se hacen los sitios Web.

Otro aspecto a tener en cuenta son las dificultades que se puede encontrar el desarrollador a la hora de implementar su mashup. Si bien antes se ha comentado que la creación de un mashup no supone ningún reto, hay que saber que no siempre es así. Uno de los grandes problemas que se pueden plantear es la integración de los datos cuando éstos son muy heterogéneos entre sí, como por ejemplo cuando están en diferentes idiomas. En ese caso se precisaría la introducción de un traductor intermedio que hiciera que ambas fuentes de datos fueran entendibles por el mashup. De igual forma, podría ser necesario realizar una serie de transformaciones a los datos, o incluso limpiarlos, por si hay información que no es útil o está incompleta. Se puede ver un ejemplo si se recuerda el framework explicado en el apartado 2.1.4 Integración de los datos, en el cual existían unos componentes, denominados

wrappers, que se encargaban de hacer las transformaciones y traducciones necesarias a los datos de las distintas fuentes y ponerlos en el mismo formato antes de que el mashup los tratase.

Muchos de los mashups utilizan Ajax en su código. Tal y como se comenta en el apartado destinado a esta tecnología, 2.1.8.1 Ajaxa pesar de los grandes beneficios que puede aportar su uso, también conlleva riesgos, como que el navegador del usuario no soporte JavaScript, quedando inutilizado el sitio Web.

Por último mencionar los riesgos de seguridad que pueden surgir al utilizar información de otras fuentes. De este tema habla Aaron Bohannon en su artículo "*Building Secure Web Mashups*" [9], al explicar que este riesgo surge porque los mashups no siguen la "política del mismo origen". Ésta impone algunas restricciones:

- Un script (aquellos que hacen visibles los documentos Web en el navegador) de un origen no puede leer o escribir el contenido o los metadatos de un documento de origen distinto, a menos que tenga una referencia suya o sea un navegador con una política de acceso permisiva.
- Un script de un origen no puede acceder al entorno JavaScript de un frame de un origen diferente.
- Un script de un origen no puede usar XMLHttpRequest para comunicarse con un sitio de diferente origen.
- Esta política restringe los accesos de los scripts a las cookies y a los plug-ins de los navegadores.

Como se puede comprobar, tras conocer todo lo explicado de mashups en los puntos anteriores, éstos violan todas las reglas impuestas por esta política. ¿Qué ocurre entonces? Que todas las técnicas de construcción de mashups permiten que estas reglas sean ignoradas, teniendo que buscar la seguridad del sitio Web por otros medios, como la inclusión de frameworks intermedios que aseguren: la privacidad del usuario para que acceda a la página que desee sin ser interceptadas sus peticiones por una tercera parte, la integridad de los datos aportados por el usuario a la Web, así como los datos mostrados al mismo por parte del sitio y la autenticidad de las partes involucradas, que la página que se visita sea realmente la que el usuario desea.

En este mismo artículo se propone una serie de soluciones:

- Caja. Es un subconjunto de JavaScript orientado a la seguridad, aunque no haya sido diseñado específicamente para mashups. Al igual que JavaScript, permite una serie de operaciones, solo que éstas están definidas de manera que actúen sin violar la política del mismo origen. Estas operaciones se basan en la comunicación basada en mensajes únicamente entre objetos, por lo que no se incurriría en la falta de entablar contacto con un frame de distinto origen, sino que se están

tratando objetos individuales sin un origen concreto, entre los cuales, además, existen referencias.

- Subspace. Es una propuesta para un sistema de paso de mensajes del lado del navegador para el uso entre integradores y proveedores, con las garantías de privacidad e integridad.

- SMash. Es un framework para permitir el paso de mensajes entre frames de diferentes orígenes. Está basado en la identificación de fragmentos del mensaje.

- MashupOS. Propone un pequeño número de elementos de DOM, cada uno con nuevas restricciones de seguridad que construyen escenarios fiables.

2.1.7. Ejemplos de Mashups

Para entender mejor cómo es un mashup, se ha decidido plantear varios ejemplos de páginas Web en las que se ha utilizado esta técnica para su composición.

Para cada uno de los tipos de mashups vistos en el documento, se ha escogido un ejemplo característico.

- Mashups de mapeado.

  La página elegida es www.ChicagoCrime.org. Esta página implementa lo que se ha llamado anteriormente un mashup de mapeado, y en ella se recogen los crímenes cometidos en cada barrio de Chicago, representándolos en un mapa de la zona.

  Los datos se han obtenido de la base de datos online del Departamento de Policía de Chicago. Estos datos se mapean sobre un mapa generado gracias a la API que proporciona Google Maps. Así, los usuarios pueden ver gráficamente el índice de criminalidad de la zona deseada, cambiar esa zona, ver únicamente algunos delitos, etc.

  La página se ve de la siguiente manera:

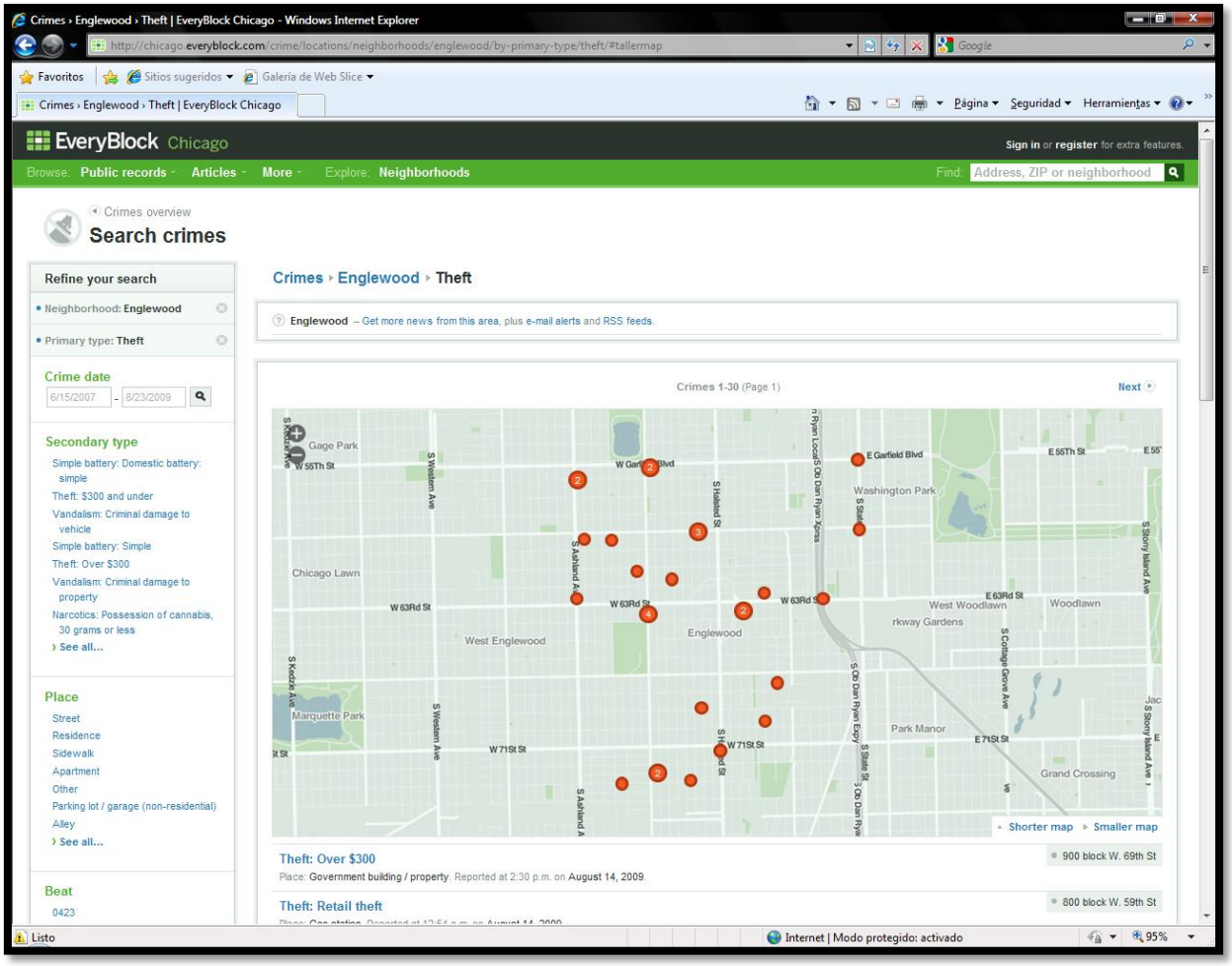

Imagen 4: Ejemplo de Mashup de mapeado (I): ChicagoCrimes.org

Hay muchos mashups de este tipo, entre los cuales encontramos el sitio Web wikiloc.com, que se llevó el primer premio de mashup generado por Google. En él se combinan los mapas de Google, cartografías del sector público y rutas memorizadas a través de GPS enviadas por los propios usuarios. Se puede ver una imagen de ella a continuación.

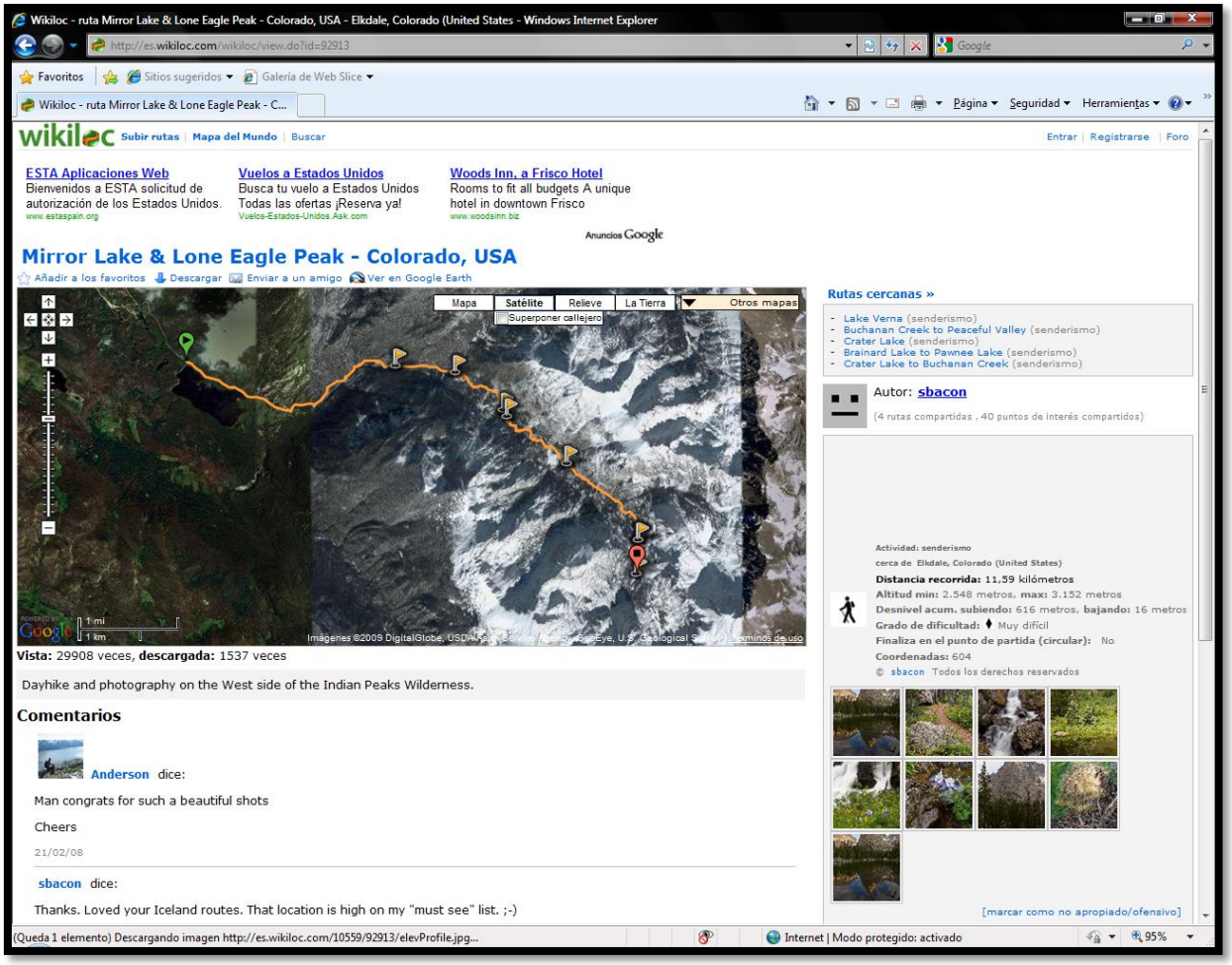

Imagen 5: Ejemplo de Mashup de mapeado (II): Wikiloc.com

- Mashups de video y fotos.

El ejemplo escogido para este tipo de mashups es la página flickrvision.com. Utiliza las páginas flikr.com y el API, nuevamente, de Google Maps. Lo que hace es mostrar continuamente los nuevos *flikrs*, es decir, imágenes, que se van añadiendo a la primera de las páginas, situándolas dentro del mapa aportado por la segunda.

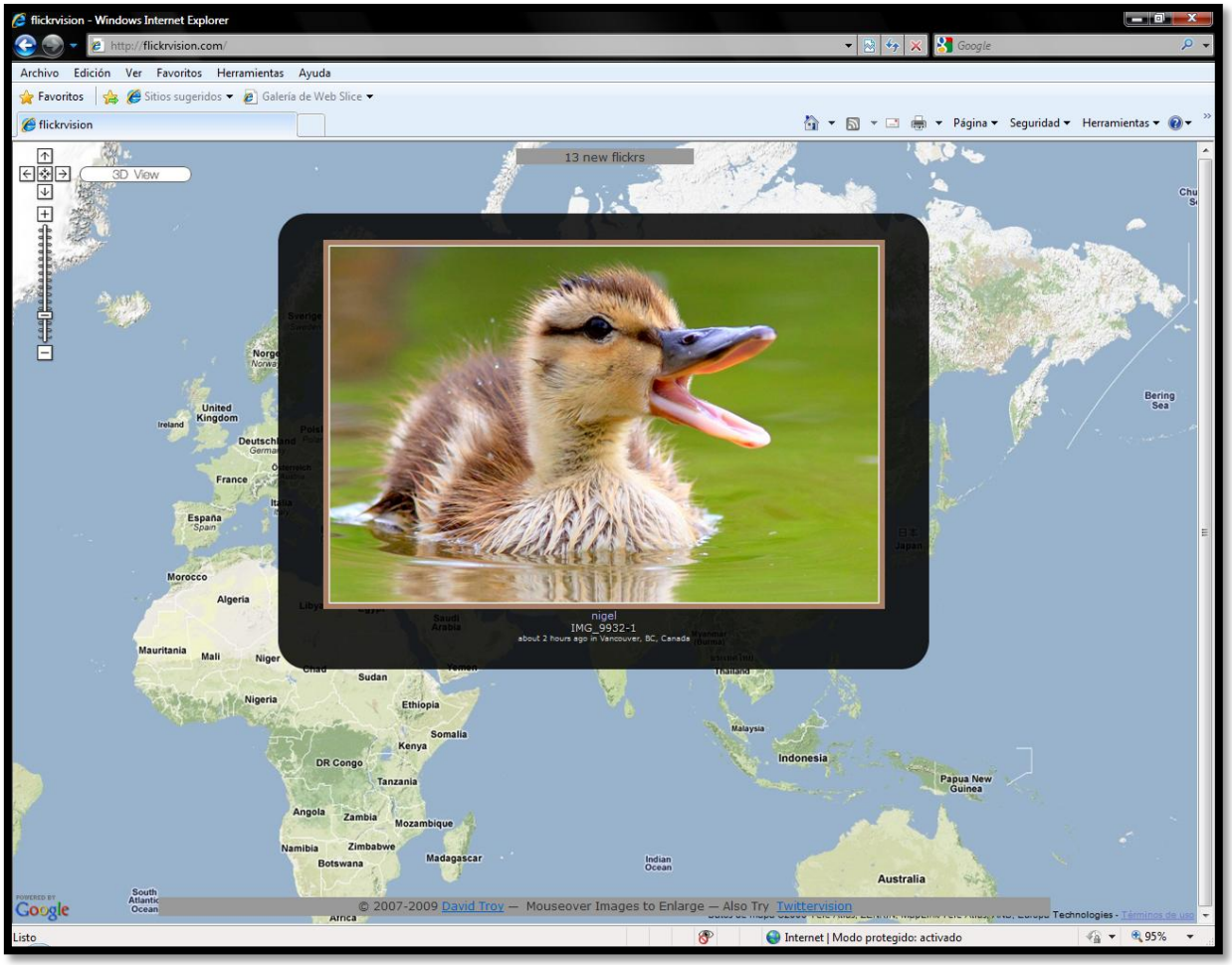

Imagen 6: Ejemplo de Mashup de fotografía y vídeo: flickrvision.com

- Mashups de búsqueda y compras.

Es muy natural hoy en día utilizar la Web para la realización de compras. Lo que ofrecen este tipo de mashups es que además de poder ver el producto, comprobar cuáles son sus características, el precio, etc., se puede hacer una comparativa entre varias ofertas, elegir las mejores promociones o comprobar cuál de los productos es el que cubre realmente las necesidades del cliente.

Una de las muchas páginas que llevan a cabo un mashup de este tipo es www.secretprices.com, la cual integra las páginas www.shooping.com y www.amozon.com.

Así es como se muestra la página:

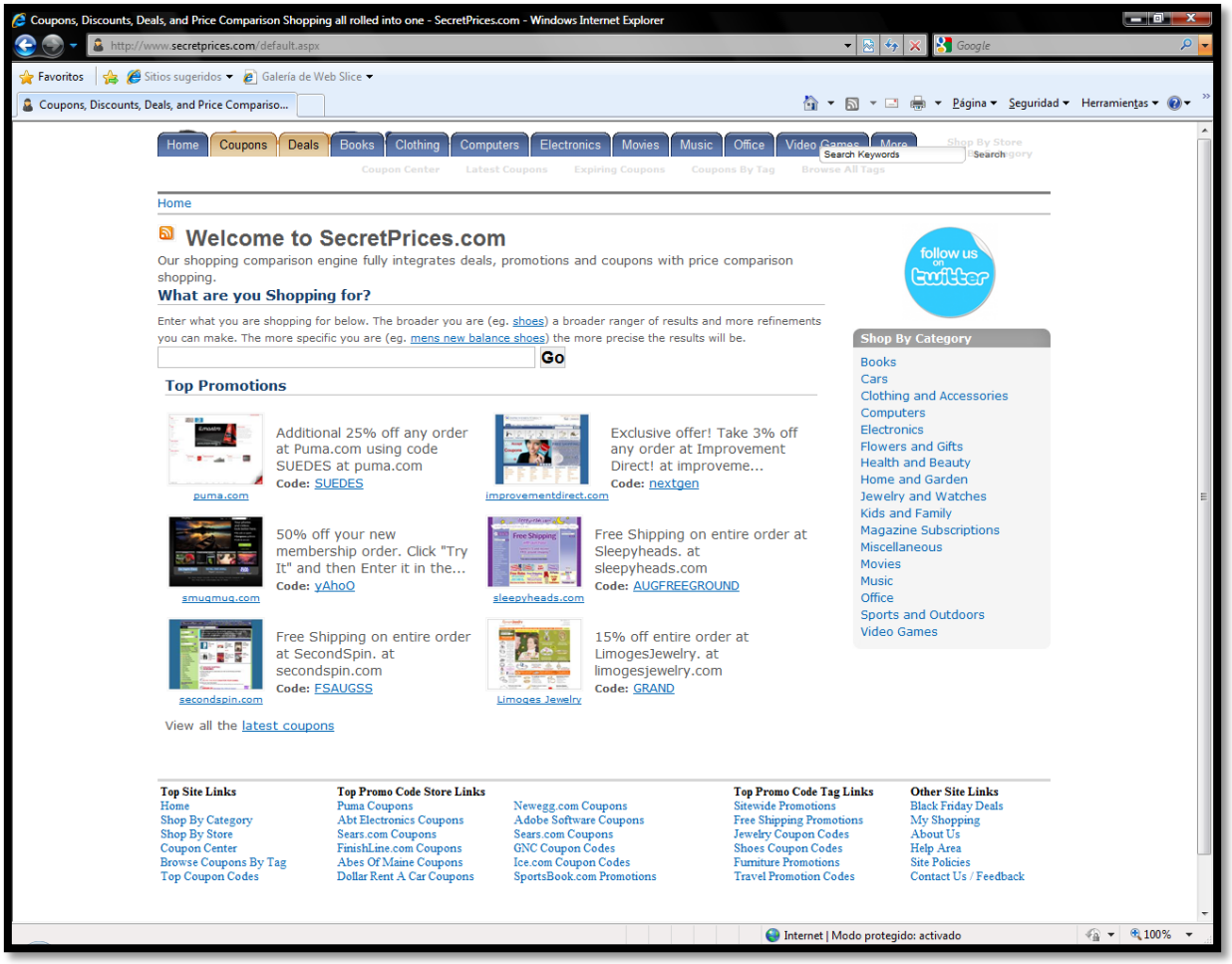

Imagen 7: Ejemplo de Mashup de búsqueda y compras: secretprices.com

- Mashups de noticias.

Cada vez se utiliza menos la prensa escrita para estar informado de lo que ocurre alrededor. Numerosas páginas son las que recopilan información de varias fuentes para ofrecer las noticias más importantes, curiosas, graciosas, etc., que salen a la luz en el mundo digital, tal y como hace la ya famosa meneame.es.

Uno de los mashups que se dedica a esta tarea es doggdot.us, la cual une las páginas digg.com, página similar a la ya mencionada menéame.es, del.icio.us, página de gestión de marcadores sociales en Web, y slashdot, sitio de noticias orientado a la tecnología. La imagen final se ve así:

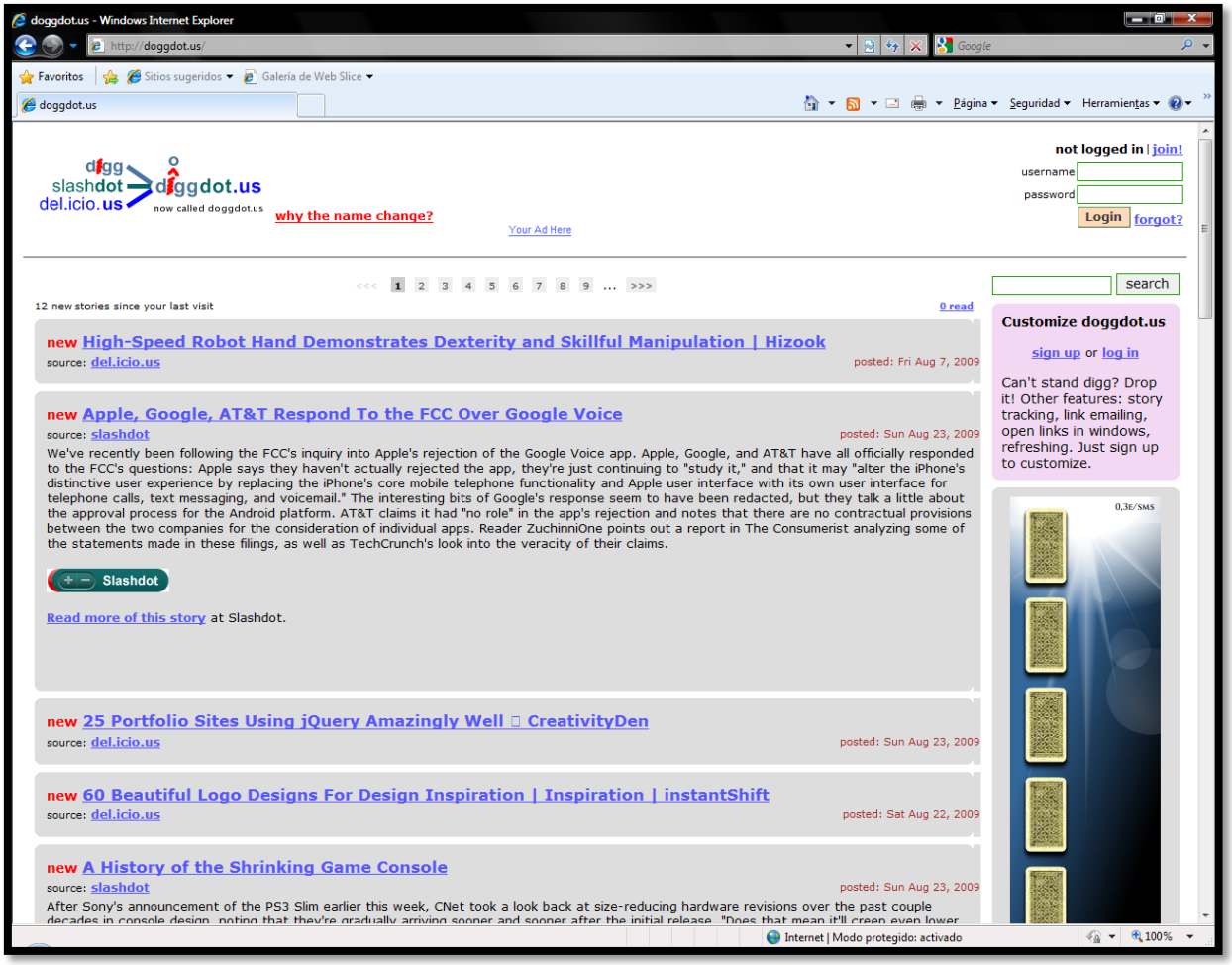

Imagen 8: Ejemplo de Mashup de noticias: doggdot.us

2.1.8. Tecnologías relacionadas

2.1.8.1. Ajax

AJAX, acrónimo de ***A**synchronous **J**avaScript **A**nd **X**ML* (JavaScript asíncrono y XML), es una técnica de desarrollo Web para crear aplicaciones interactivas o RIA (Rich Internet Applications). Estas aplicaciones se ejecutan en el cliente, es decir, en el navegador de los usuarios, mientras se mantiene la comunicación asíncrona con el servidor en segundo plano. De esta forma es posible realizar cambios sobre las páginas sin necesidad de recargarlas, lo que significa aumentar la interactividad y velocidad en las aplicaciones [10].

Ajax no es un lenguaje propiamente dicho, sino que para funcionar se hace eco de un conjunto de tecnologías que, aunadas, permiten que la comunicación entre cliente y servidor se haga de forma asíncrona.

Para poder utilizar Ajax, tan sólo se necesita un navegador que lo soporte, puesto que la lógica de la aplicación está basada en JavaScript, por lo que se ejecuta en el cliente.

Lo que hace sencillo el uso de Ajax es la creación de un objeto, el XMLHttpRequest, que permite una comunicación más fácil y directa con el servidor, al cual se le solicitan únicamente los datos que son necesarios para actualizar la página. Para transferir las peticiones y respuestas, se utilizan lenguajes de marcado, puesto que ofrecen un estándar, lo cual facilita el entendimiento entre cliente y servidor. El más usado es el lenguaje XML, aunque existen otros que son similares a éste. Para ayudar a interpretar esos datos, se utiliza DOM.

La característica más importante es la forma que tiene la página de interactuar con el cliente y el servidor. Además de ejecutarse en el primero de éstos gracias al aporte del lenguaje JavaScript, cuando un usuario necesita información del servidor, a éste no se le manda la página entera, sino que únicamente se actualiza la parte de la página que sea necesaria. El ejemplo más común es el de un formulario en el que hay varias preguntas, y cada una de las preguntas ofrece diferentes opciones en las siguientes. Mediante esta técnica, no sería necesario mandar y recibir la página entera, sino que se enviaría la respuesta de la pregunta, y se recogerían del servidor las opciones para la siguiente pregunta.

Sin embargo, esta forma de intercambiar información no queda en el simple hecho de que se transmite lo que se necesita, sino que también se hace de manera asíncrona, es decir, la ejecución no se queda parada hasta que conteste el servidor: se manda la petición y el servidor contesta cuando esté libre para poder hacerlo.

AJAX se utilizó como término por primera vez en el artículo "*Ajax: A New Approach to Web Applications*", de Jesse James [11], escrito el 18 de Febrero de 2005. Hasta ese momento la nueva vertiente de aplicaciones Web que estaba surgiendo carecía de término normalizado.

Según lo expuesto en dicho artículo: "AJAX nos es una tecnología en sí misma. En realidad, se trata de varias tecnologías independientes que se unen de formas nuevas y sorprendentes."

Las distintas tecnologías que conforman AJAX son:

- XHTML y CSS para realizar el diseño de la página, la presentación de la misma de cara al usuario, basada en estándares extendidos para la realización de dicha tarea.
- DOM, para la interacción y manipulación dinámica de la presentación.
- XML para el intercambio y la manipulación de la información. También se pueden utilizar otras tecnologías como XSLT y JSON, aunque su uso es menos extendido.
- XMLHttpRequest, para el intercambio asíncrono de información.
- JavaScript, para unir todas las demás tecnologías.

Si se realizase un esquema que agrupara a todas estas tecnologías aplicadas en AJAX, dicho diagrama sería el siguiente:

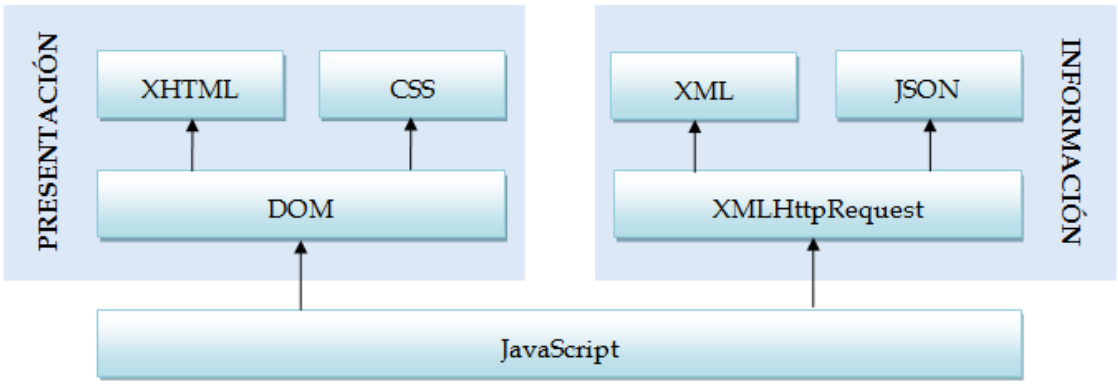

Imagen 9: Árbol de tecnologías AJAX

Como se mencionó anteriormente, cada rama de tecnologías se ocupa de una función específica, de este modo DOM, junto con XHTML y CSS se encargarían de la capa de presentación, mientras que XMLHttpRequest, junto con XML y JSON, cubrirían la capa de datos de la aplicación.

La aparición de Ajax ha sido una gran novedad a la hora de realizar páginas Web, puesto que es una visión totalmente diferente al uso que existía hasta entonces. En el siguiente gráfico se presentan las diferencias en el funcionamiento entre las aplicaciones Web tradicionales y aquellas en las que se ha introducido Ajax:

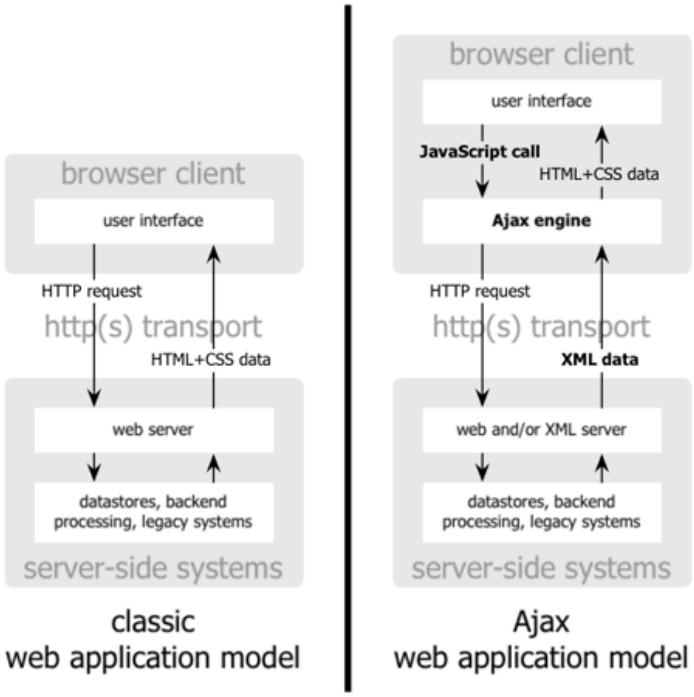

Imagen 10: Diferencia entre el modelo clásico y AJAX

Lo que Ajax pretende es solucionar la sobrecarga que se produce si se realizan muchas peticiones al servidor, provocando que el cliente soporte pesadas esperas para obtener la respuesta.

Esta recarga constante es evitada en las aplicaciones construidas con AJAX mediante la creación de un elemento intermedio entre el usuario y el servidor. La nueva capa intermedia de AJAX mejora la respuesta de la aplicación, ya que el usuario nunca se encuentra con una ventana del navegador vacía esperando la contestación del servidor. Ya no es primordial recargar la página, sino mandar y recibir la información que hace que la página vaya actualizando sólo la parte de la página que realmente necesita cambiar, aquella que está solicitando el usuario.

El siguiente esquema muestra la diferencia más importante entre una aplicación Web tradicional y una aplicación Web creada con AJAX. La imagen superior muestra la interacción síncrona propia de las aplicaciones Web tradicionales. La imagen inferior muestra la comunicación asíncrona de las aplicaciones creadas con AJAX.

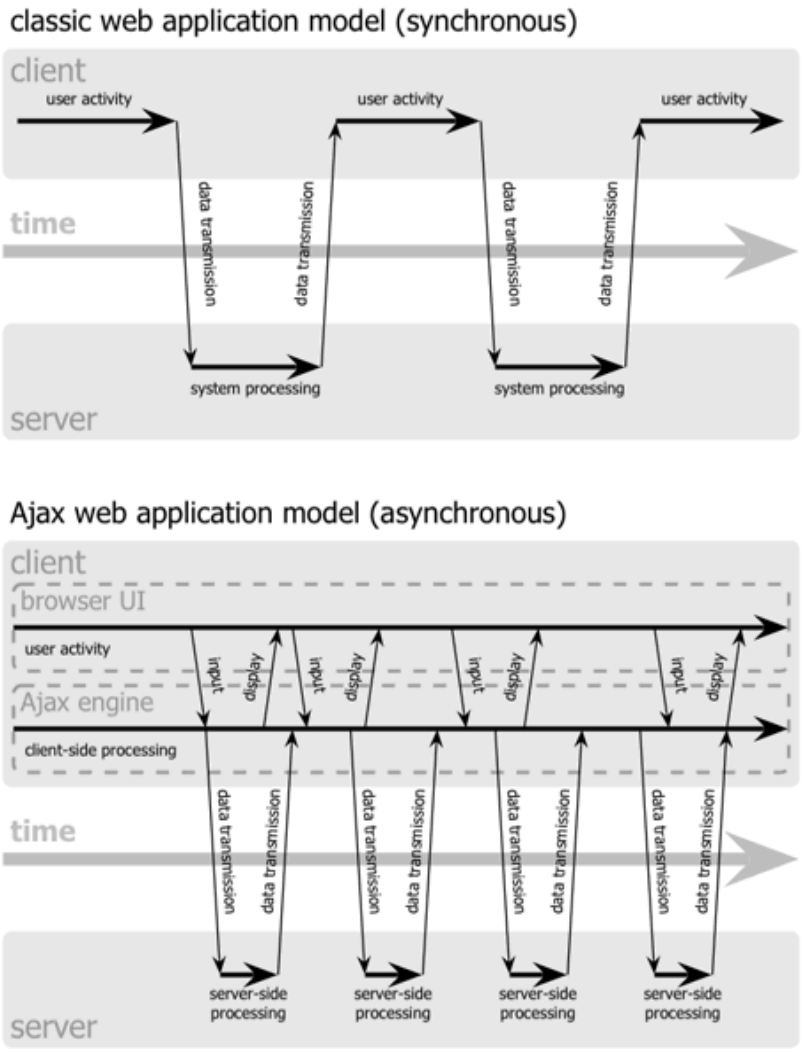

Imagen 11: Comparación entre comunicación síncrona y asíncrona

Las peticiones HTTP al servidor se sustituyen por peticiones JavaScript que se realizan al elemento encargado de AJAX. Las más simples no necesitan intervención del servidor, por lo que la respuesta es inmediata. Si la interacción requiere la participación de éste, la petición se

realiza de forma asíncrona mediante AJAX. En este caso, la actividad del usuario tampoco se ve interrumpida por recargas de página o largas esperas por la respuesta del servidor.

Desde su aparición, se han creado cientos de aplicaciones Web basadas en AJAX, y en la mayoría de los casos puede sustituir completamente a otras técnicas como Flash. Además, en el caso de las aplicaciones Web más avanzadas, pueden llegar a suplantar a las aplicaciones de escritorio.

A continuación se muestra una lista de algunas de las aplicaciones más conocidas basadas en AJAX:

- Gestores de correo electrónico: Gmail, Yahoo Mail, Windows Live Mail.
- Cartografía: Google Maps, Yahoo Maps, Windows Live Local.
- Aplicaciones Web y productividad: Google Docs, Zimbra, Zoho.
- Otras: Netvibes (metapágina), Digg (noticias), Meebo (mensajería), 30 Boxes (calendario), Flickr (fotografía).

A pesar de las comodidades que ofrece Ajax, evidentemente no se trata de una tecnología que solucione todos los problemas que surgen a la hora de realizar una aplicación Web. Tiene sus ventajas y desventajas, tanto frente a las aplicaciones de escritorio como frente a las aplicaciones Web tradicionales.

A continuación se enumeran las ventajas y desventajas de una aplicación Web general frente a una aplicación nativa.

- Ventajas:
  - No es necesario instalar una aplicación AJAX, basta con introducir la URL adecuada en un navegador para obtener su interfaz y poder comenzar a trabajar.
  - La versión disponible a través de la URL siempre es la última, con lo cual el usuario nunca tendrá que realizar actualizaciones sobre la aplicación.
  - Los programas nativos sólo pueden ejecutarse sobre un determinado sistema operativo y, en ocasiones, en varios tras instalar software adicional. Por el contrario, las aplicaciones AJAX, al ser aplicaciones Web, son independientes de la máquina, sólo precisa de un navegador para ejecutarse.
  - Las aplicaciones AJAX, como aplicaciones Web que son, tienen una forma de operar muy parecida al resto de aplicaciones Web, lo que no supone una carga inicial para el usuario que desee utilizarla.

- Una aplicación AJAX precisa de muchos menos recursos, memoria o espacio en disco que las aplicaciones nativas.
- Es más fácil migrar una aplicación AJAX a otras plataformas de lo que sería la migración de un programa nativo.

Inconvenientes:

- Una aplicación AJAX tiene un acceso limitado a los recursos locales del ordenador del usuario, en contraposición a una aplicación nativa, que puede aprovechar toda la potencia que le ofrece el sistema donde se ejecuta.
- La transmisión de datos en este tipo de aplicaciones es más lenta que la de las aplicaciones nativas, las cuales se sustentan de bases de datos locales.

A continuación se exponen las ventajas y desventajas de Ajax frente al resto de aplicaciones Web:

Ventajas:

- Para responder a las peticiones solicitadas por los usuarios, AJAX actualiza partes de las páginas que actúan como una interfaz, en lugar de renovar la página entera con el consiguiente parpadeo, típico de las aplicaciones Web clásicas.
- Cuando se obtiene información del servidor, en AJAX sólo se solicita la que es necesaria para actualizar la parte de la interfaz solicitada, en lugar de toda la necesaria para representar la página completa.
- La funcionalidad de una aplicación AJAX es similar a la de una aplicación Web que utilice applets Java o Flash, pero sin necesidad de descargar pesados programas hasta el cliente.
- La forma en que el usuario interactúa con una aplicación AJAX es mucho más fluida, más cercana a la de un programa nativo, que la que puede ofrecer una aplicación Web clásica.

Inconvenientes:

- El navegador notifica al usuario, visualmente y de manera automática, cuándo una aplicación clásica está esperando la recepción de una nueva página. Esa notificación no tiene lugar para las aplicaciones AJAX.

- Las aplicaciones AJAX carecen de una integración con los botones de navegación del explorador, lo que hace que no se pueda ni avanzar ni retroceder entre páginas.

- Para que una aplicación AJAX funcione es imprescindible que el navegador permita la ejecución de código JavaScript. Si el usuario ha desactivado el motor JavaScript del navegador las aplicaciones AJAX no funcionarán.

- Las aplicaciones Web clásicas son relativamente más fáciles de desarrollar que las aplicaciones AJAX, al no tener que ejecutar una lógica compleja en el cliente, usar DOM, etc.

Además, Ajax implica algunos problemas en cuanto a las normas de accesibilidad y usabilidad tan extendidas en los últimos años por la Web. El principal problema viene dado precisamente por la misma razón por la que es bueno utilizar Ajax: la forma de interactuar con el servidor. Al utilizar JavaScript y XMLHttpRequest, es posible que los navegadores de algunos clientes no sean capaces de ver correctamente las páginas, puesto que pueden no tener habilitados los controles que soporten estas tecnologías. Además, la respuesta a determinados eventos hace inaccesibles a ciertos usuarios algunas funcionalidades. Por ejemplo, si una acción se desencadena únicamente por hacer click con el ratón sobre un botón, si el usuario no dispone de ratón, este botón será inservible para dicho usuario. Es por esto por lo que se recomienda una alternativa que no use Ajax para aquellos aspectos susceptibles de crear problemas, aunque esto supone una mayor carga de trabajo para el programador.

Otro de los grandes inconvenientes es que los formularios basados en Ajax no actúan como el usuario está acostumbrado. En un formulario clásico uno puede hacer y deshacer a placer, sabiendo que hasta que no se pulse el botón “enviar” podemos cambiar los datos que haga falta. Con Ajax, sin embargo, no funciona así, los datos se envían campo a campo, lo que desvirtúa el concepto clásico de formulario y se introduce un elemento que genera gran perturbación y puede confundir al usuario. Además, la carga instantánea de una nueva página (que en realidad es la misma) puede producir en el usuario la sensación de que nada ha pasado, pues dicha carga ha sido muy rápida como para que pudiera percibirla y, además, la URL no ha variado. Siguiendo este mismo razonamiento, el usuario no puede utilizar los botones de navegación de los navegadores como los ha utilizado hasta el momento, puesto que al no recargar páginas enteras, éstos no guardan constancia de páginas anteriores o posteriores.

En cualquier caso, a la hora de realizar un sitio Web simplemente se tiene que tener claro cuál es la finalidad de la misma y cómo se desea hacer, valorando si el uso de Ajax es beneficioso o perjudicial en esos casos.

2.1.8.2. SOA

En la actualidad, se ha presentado un cambio a la hora de realizar las aplicaciones para una empresa y para sus áreas de negocio. Ahora son más conjuntos de sistemas que un único sistema, debido a la tendencia de hacerlos más reutilizables y modulados. Esto es debido a que en el mundo de la creación de software se ha visto la necesidad de desacoplar todo lo posible los distintos componentes, de manera que si hay que adaptar el sistema al mundo cambiante, las modificaciones sean mínimas.

Esta ventaja, sin embargo, puede llegar a suponer otros inconvenientes. Si se supone que un componente A hace uso de otro B pidiéndole cierta información que éste tiene almacenada, A debería conocer cómo guarda dicha información B, lo cual supone volver a la dependencia entre dos componentes.

En el mundo real, se producen cambios constantemente para adaptarse a las necesidades de los clientes y a las necesidades del negocio, provocando la casi obligación de utilizar nuevas arquitecturas que minimicen las dependencias entre componentes. Las empresas, además, necesitan que su organización esté conectada directamente con clientes, procesos, información tanto propia como externa, etcétera. Si sus sistemas de información no son capaces de integrar estos componentes puede suponer una importante pérdida de efectividad en sus áreas de negocio.

Es por esto que surge SOA (Service Oriented Architecture, o Arquitectura Orientada a Servicios), un paradigma que permite diseñar y construir sistemas más flexibles, escalables y utilizables. SOA es un estilo arquitectónico cuya meta es conseguir eliminar las dependencias entre los agentes software implicados en el sistema [12]. Esto quiere decir que es una arquitectura para los sistemas de información basada esencialmente en servicios.

Para entender el enfoque de SOA, es necesario entender que cada agente software se ve como un proveedor de servicios, a la vez que consumidor de éstos. Un servicio es una unidad de trabajo realizada por un proveedor de servicios para conseguir los resultados finales deseados por un consumidor de servicios [12]. En definitiva, cada agente software, o módulo del sistema, provee ciertos servicios a otros agentes, a la vez que necesitan de otros servicios proporcionados por alguno de los demás agentes del sistema.

La ventaja de dividir el trabajo en distintas unidades es que es más efectivo hacer uso de un servicio ofrecido que hacerlo uno mismo. Así, se permite que cada unidad sea especialista en unas operaciones concretas, permitiéndolas ser más específicas, y no colapsarlas teniendo que hacer todas las operaciones en una única unidad.

Hasta ahora, la construcción de los sistemas se hace dividiéndolos igualmente en módulos, aunque con la diferencia de que las interconexiones entre ellos son directas, integrando bases de datos, etc. Esta solución mejora la efectividad frente a la idea de que un mismo componente haga todo el trabajo, aunque es muy sensible a los cambios de cada una de las unidades. Como se ha comentado, la idea de SOA es desacoplar los distintos módulos, o

agentes software, lo máximo posible, eliminando las dependencias sobre ellos y permitiendo que éstos sean más robustos y flexibles al cambio. Según comenta Gebhardt, la arquitectura de software describe la estructura de un sistema, cuyo núcleo está compuesto de componentes, que gozan de características externamente visibles y de capacidades especiales para comunicarse con otros componentes. Sin embargo, todo anhelo de toda filosofía de construcción de sistemas informáticos radica en el hecho de minimizar las dependencias entre componentes [13].

SOA adopta esta definición, construyendo sistemas divididos en componentes, lo que antes se ha llamado unidades software, cada una especializada en unos servicios, y que a su vez son consumidoras y proveedoras de los mismos, haciendo que todas ellas juntas realicen la tarea esperada por el cliente. Estas unidades son independientes entre sí, así como deben ser independientes del hardware, del sistema operativo y del lenguaje de programación. Se puede decir que la estructura es la siguiente [14]:

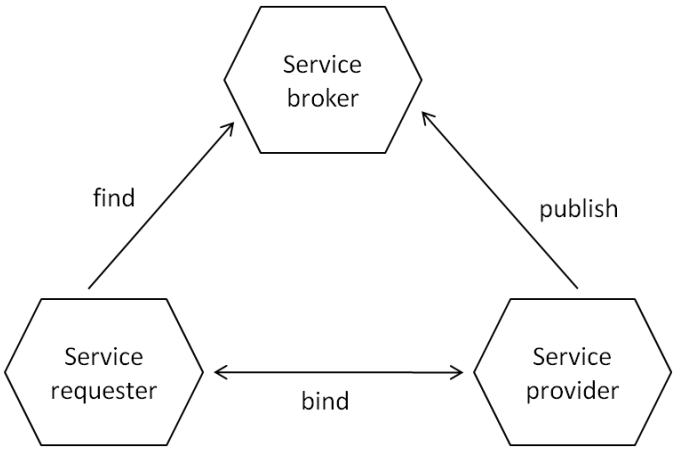

Imagen 12: Triángulo de la arquitectura orientada a servicios

El service provider, o proveedor de servicios, es aquel que ofrece un servicio cuando recibe una petición del mismo desde un consumidor. El service requester, o consumidor de servicios, es aquel que ha solicitado el servicio y por lo tanto el que lo consume. Por último, el service bróker, o registro de servicios, es aquel que facilita el consumo y registro de los servicios que son ofertados. Aquí, los servicios son descritos mediante protocolos y mensajes.

Para conseguir sus objetivos e implementar la arquitectura mostrada en la imagen anterior, se hace uso de una entidad intermedia, un middleware, que hace de intermediario entre los distintos proveedores de servicios. Su finalidad es adaptar los mensajes que recibe para que el destinatario pueda entenderlos. En el ejemplo comentado anteriormente, si el sistema A necesita pedirle información a B, en vez de pedírsela directamente, implicando conocer su funcionamiento, se lo pedirá al middleware, y éste es el que se lo solicitará a B, devolviendo posteriormente la respuesta a A, que ya no es dependiente de B. Esta estructura se puede ver más clara en la siguiente imagen [15]:

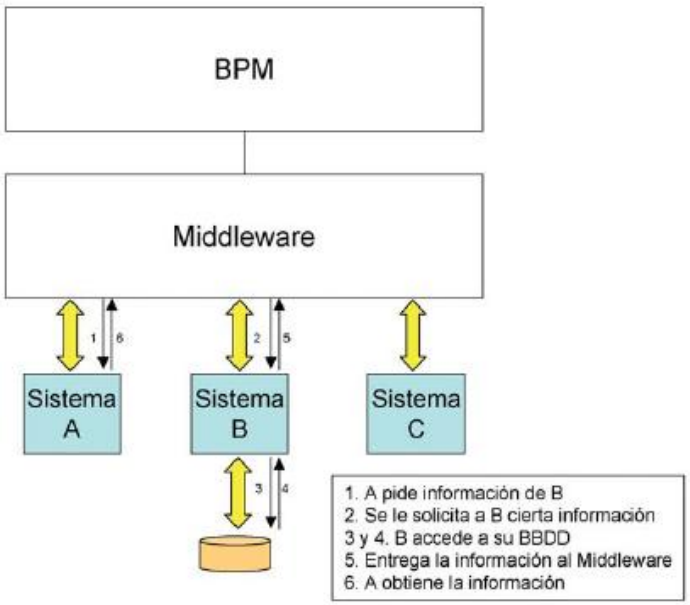

Imagen 13: Sistemas relacionados entre sí en arquitectura SOA

SOA requiere que se cumplan dos restricciones para que no haya acoplamiento entre los agentes [12]:

1. Una serie de interfaces comunes a todos los sistemas involucrados. En estas interfaces sólo está la semántica general del sistema, y deben ser accesibles por todos los proveedores y consumidores.

2. Mensajes descriptivos que deben seguir un esquema impuesto por las interfaces. El comportamiento del sistema, y la comunicación entre los agentes se debe hacer por medio de estos mensajes, cuyo contenido y estructura están limitados por el esquema propuesto. Es conveniente que se diseñe un esquema flexible y extensible, de manera que se puedan introducir nuevos servicios sin tener que modificar los ya existentes.

Definir unas buenas interfaces es algo crucial, puesto que puede hacer que el sistema sea más eficiente. De la misma forma, los mensajes deben cubrir algunas especificaciones para poder decir que siguen una arquitectura SOA:

1. Los mensajes deben ser descriptivos. Deben decir qué es lo que se necesita, no cómo se necesita conseguirlo. Si se hiciera de la segunda manera, se estaría haciendo de nuevo que los sistemas fueran dependientes unos de otros, puesto que el consumidor necesitaría saber cómo puede realizar la tarea el proveedor de servicios. Es responsabilidad del proveedor decidir cómo solventar el problema propuesto al solicitarle un servicio.

2. Es imprescindible que el mensaje siga la estructura fijada por el middleware, ya que de no ser así, el proveedor de servicios no será capaz de entender el mensaje.

3. Es esencial que la estructura de los mensajes sea extensible. No se puede garantizar que un sistema no vaya a cambiar y a ampliar las funcionalidades que ofrece. Si los mensajes no son extensibles, se provocará que se deba cambiar la estructura completa, lo que conlleva un gran cambio dentro de todas las unidades software que compongan la aplicación. El problema de cumplir esta norma es que cuanta más flexibilidad, menos restricciones se pueden llevar a cabo, las cuales también son necesarias para facilitar la comunicación entre componentes. Se deberá llegar a una solución de compromiso entre ambas.

4. Un consumidor debe ser capaz de localizar a un proveedor que le facilite el servicio que necesita. Este mecanismo puede ser muy flexible e implementarse de diferentes maneras, no tiene por qué ser un registro centralizado.

Todas las restricciones mencionadas deben ser cumplidas para poder decir que un sistema sigue una arquitectura SOA y, por tanto, que pueda beneficiarse de sus ventajas, sin embargo, existen otras condiciones que podrían incluirse en el sistema para mejorarlo [12].

1. Si cada mensaje lleva en él toda la información que necesita el proveedor para realizar el servicio pedido, no sería necesario guardar los datos de la "conversación" entre consumidor y proveedor, lo que se llama información de estado. Esta restricción mejora la visibilidad, puesto que leyendo el mensaje se puede conocer qué es lo que se está pidiendo, sin necesidad de haber visto los mensajes intercambiados anteriormente. Una consecuencia más es la capacidad de recuperarse ante fallos parciales, no teniendo que restablecer el estado entre receptor y emisor. A este tipo de servicios se les llama "servicios sin estado" o "*stateless service*".

2. El tipo de servicios anterior conlleva numerosas ventajas, sin embargo no siempre se puede carecer del estado, como ocurre, por ejemplo, en aquellas ocasiones en las que se debe realizar un intercambio de claves de seguridad para aceptar un determinado certificado. Los "servicios con estado", o "*stateful service*" necesitan que el consumidor y proveedor estén en el mismo contexto, para poder responder bien al intercambio de mensajes entre ambos. Evidentemente este tipo de servicios cargan con la desventaja de que produce un mayor acoplamiento y dependencia entre los proveedores y consumidores de los servicios, además de reducir la capacidad del primero al tener que guardar en memoria el estado de las conexiones con todos los consumidores con los que esté intercambiando mensajes.

3. La última restricción se refiere a la idempotencia de las peticiones. Esto viene a decir que si se hace varias veces una petición, el proveedor deberá responder en todas ellas con el mismo mensaje. La ventaja que proporciona esta restricción es que si ocurre algún fallo en la transmisión, recepción o tratamiento de la respuesta, el proveedor únicamente tendrá que repetir el mensaje, sin necesidad de volver a tramitar la petición.

Gracias al aumento del uso de Internet, se están extendiendo los "*servicios Web*". Un servicio Web funciona igual que los servicios explicados anteriormente, aunque con algunas modificaciones: las interfaces tienen que estar basadas en protocolos de Internet como HTTP o FTP, y los mensajes deben estar en XML, excepto aquellos que lleven datos binarios adjuntos. Este tipo de servicios se pueden realizar mediante SOAP y mediante REST, teniendo cada uno de ellos sus propias particularidades.

SOAP es un protocolo que deriva del que fue creado en 1998 por David Winer, llamado XML-RPC. El protocolo SOAP que se utiliza hoy en día lo crearon Microsoft e IBM, y su especificación está desarrollada por el W3C. SOAP define la Web como el transporte universal de mensajes. Lo que caracteriza este tipo de servicio Web es que los mensajes deben ser transportados por el protocolo SOAP, y la descripción del servicio debe estar en WSDL. El paso de mensajes necesita un entorno relativamente acotado, o al menos conocido, puesto que tanto emisor como receptor deben ser capaces de interpretar los mensajes intercambiados. Esto hace que si el entorno sufre cambios importantes, como el aumento del número de usuarios o un cambio en la forma de tratar el negocio, etc., sea inevitable un cambio radical, lo que resta flexibilidad y, por tanto, infiere en un aumento del coste para la empresa destinado a la adaptación del sistema a las nuevas necesidades.

REST, o Transferencia de Estado Representacional, es una técnica que utilizan los servicios Web basada en la utilización de la Web para transmitir los recursos necesarios entre ellos, para lo cual emplean las operaciones estándar del protocolo HTTP: GET, PUT, etc. Un recurso es cualquier pieza de información de interés albergada en la Web y, el cual, suele estar definido por tener una dirección URI concreta. Gracias a esta forma de mantener la comunicación, las empresas pueden permitirse hacer sistemas más escalables, sin miedo a tener que cambiar la estructura de los mensajes acuñados con SOAP cuando el entorno prospere y evolucione.

Una de las grandes razones por las que REST está creciendo en adeptos es porque utiliza estándares ya conocidos por todos los desarrolladores: HTTP para el intercambio de recursos entre los servicios; URL para la identificación y localización de los recursos; XML, HTML, GIF, JPEG, etc., para la representación de los recursos.

La diferencia principal con SOAP es la manera de diseñar los servicios. Cuando se quiere utilizar REST se tiene que pensar que se están mandado recursos traídos por una determinada dirección, ya estén éstos en HTML o XML, pero no llevan consigo parámetros, tal y como podría pasar con SOAP y sus mensajes.

En el mundo empresarial REST se está imponiendo a SOA, y la principal razón es la ya mencionada escalabilidad que permite. Dicha escalabilidad viene dada gracias a los principios estructurales y de diseño que adopta REST:

- Es un protocolo sin estado, cada una de las peticiones / respuestas HTTP contiene la información necesaria para que el servicio implicado pueda entender cuál es el recurso solicitado o recibido y qué hacer a continuación. Esto evita que el cliente y el servidor tengan que recordar el estado de la comunicación entre ambos,

consumiendo unos recursos que pueden necesitarse para operaciones más críticas.

- Las operaciones definidas por REST son las mismas que las definidas por HTTP, un protocolo más que asentado en la vida de la Web.
- La identificación de recursos se hace mediante un sistema universal e inequívoco de nombrado: a través de su URI.
- Permite navegar de un recurso REST a otros relacionados gracias a los enlaces de los mismos, haciendo que el cliente pueda pasar de un estado a otro fácilmente.

Es más, la efectividad de REST podría incluso mejorarse en aquellos entornos en los que los recursos no sean generados automáticamente gracias a un sistema de caché intermedio, que mejorara el rendimiento del paso de mensajes HTTP. De la misma forma se podrían interponer entre cliente y servidor proxys, puertas de enlace, etc., para mejorar la seguridad, el acceso, etc.

En cualquier caso, utilizar SOAP o REST dependerá de la situación en la que se encuentre la empresa y el área de negocio que quiera cubrir. Se podrá usar el primero cuando haya un contrato formal para la descripción de la interfaz del servicio ofrecido, cuya definición se puede hacer gracias a WSDL, y cuando se necesite mantener el estado durante las "conversaciones" entre distintos servicios Web, situación que puede acontecer en aquellos servicios que requieran transacciones con un alto nivel de seguridad.

En contraposición, los servicios REST se suelen utilizar cuando los servicios Web no tienen estado, de manera que con el mensaje HTTP se puedan realizar todas las comunicaciones necesarias. También se hace interesante su uso en aquellos dispositivos que no dispongan de unos recursos elevados, beneficiándose del escaso consumo de éstos que hace REST. Por último, aplicando su característica más importante, REST resulta especialmente valioso en servicios que necesitan ser ampliamente escalables.

Tras todo lo comentado, se pueden observar las grandes ventajas que aporta SOA [17]:

- Permite sustituir componentes individuales sin que eso afecte a otros componentes, puesto que la comunicación se hace siempre a través del middleware.
- Todos los sistemas se conectan al bus de la misma forma, con lo que se gana en homogeneidad y lo que conlleva a la reutilización.
- Facilidad en la operación y mantenimiento al ser independiente de tecnologías y productos concretos.
- Arquitectura sencilla, robusta y escalable.

- Facilidad en la adaptación de nuevos servicios y en la reestructuración de sistemas.
- Minimización de la redundancia de datos.

Pese a ello, también tiene sus desventajas, como la velocidad de transferencia de los datos o la posible sobrecarga del bus en el caso en el que se intercambien grandes volúmenes de información. Es por esto que no siempre es aconsejable la utilización de esta estructura. Hay en algunas ocasiones en las que es totalmente desaconsejable la aplicación de SOA [16]:

- Cuando se tienen un ambiente TI homogéneo. Se produce cuando se utiliza la tecnología de un único proveedor y, en este caso, es innecesario soportar la sobrecarga que produce la instalación de un SOA.
- Cuando se tienen una aplicación en la que es crítico que actúe en tiempo real. En este caso no es aconsejable su utilización puesto que no se puede garantizar que la comunicación entre consumidor y proveedor sea lo suficientemente rápida como para cubrir las necesidades impuestas por el sistema.
- Cuando no se prevé que vayan a ocurrir cambios. En estos casos, la inclusión de un SOA podría no merecer la pena, puesto que las ventajas que ofrece se verían cubiertas ante la perspectiva de encontrarse ante un sistema estable.
- Cuando la pérdida de dependencia es una desventaja, no una ventaja. Ocurre en algunos sistemas, cuando éstos se ejecutan sobre una única máquina y la introducción de SOA trae consigo una carga innecesaria de trabajo.

En las empresas, sin embargo, estas situaciones no suelen darse, y se está viendo un gran beneficio a la hora de realizar las aplicaciones siguiendo estas directrices. Es por esto que cada vez se ven en más empresas.

2.1.8.3. Screen scraping

Cuando se habla de extracción de datos o información en general de una página Web, lo primero que se viene a la cabeza son las útiles APIs o técnicas como REST o RSS. Sin embargo, hay portales que no dan acceso a su contenido tan fácilmente, por lo que hay que aplicar otras estrategias.

Una de las opciones disponibles es Screen scraping, una técnica que usa herramientas software para parsear y analizar el contenido de la página Web deseada, en caso de mashups, o de un documento deseado en general, pudiendo contener tanto texto como imágenes.

El objetivo es extraer las estructuras de datos semánticos de esa información para que pueda ser usada y manipulada por medio de instrucciones programadas. Si se está hablando

de mashups, exploraría mediante esta técnica, también llamada Web scraping, la página Web que utilice como fuente para poder construir su propio contenido.

Hay diferentes niveles de automatización dentro del scraping:

- Para sitios Web en los que el acceso a la información es especialmente complicado e intratable, tiene que entrar la mano humana, que será el encargado de "copiar y pegar" los datos importantes que en ella aparezca.
- Una técnica simple es la basada en el comando "grep" de UNIX, el cual extrae todas las palabras o conjunto de ellas que coinciden con una expresión regular dada.
- Para páginas tanto estáticas como dinámicas es posible recuperar la información haciendo peticiones vía HTTP mediante programación con sockets.
- Algunos navegadores Web, como Internet Explorer o Mozilla Firefox, recuperan la información de las páginas construyendo árboles DOM, repartiendo en sus nodos la estructura que sigue.
- Existen lenguajes de consulta semiestructurados, como XQL (XML Query Language), que parsean las páginas HTML para recuperar su contenido.
- Se ha creado software específico para este fin, son herramientas propias de Web scraping.
- Cada vez más, las páginas Web incluyen metainformación sobre su contenido incluyendo, por ejemplo, microformatos. A través de éstos se puede acceder a fragmentos de datos específicos, pudiendo recuperarse para su uso.

Esta técnica, sin embargo, tiene algunos inconvenientes importantes: intenta acceder, recuperar y manipular información desestructurada, que no tiene por qué estar siempre dispuesta de la misma manera; obliga a que el desarrollador que desee acceder a la misma se adapte especialmente a la organización seguida por la página, teniendo que adaptarla si quiere acceder a otra distinta o si ésta cambia, cosa que ocurre muy a menudo al querer el dueño que su sitio Web se mantenga actualizado.

Otro de los problemas que plantea es la escasez de herramientas software que existen para esta tecnología, siendo las que hay poco sofisticadas y reusables.

2.1.8.4. RDF

RDF, del inglés Resource Description Framework, o en español Marco de Descripción de Recursos, fue creado en 1999 por Ramanathan V. Guha. Es un framework que sirve para describir e intercambiar metadatos asociados a los recursos, siendo dichos metadatos información sobre el propio recurso que permite añadir significado a las páginas y otorgarles información semántica [20] [21].

Los recursos tienen asociados propiedades que definen sus características, siendo éstas propiedades a su vez recursos, como por ejemplo el título de una página Web. La clave de RDF consiste en especificar cada propiedad de los recursos en tripletes de la forma sujeto – predicado – objeto. El sujeto se refiere al recurso que se está describiendo; el predicado es la propiedad, es decir, lo que se está especificando del recurso; y el objeto es el valor de la propiedad.

La información que proporciona RDF es usada normalmente por las propias máquinas. Un posible caso sería para recabar información de la misma por parte del índice de un buscador para, por ejemplo, saber cuál es la temática de la página y mostrarla cuando el usuario esté interesado en ese tema en concreto. Otro escenario en el que es útil es en mashups, para que éste recoja los datos necesarios de la página y pueda mostrarlos en el sitio Web final.

Según el W3C, el modelo de RDF se limita a cuatro reglas:

1. Un hecho está expresado como una tripleta Sujeto – Predicado – Objeto, conocido también como sentencia.

2. Sujetos, predicados y objetos vienen dados como nombres de entidades, también llamados recursos. Las entidades representan una persona, un sitio Web, o algo más abstracto como estados y relaciones.

3. Los nombres son URIs que tienen un alcance global, siempre refiriéndose a la misma entidad en cualquier documento RDF en el que aparezcan.

4. Los objetos pueden ser valores textuales, llamados valores literales, que pueden ser o no descritos usando los tipos de datos de XML Schema.

RDF, además, tiene las siguientes características:

- Las propiedades pueden ser inventadas por cualquiera, no están predefinidas.
- Las sentencias RDF pueden ser escritas en XML, por lo que se podrían usar para el intercambio de información.
- La simplicidad de las sentencias, únicamente tres campos, hace que RDF sea fácilmente manejable aunque aumente el número de este tipo de documentos en la red.

- Las propiedades son también recursos, por lo que pueden tener más propiedades asociadas que la describan.
- Los objetos pueden ser valores literales u otros recursos.
- Las sentencias pueden ser recursos y tener sus propias propiedades.

Existen dos formas de escribir RDF, mediante un formato similar a XML y mediante la llamada Notation 3. Para el intercambio de información, es la primera la que se usa.

Un ejemplo de cómo sería es el siguiente:

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:ex="http://www.example.org/">

    <rdf:Description
     rdf:about="http://www.example.org/vincent_donofrio">
        <ex:starred_in>
            <ex:tv_show
              rdf:about="http://www.example.org/law_and_order_ci" />
        </ex:starred_in>
    </rdf:Description>

    <rdf:Description
      rdf:about="http://www.example.org/the_thirteenth_floor">
        <ex:similar_plot_as
          rdf:resource="http://www.example.org/the_matrix" />
    </rdf:Description>

</rdf:RDF>
```

Este fragmento se correspondería con la tabla descrita a continuación:

Sujeto	Predicado	Objeto
vincent_donofrio	starred_in	law_&_order_ci
law_&_order_ci	is_a	tv_show
the_thirteenth_floor	similar_plot_as	the_matrix

RDF tiene dos tipos de nodos: los que representan a los recursos y, por tanto, sujetos y objetos, y los que representan las propiedades. La etiqueta “rdf:about” es propia de los primeros y a través de ella se especifica la URI del recurso. Estos nodos son los que contienen a los nodos de propiedades, los cuales a su vez pueden contener un valor literal especificado

entre comillas, un recurso etiquetado mediante “rdf:resource”, o un nodo completo con otro recurso.

2.1.8.5. RSS y ATOM

RSS y Atom son documentos con formato XML que sirven para la distribución de los contenidos de los sitios Web a los que pertenecen, comúnmente llamado por el sector sindicación Web. Gracias a ellos otras páginas Web pueden leer el contenido de la primera e incluir ciertas partes del mismo.

Estos archivos tienen el nombre de feeds y es común verlos en páginas sobre noticias o foros, páginas que se actualizan a menudo y cuyas modificaciones provocan cierto interés en los usuarios. Los programas que se encargan de leer estos archivos se llaman agregadores o lectores de feeds, e incluyen una nueva página cuando el usuario se suscribe a ese canal de feed. La cualidad más importante de éstos es que el usuario, conectándose a su programa agregador, puede ver, por ejemplo, las noticias de todas las páginas a las que se ha suscrito en una sola, mostrándole las últimas actualizaciones de éstas.

Ambos formatos, RSS y Atom, tienen la misma finalidad, aunque el segundo surgió como alternativa al primero en el año 2005. Las diferencias entre ambos son técnicas, siendo Atom más completo. Algunos ejemplos de las ventajas que ofrece son las siguientes:

- Indica el tipo de contenidos que envía.
- Especifica si se envía sólo el titular del texto con un enlace o bien el pasaje íntegro.
- Atom, además, permite la exportación de contenidos generados de un sitio Web a otro, muy útil para la creación de mashups.
- Atom permite que se agreguen varias fuentes en un mismo contenido, conservando la información de su creador y manteniendo enlaces a la información original.

A partir de Atom ha surgido hAtom, un microformato válido con el navegador Internet Explorer que provoca que surjan ventanas emergentes con la información actualizada de los sitios Web a los que el usuario esté suscrito.

A pesar de que Atom se está extendiendo cada vez más, es un hecho que hoy por hoy el más utilizado es RSS, aun con las ventajas añadidas que presenta el nuevo sublenguaje.

Un ejemplo de lo que sería un documento RSS es el siguiente:

```
<?xml version="1.0"?>
<rss version="2.0">

  <channel>
    <title>Lift Off News</title>
```

```
    <link>http://liftoff.msfc.nasa.gov/</link>
    <description>Liftoff to Space Exploration.</description>
    <language>en-us</language>
    <pubDate>Tue, 10 Jun 2003 04:00:00 GMT</pubDate>
    <lastBuildDate>Tue, 10 Jun 2003 09:41:01 GMT</lastBuildDate>
    <docs>http://blogs.law.harvard.edu/tech/rss</docs>
    <generator>Weblog Editor 2.0</generator>
    <managingEditor>editor@example.com</managingEditor>
    <webMaster>webmaster@example.com</webMaster>
    <ttl>5</ttl>

    <item>
      <title>Star City</title>
      <link>
         http://liftoff.msfc.nasa.gov/news/2003/news-starcity.asp
      </link>
      <description>
        How do Americans get ready to work with Russians aboard the
        International Space Station? They take a crash course in
        culture, language and protocol at Russia's Star City.
      </description>
      <pubDate>Tue, 03 Jun 2003 09:39:21 GMT</pubDate>
      <guid>
        http://liftoff.msfc.nasa.gov/2003/06/03.html#item573
      </guid>
    </item>

  </channel>
</rss>
```

2.2. Entornos virtuales

2.2.1. Introducción

Un mundo virtual es un entorno simulado, normalmente en el ordenador, en el que los usuarios se introducen identificándose a sí mismos por medio de un personaje o avatar, e interactúan en dicho medio como si del mundo real se tratara. La interacción con ese mundo puede ser textual, en dos dimensiones o en tres dimensiones, además de poder aceptar o no múltiples usuarios. El mundo que se represente, a su vez, puede ser una copia exacta del mundo real o algo fantástico, aunque normalmente las reglas generales que se aplican son las mismas que las de la realidad.

Cuando se habla de realidad virtual, o de entornos o mundos virtuales, siempre se recuerda en primera instancia la famosa película Matrix, en la que la mayoría de la gente vivía en un mundo inventado pensando que se trataba del real.

Hoy en día, las personas utilizan estos mundos virtuales como complemento a la vida que ya llevan en la realidad, ya sea para realizar compras, hacer nuevas amistades, aprender o jugar. Las limitaciones en estos entornos no existen, y poco a poco van apareciendo más para cubrir cualquier necesidad que pueda surgir entre la población.

En los años sesenta se diseñó lo que podría llamarse en cierta forma la primera máquina, o más bien prototipo, de realidad virtual, llamada *Sensorama*. El artífice de dicha máquina fue Morton Heilig, y lo que pretendía era construir un artilugio que permitiera a aquel que la utilizara sumergirse en un mundo de imágenes tridimensionales, olores, sonidos, vibraciones, hasta tal punto que esa persona pudiera sentir que realmente estaba en el lugar que la máquina le mostraba. La imagen siguiente muestra el prototipo de Sensorama:

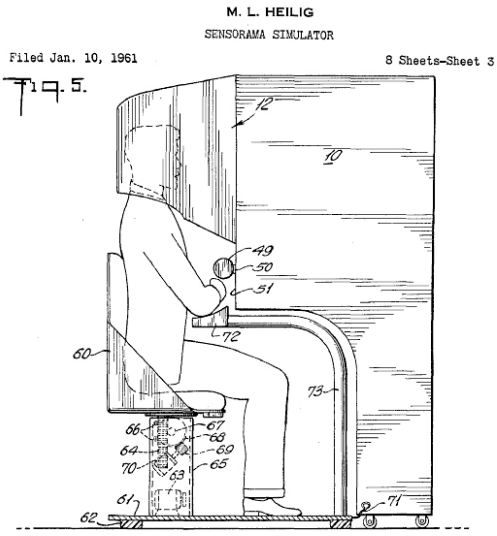

Imagen 14: Prototipo de Sensorama

Por esa misma época, pocos años después, Ivan Sutherland presentó en un congreso el concepto de *Mundo Virtual* como tal. Según sus propias palabras: "*Queremos usar todos los canales para comunicarnos con el ser humano que la mente ya sabe interpretar*". Se refería a la importancia de interactuar en los mundos virtuales tal y como se hace con el mundo real, percibiendo la nueva realidad con todos los sentidos, incluido el tacto.

La forma actual de crear mundos o entornos virtuales es a través de los ordenadores, quedando las ideas de Sutherland y Heilig ligeramente aparcadas, al buscar una experiencia mental y emocional dentro de la realidad inventada, más que una percepción sensorial física.

Siguiendo esta nueva línea de realidad virtual, los primeros en alcanzar estos objetivos fueron los juegos de ordenador. Maze War fue un juego de ordenador de principios de los años setenta, y es considerado el primer juego de disparos en tres dimensiones en primera persona, además de ser online en Arpanet. Los jugadores, con forma de globo ocular, iban recorriendo un laberinto. Cada vez que se encontraban con otro jugador y le veían, se consideraba que le había disparado, ganando puntos el jugador que disparaba, perdiéndolos el otro.

Tan sólo cuatro años después, en 1978, surgió un nuevo juego a través de Internet, en este caso de role. A pesar de ser posterior, éste no incluía tecnología de tres dimensiones, sino que bastaba con un terminal básico, accediendo al servidor donde se alojaba el juego a través de Telnet. Los cambios en el mundo, las decisiones y acciones tomadas por el jugador, la información de su personaje, etc., se enviaban y recibían a través de texto. Este juego tampoco podría decirse que creaba un mundo virtual como se concibe ahora, precisamente por la falta de la visión del mundo en sí, aunque sí conserva los preceptos de incluir al usuario en un mundo no real e interactuar en él con el entorno.

En los años ochenta LucasFilmGames despuntó y creó Habitat para el ordenador Commodore 64, un juego multijugador online en tres dimensiones que fue el percusor de muchos otros. WorldsAway siguió su estela, convirtiéndose en un chat virtual que vio la luz en el Internet que hoy conocemos en el año 1997.

Sin embargo, destaca la original idea de crear ciudades enteras digitales. Estas ciudades son un lugar, dentro de Internet, en el que las personas de una zona pueden interactuar, compartir información, intercambiar experiencias, etc. Una de las primeras en convertirse en ciudad virtual fue Ámsterdam, aunque una de las más famosas es la ciudad de Helsinki, con el proyecto Helsinki Arena 2000. Este proyecto aprovecha la red metropolitana de comunicaciones para que sus habitantes puedan, por ejemplo, comunicarse mediante videollamadas en tiempo real, o ver en tres dimensiones los números de teléfono de cada edificio y usarlos para llamarlos con sólo pulsar encima. Lo que más llama la atención es el Virtuaali Museo, que aporta la posibilidad de visitar la ciudad en dos y tres dimensiones. La visita cuenta la historia de la Helsinki, mostrando cómo era a lo largo de los siglos, pudiendo ver incluso edificios ya destruidos por el paso del tiempo.

2.2.2. La vida en los mundos virtuales

Si se acercase el mundo virtual realmente al mundo real, éste debería estar en funcionamiento todos los días a todas horas, siendo capaz de albergar en cualquier momento a un número indeterminado de personas. Esto apenas es posible en mundos pequeños, aunque es realmente irrealizable en mundos online que albergan cierta cantidad de gente. En éstos, se hace necesario “parar” el mundo para realizar las actualizaciones pertinentes.

Otra particularidad es el tiempo. Sería lógico pensar que éste debería pasar igual en ambos mundos, aunque es común que vaya más deprisa en el digital, sobre todo en los juegos, en los que ver pasar el día tal y como se ve en la realidad podría llegar a ser tedioso.

Cuando alguien entra en un mundo virtual, se adapta a las características de ese mundo, sus leyes físicas, sus reglas, etc., aunque realmente se lleva parte del mundo real, su manera de pensar, su actitud. Su comportamiento en la realidad afectará en cierta medida su comportamiento en la vida artificial.

Dejando aparte si el mundo es de fantasía o imita a la realidad, hay ciertos aspectos que siempre irán ligados a cualquiera de ellos. Uno de estos aspectos es la economía. En cualquier mundo, ya sean juegos o no, la venta de artículos se lleva a cabo. Si son juegos, los objetos con los que puede equiparse un personaje se venden y se compran, bien a personajes creados en el propio juego o bien a otros jugadores que se encuentren en la misma realidad. Al igual que en el mundo real, se desarrolla toda una economía dentro de esos mundos. Los objetos tienen más o menos valor dependiendo de su utilidad y de la cantidad de los mismos que haya, encareciendo su precio cuanto más difíciles de encontrar sean.

El valor de la propiedad en los mundos virtuales ha llegado incluso a expandirse en el mundo real, pudiendo encontrar gente que vende objetos digitales en tiendas electrónicas del mundo real, como eBay, a pesar incluso de existir una ley que indica que las propiedades virtuales carecen de valor y que realmente los que las poseen en el mundo virtual no tienen derecho de propiedad sobre ellas.

Al igual que la economía del mundo virtual entra en el real, también ocurre lo contrario. Algunas industrias contemplan introducir publicidad en aquellos entornos virtuales tan visitados, viendo un gran filón al conocer la gran cantidad de personas que los frecuentan, y el dinero que se podrían ahorrar en caras campañas de publicidad. Otras compañías, como la propia Apple, han creado su propia tienda dentro del mundo virtual *Second Life* para ofrecer a los usuarios información sobre sus productos. El uso de esta nueva tecnología les permite, además, ver cómo reacciona la gente ante los productos mostrados y si éstos tendrán una buena acogida, una información importante para focalizar un artículo a personas con unas características específicas o para comprobar si se necesita mejorarlo antes de lanzarlo a nivel global.

La interacción entre los avatares se hace tal y como se haría en la vida real: desde relaciones simplemente comerciales, hasta encuentros amistosos o incluso amorosos en algunos de los mundos existentes, como el ya mencionado *Second Life*.

2.2.3. Mundos virtuales y la psicología

Es notable el aumento del uso de Internet entre la población, incluyendo el uso de mundos virtuales en todas sus variantes: juegos de role online multijugador, salas de chats, foros, blogs, etc. En estos mundos, al no ser reales, la gente tiene unas reacciones diferentes que si interactuaran con personas cara a cara. Debido a esto, la psicología se ha interesado por el comportamiento de la población dentro de la realidad virtual creada en estos entornos, buscando además la manera de ayudar a la gente utilizando las nuevas tecnologías puestas a su disposición.

Los mundos virtuales más extendidos son aquellos formados por juegos online multijugador, en el que cada persona maneja su propio avatar y se comunica con otras personas reales a través de sus respectivos personajes. La profesora de sociología Sherry

Turkle comenta que "*el mundo de la simulación es el nuevo escenario para jugar con nuestras fantasías, tanto emocionales como intelectuales*". En ellos, las personas actúan como les gustaría comportarse en los escenarios propuestos, que no siempre coincide con las reacciones que tendrían en la realidad. Griffith y colaboradores suyos corroboran esta idea diciendo: "*Con la llegada de los nuevos mundos virtuales online, existe la oportunidad, tanto de explorar la psicología de los jugadores que encaja con esta nueva forma de ocio, como también la psicología de los jugadores dentro del mismo juego*".

Si queda alguna duda de lo atrayentes que son los juegos basados en mundos virtuales, se puede ver el ejemplo de *Los Sims*, un juego de simulación en el que el usuario crea y maneja a una serie de personajes en su vida diaria, teniendo éstos que mantener un equilibrio perfecto entre su trabajo y sus relaciones interpersonales.

Un posible motivo del uso de este tipo de juegos es que las personas que se sienten solas y, en este ambiente, parecen sentirse algo más acompañadas, incluso cuando el objeto de su interacción no sean personas reales, como es en el caso de *Los Sims*. Sin embargo, la gente se introduce mentalmente en el juego, hasta el punto de sentirse dentro de él, como si éste, en cierta manera, fuera real.

Félix Moral ha escrito un artículo que se ocupa del aspecto social dentro de las relaciones a través de Internet [22], dando énfasis al hecho de que la mayoría de las personas usa Internet en su hogar para mantener contacto con otras personas. En él, además, enumera numerosos estudios que hablan de los resultados observados en estos escenarios.

Uno de ellos, el de McKenna y Bargh, plantea interesantes cuestiones sobre la utilización de Internet como medio de socialización: por qué se usa y qué consecuencias tiene en su vida social y en sí mismo. Respecto al por qué, una de las razones sería la falta de tiempo para conocer gente: es más fácil y rápido hacerlo desde casa. También se argumentan motivaciones de soledad y angustia social, causada por el estrés de conocer nuevas personas y enfrentarse a ellas en persona. Si se utilizan los mundos virtuales para entablar estas relaciones, el anonimato proporcionado puede hacer que esas personas se sientan más seguras de sí mismas, pudiendo evitar, al menos en parte, la angustia que sienten. Siguiendo esta línea, Turkle incluye que las personas con algún rasgo físico o psíquico que provoque falta de autoestima, o falta de aceptación por parte de la sociedad, encontrarían en los mundos virtuales una vía de escape y una manera de conocer a gente con la que se sientan identificados, disminuyendo su sentimiento de soledad y pudiendo mostrar su propia personalidad. Se puede aprovechar este anonimato, por otra parte, para crear una versión de uno mismo distinta a la realidad, probar determinados comportamientos que no se harían de normal para ver sus resultados, sabiendo que en cualquier momento se puede abandonar y volver a la realidad.

Para responder a la segunda pregunta, qué consecuencias conlleva, hay más diversidad de opiniones. Hay quienes afirman que es perjudicial y que las personas con depresión o sentimiento de soledad que han usado estos medios han sufrido recaídas. McKenna y Bargh hicieron un estudio para constatar la veracidad de estas afirmaciones, encontrando que no

tenían razón, únicamente un pequeño porcentaje sentía aumentar su grado de depresión. El resto, en aquellos que se sentían solos se repartía casi igual entre no sufrir cambio alguno o sentir una mejoría, y en los que sufrían depresión, la mayoría no expresó ningún efecto de su uso de Internet.

Como consecuencias individuales, tanto Turkle como McKenna y Bargh sostienen que las personas con personalidad estigmatizada dentro de la sociedad se sienten menos solas y más comprendidas, lo que supone que aumenten su autoaceptación y se valoren mejor a sí mismas.

Otro de los artículos habla sobre el uso de la realidad virtual en pacientes con trastornos, fobias, y diversos problemas psicológicos [23], así como varios estudios en los que se investigan los resultados obtenidos con pacientes reales. Uno de estos problemas tratados son los trastornos de ansiedad. El tratamiento que se suele utilizar en estos casos es la exposición del paciente a aquella situación que le produce esa angustia. Recrear esa imagen ahora se hace más sencillo utilizando mundos virtuales a través de la realidad virtual. En ella el médico puede simular de manera convincente la situación deseada, pudiendo avanzar poco a poco y metiendo a la persona en diferentes escenarios en un mismo día sin moverse de la consulta. El paciente, en estos casos, se muestra más abierto al no sentirse expuesto tan públicamente, a la vez que aprecia las circunstancias que le rodean de manera suficientemente real como para que le sirva para enfrentarse a ellas. Los trastornos alimentarios, pudiendo exponer al enfermo ante su imagen corporal, es otra de las posibilidades que ofrecen estos mundos, así como situaciones en las que el afectado sufre alguna adicción.

Enfermedades sociales como el autismo entrarían dentro de estas posibles situaciones en las que un mundo virtual podría ayudar, al no tener éstos que enfrentarse directamente con personas. Enfermos incapacitados, de movilidad reducida, niños hospitalizados, etc., también pueden encontrar una mejoría al utilizarlos, siendo una distracción para los mismos, en los cuales además no tienen la carga que soportan en la vida real, pudiendo hacer cualquier actividad que deseen.

2.2.4. Mundos virtuales para la enseñanza

Los mundos virtuales sirven para conectar personas en el mundo entero, haciendo parecer que coexisten en un mismo entorno. Pero esta tecnología puede usarse para muchas más cosas, entre ellas la enseñanza. Los profesores deben sacar partido a todos los medios disponibles para transmitir sus conocimientos a sus alumnos, incluyendo los avances tecnológicos. Los mundos virtuales se pueden usar como complemento a clases presenciales o bien como ayuda para aquellos alumnos que no puedan acudir regularmente a un edificio docente.

Los mundos virtuales orientados a la enseñanza se caracterizan por acoger estudiantes y recrear de alguna manera lo que se desea enseñar, transmitiendo los conocimientos de las

diferentes lecciones a través de complementos de texto y audiovisuales. Aquellos que admiten a varios usuarios simultáneamente, además, promueven la colaboración y el trabajo en equipo, siendo imprescindible, además, la figura de un profesor o tutor que pueda guiarlos siempre que lo necesiten.

Para que los profesores puedan educar correctamente a través de este medio, es necesario que tengan los conocimientos necesarios sobre la tecnología apropiada y que adapten la forma de dar las clases al nuevo entorno. La preparación de los mismos, así como sus clases, deben alcanzar una serie de puntos [24].

- Formación del profesorado para el uso crítico de las nuevas tecnologías.
- Desarrollar la motivación en el usuario.
- Aprendizaje de situaciones reales.
- Diseño de modelos de experimentación.
- Realización de propuestas didácticas en el aula.
- Colaboración con centros educativos.

Es evidente que la información no llega al alumno de la misma forma que si fuera cara a cara con el profesor en una clase. En el mundo virtual se cuentan con más medios, aunque se pierde el contacto humano que puede hacer crear un mayor interés del alumno por lo que intenta aprender. Sin embargo, es necesario que el estudiante pueda acceder a todo el contenido pedagógico que necesite, estando éste estructurado de manera que se adapte al marco tecnológico en el que se halla.

Javier Onrubia [25] habla del aprendizaje como un proceso de construcción que hace el alumno para asimilar el temario, definiéndolo como un proceso de reconstrucción del mismo en su mente, adecuado a su manera de pensar. Habla, siguiendo esta conclusión, del postulado constructivista, y de sus dos implicaciones más importantes. La primera es que el contenido debe tener dos diferenciaciones en cuanto a su estructura, pudiendo dividirse en su "estructura lógica" y su estructura física. Según sus propias palabras: "*La estructura lógica de un contenido remite a la organización interna del material de aprendizaje en sí mismo, y puede considerarse estable entre contextos, situaciones y aprendices. La estructura psicológica del contenido, en cambio, remite a la organización de ese material para un alumno concreto, y depende de lo que, en cada momento, el alumno aporta al proceso de aprendizaje. […] Por un lado está la significatividad lógica, relacionada con la estructura y organización interna del contenido a aprender. Por otro, la significatividad psicológica, relacionada con el hecho de que el aprendiz disponga de elementos en su estructura cognitiva que pueda poner en relación de manera sustantiva y no arbitraria, con ese contenido*". La otra implicación que menciona está relacionada con el propio usuario, el cual debe tener en cuenta tanto qué es lo que va a aprender y su significado, como el motivo por el cual va a hacerlo.

Un punto en el que Onrubia pone especial hincapié es el papel del profesor, siendo un pilar fundamental para la ayuda del alumno. Esta ayuda consiste en apoyar al alumno y mantener un flujo constante de comunicación con él, aunque esta comunicación sea de forma asíncrona y no directa, como podría ser utilizando foros, corrigiendo ejercicios del alumno o enviándole información extra sobre las lecciones que debe aprender. El alumno debe ser retado y, tanto él mismo como el profesor, deben comprobar cuál es su avance y su grado de comprensión de la materia.

Si se habla de involucrarse realmente dentro de un mundo virtual, se está hablando directamente de realidad virtual. Hilera, Otón y Martínez proponen esta tecnología como el medio ideal para enseñar y aprender [26]. Según ellos, "*el objetivo de la realidad virtual es crear una experiencia que haga sentir al usuario que se encuentra inmerso en un mundo virtual, aparentemente real. […] La realidad virtual utiliza la visión de un observador, el usuario, quien se mueve dentro del mundo virtual utilizando dispositivos adecuados, como gafas o guantes electrónicos*". En un mundo virtual de estas características, el usuario se siente sumergido completamente, puesto que es capaz de registrar a través de todos sus sentidos lo que ese mundo le puede ofrecer.

El problema de utilizar la realidad virtual de esta manera, es que es demasiado caro como para que sea accesible a centros de enseñanza, y más aún, a la mayoría de aquellos que quisieran ser estudiantes en un entorno de estas características. Es por esto que estos tres autores proponen el uso de VRLM (Virtual Reality Modeling Language), un lenguaje para la creación de mundos virtuales en Internet, algo mucho más accesible y igualmente válido a pesar de carecer de algunas de las ventajas de la realidad virtual original, como es la recepción de señales táctiles y olfativas.

Aún así, el estudiante se ve introducido por completo en el mundo, siendo más fácil captar su atención, así como que interiorice mejor los conocimientos que pueda ir adquiriendo al tener disponibles medios visuales y sonoros, o incluso táctiles en el caso de poder acceder a ellos, desde el propio ordenador, sin tener que viajar a ningún sitio real. El alumno se ve envuelto por un conjunto de sensaciones captadas a través del ordenador o mediante los dispositivos de realidad virtual, poniendo toda su atención en la materia a estudiar.

Sherman y Judkins explican la importancia de esta manera de aprender en determinadas profesiones, como puede ser la medicina, en particular la cirugía, ya que los futuros profesionales pueden ensayar en el mundo virtual sin posibilidad de crear un peligro real a ningún paciente, permitiendo además crear las características deseadas de la situación expuesta ante el estudiante. García Ruíz opina que otra de las profesiones más beneficiadas por este tipo de enseñanza es la del entrenamiento técnico, en especial los pilotos. De nuevo, entra en juego la posibilidad de que los estudiantes entrenen sin necesidad de poner en peligro ni a sí mismos ni a otros.

2.2.5. El futuro de los mundos virtuales

Se prevé que, dentro de unos años, la gran mayoría de la población que utiliza Internet sea también usuario de los mundos virtuales. Gente de todo el mundo se reunirá en estos emplazamientos y conversará entre sí sin tener en cuenta la localización geográfica real en la que se encuentren. Es cierto que esto ya se hace mediante los múltiples salones de chats existentes en la red, pero el hecho de encontrarse dentro de un mundo similar al real, con un avatar con el que el usuario se identifica, hace que, en la mayoría de los casos, las personas se sientan realmente en dicho mundo, mostrándose más abiertas.

De hecho, hay gente que ha hecho su vida en estos mundos virtuales. En Second Life hay multitud de empresas y empresarios que hacen negocios dentro de la realidad creada en ese mundo, ganando dinero real. Otras personas que tenían negocios fuera, los han dejado por negocios dentro del mundo, ya que en este segundo trabajo ganaban incluso más dinero que con el trabajo de la vida real. ¿Es posible que se creen puestos de trabajo dentro de los mundos virtuales? Siguiendo con el ejemplo de Second Life, existen personas que se dedican a decorar casas construidas por los residentes, construir ropa y complementos para éstos, etc., ganándose la vida con ello.

Hasta tal punto se consideran importantes los mundos virtuales que ya se están creando estudios para que llegue a toda la población, en especial a personas invidentes. IBM anunció que está interesada en crear un mundo que sea apto para personas con esta discapacidad, por lo que ha reunido a un grupo de estudiantes para que trabajen en el problema planteado. El objetivo es construir un mundo en el que el usuario se sienta inmerso gracias a los sonidos ambientales del entorno en el que se encuentran y a las descripciones orales de las personas y objetos de alrededor, así como de su ubicación.

Sobre los mundos virtuales, Rita J. King dijo que la gente se moverá a una posición donde la gente pueda complementar sus vidas físicas con las realidades virtuales. Nergiz Kern comenta: "*Las cosas cambian y se desarrollan demasiado rápido. Pero creo que los mundos virtuales se convertirán en algo tan normal como lo es Internet ahora. Mucha gente que ahora está online tendrá un avatar y usará mundos virtuales para toda clase de actividades, desde encontrarse y chatear con amigos hasta aprender y hacer negocios*".

2.2.6. Ejemplos de mundos virtuales

- **Second Life**

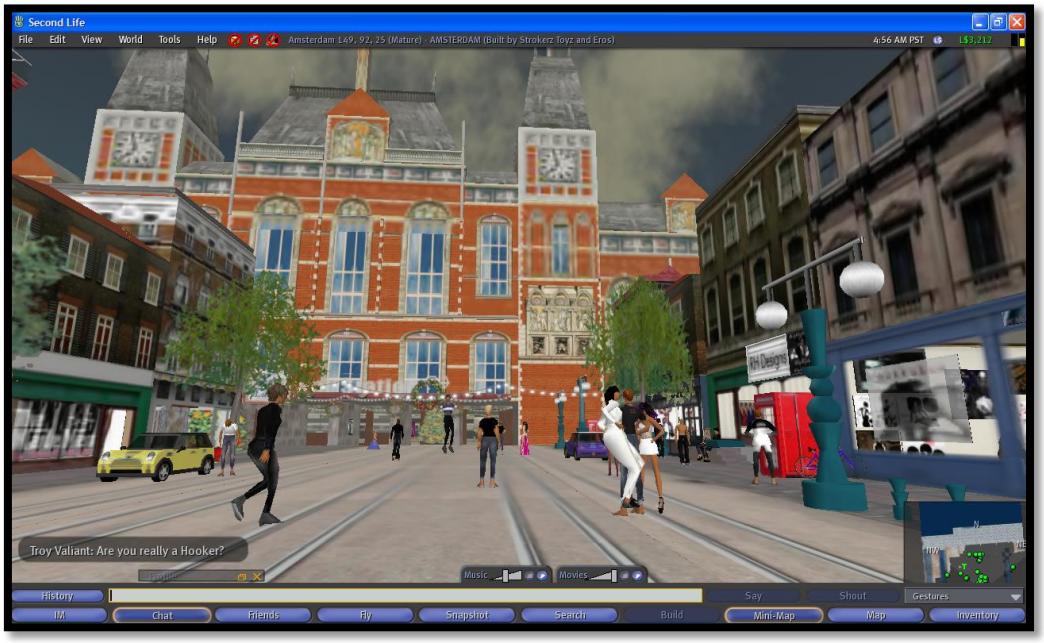

Imagen 15: Imagen de Second Life

Sin duda, uno de los mundos más famosos en Internet es Second Life, abreviado como SL. Fue concebido por Linden Lab, fundada en 1999 por Philip Rosedale y hoy en día millones de personas de todas partes han creado a sus propios avatares y se han unido a este nuevo mundo en el que pueden mantener, casi literalmente, una segunda vida. Se construyó basándose en la novela de ciencia ficción “Snow Crash”, escrita en 1992 por Neal Stephenson. En ella existe el Metaverso, una especie de mundo paralelo en el que las personas tienen sus propios avatares y llevan una realidad paralela a la suya.

La finalidad de este programa es discutida, pues no sigue los patrones normales de los juegos al no tener un objetivo concreto. SL se ha convertido más bien en un punto de reunión donde conocer gente, hablar con ellos, aprender, interactuar con todos los objetos que en él se encuentran, y un largo etcétera. ¿Por qué se ha hecho tan famoso? Precisamente por esa capacidad de interconectar personas en un mundo enorme, aunque sea de manera virtual.

Los usuarios, llamados residentes, tienen a sus propios avatares para representarles, pudiendo tomar la forma que ellos deseen, desde una imagen aproximada a ellos mismos hasta objetos inanimados o abstractos.

Una vez se introducen en el mundo, se pueden comunicar entre ellos de dos maneras: haciendo uso de un chat general en el que participarán aquellos usuarios que se encuentren cerca, o por medio de mensajes instantáneos y privados desde cualquier lugar del amplio mapa.

La vida en Second Life sigue los principios de la vida en el mundo real. La economía, de igual forma, está presente en este entorno en forma del Linden Dólar, la cual incluso puede canjearse por moneda real dentro del juego. Un dólar estadounidense se corresponde con aproximadamente 250 dólares linden. Con este dinero se pueden comprar terrenos, edificios, ropa, joyas, obras de arte, etc. Como se puede ver, hay una amplia gama de objetos y bienes materiales que se pueden adquirir.

En cuanto a qué se puede hacer en este mundo, sería quizá más adecuando preguntar qué no se puede hacer. Second Life ha sido usado como plataforma para la educación por universidades e institutos, que hacen que existan zonas destinadas al aprendizaje de determinadas materias, incluyendo el estudio de idiomas. Es posible acceder a exhibiciones de arte, música en vivo, o ver una obra de teatro. Se puede incluso practicar o ir a ver deportes como boxeo, futbol americano, lucha, etc., eso sí, sin apostar, ya que está prohibido. El juego es algo que también está presente. Se pueden jugar desde juegos de role multijugador, con varias temáticas, a juegos de mesa como el ajedrez.

Second Life ha llegado a alcanzar tal fama que algunos organismos oficiales han decidido ocupar un pequeño espacio de este mundo para llegar a más gente. Apple ha puesto a disposición de los clientes una tienda en la que podrán ver sus productos, algunas iglesias tienen sus propias islas en las que proclaman su fe y atraen a la gente con el fin de que allí se sientan más libres de hablar cuando lo necesiten, los partidos políticos de Izquierda Unida, Partido Popular y PSOE tienen también su pequeño espacio en el mundo virtual.

- **World of Warcraft**

Imagen 16: Imagen de World of Warcraft

Otro mundo virtual algo diferente es el que creó Blizzard Entertaiment con World of Warcraft, llamado más comúnmente WoW. En este caso se trata de un juego de role masivo por Internet considerado como el MMORPG más grande y más popular del mundo.

Los jugadores crean a su avatar dentro del mundo, eligiendo la raza y la clase, a qué se dedican, siempre dentro del mundo de la fantasía. Así se puede tener un humano mago o un enano guerrero. Antes de entrar, deberán elegir un reino en el que jugar, en cada uno los cuales interactuarán de manera distinta con el medio y con otros jugadores. Una vez dentro se dedicarán a hacer misiones en las que matarán a criaturas y se relacionarán con otros personajes, tanto llevados por otros jugadores como creados por los desarrolladores.

Una vez más, se hace tan importante la parte lúdica como la de relacionarse con los demás jugadores. Se pueden hacer grupos con los que ir de aventuras, o hacer tratos comerciales con otros avatares.

- **Moose Crossing**

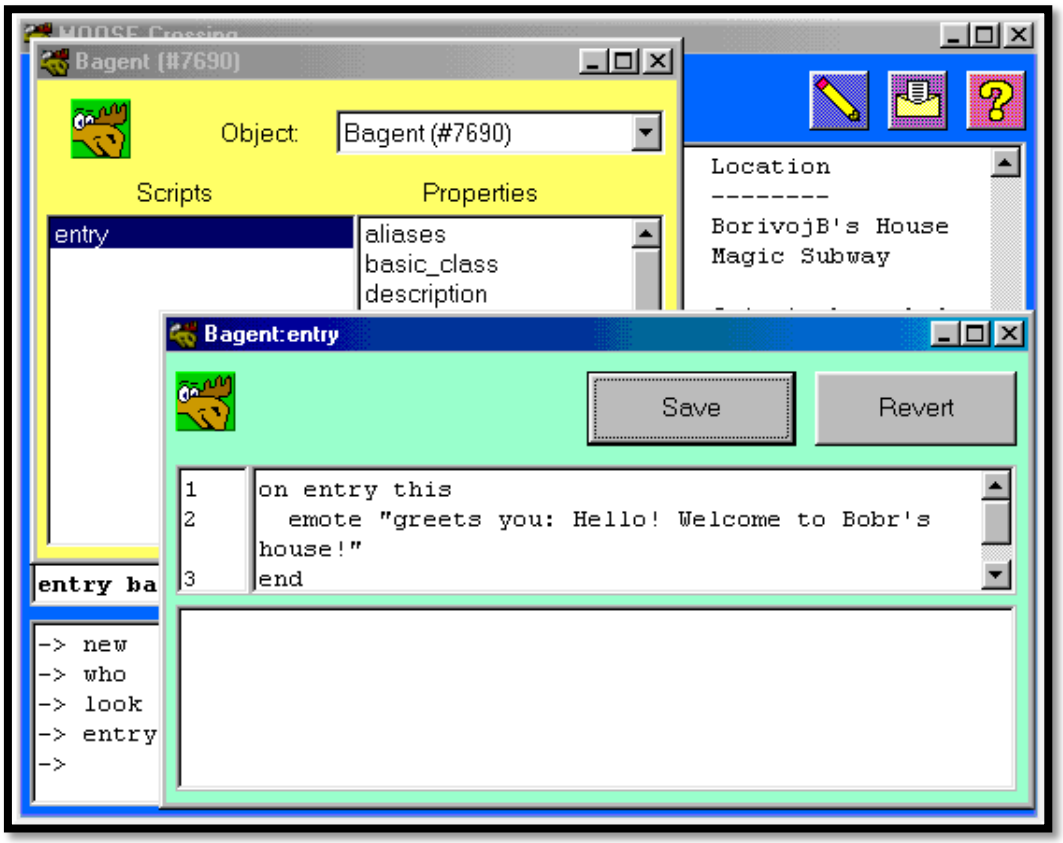

Imagen 17: Imagen de Moose Crossing

Moose Crossing es un tipo de MUD. Los MUD (siglas de Multi User Dungeon en un principio, y Multi User Domain más tarde) son unos mundos virtuales multijugador basados en texto [27]. Los MUDs tienen su origen en un juego que crearon en 1967 Will Crowther y Don Woods, al que llamaron *ADVENT*. En 1975, Crowther creó *Adventure*, el primer MUD.

Durante sus primeros años, los MUDs estaban basados en juegos de role dentro del mundo de la fantasía, donde los jugadores tenían que ir ganando experiencia y objetos según iban completando misiones y matando monstruos. En 1989 James Aspnes decidió darles un pequeño giro y dejó de un lado esta fantasía para convertirlo en algo que se parecía más a un conjunto de salas de chats creadas por los usuarios.

Moose Crossing, por tanto, es un mundo virtual creado por Amy Bruckman, destinado al aprendizaje por parte de niños de entre siete y trece años. Estos estudiantes pueden construir lugares y objetos, así como manipular libremente los que ya están construidos.

Dentro de esta comunidad están por una parte los niños, que son los que interactúan con el medio, y por otro, los adultos, que tiene el role de cuidadores. Estos

últimos están en el mundo virtual para ayudar a los jóvenes en todo aquello que necesiten, como problemas técnicos o sociales.

Lo que aquí se aprende es a programar a través de ir creando pequeños objetos en el mundo de Moose Crossing; los niños más pequeños aprenden cómo realizar sencillos programas mientras que los mayores los realizan más complejos. Los objetivos de este mundo virtual son el desarrollo de nuevas tecnologías, el uso de éstas con niños y el análisis de estas actividades y su aprendizaje [27].

Una vez se entra dentro del mundo, los usuarios deben construir su propia casa dentro del barrio Digital Acres, donde se encuentran las viviendas de todos los personajes. A continuación, es momento de crear una mascota, pues todo el mundo tiene, al menos, una. De esta forma, y construyendo nuevas cosas, se va aprendiendo a entenderse con la máquina en su propio lenguaje, es decir, a través de instrucciones. Como complemento adicional, los niños pueden relacionarse entre sí hablando y visitándose mutuamente allá donde se encuentren.

Moose Crossing es, por tanto, un sitio de aprendizaje para los niños, tanto de aspectos técnicos y cercanos a la programación como sociales.

Otros ejemplos de sistemas de aprendizaje virtual son:

- **Apex Learning**.

Es un sistema de e-Learning privado especialmente diseñado para la educación primaria y secundaria, que ofrece cursos de matemáticas, ciencias, inglés, sociales y lenguas romances [101]. Fue fundada en 1997 por Microsoft y Paul Allen.

Ofrecen cuatro tipos de productos:

- Clases y herramientas virtuales para el aprendizaje a distancia: más de sesenta cursos online en áreas como matemáticas, ciencias, inglés, sociales y lenguas del mundo.
- Herramientas de clases: provee contenido digital y evaluaciones que ayudan a los profesores a dirigir el aprendizaje de los alumnos.
- Revisión de exámenes: ofrece tests de preparación online para que los alumnos puedan practicar para los exámenes.
- Preparación de exámenes: son cursos de preparación de determinados exámenes.

- **ATutor**.

Es un sistema de Gestión de Contenidos de Aprendizaje diseñado con el objetivo de lograr accesibilidad y adaptabilidad [102]. El proyecto comenzó en el año 2002

como un producto de Adaptive Technology Resource Centre (ATRC) de la Universidad de Toronto.

Los profesores pueden ensamblar, empaquetar y redistribuir contenido educativo, además de llevar a cabo sus clases online. Los estudiantes pueden aprender en un entorno de aprendizaje adaptativo. Contiene herramientas de gerencia y administración de alumnos, tutores y cursos. Provee evaluaciones en línea, y herramientas de autoría y colaboración.

ATutor es un programa diseñado en PHP, Apache, MySQL, trabaja sobre plataformas Windows, GNU/Linux, Unix, Solaris, y da soporte a 32 idiomas.

- **OLAT**.

OLAT es el acrónimo de **O**nline **L**earning **A**nd **T**raining (Aprendizaje y Prácticas Online). Es una aplicación web gratuita que permite cualquier tipo de enseñanza, aprendizaje y tutorías online [103]. Se desarrolló a partir de 1999 en la Universidad de Zúrich.

Proporciona las siguientes características:

- Gestión de contenido.
- Foros.
- Pruebas con diferentes tipos de preguntas.
- Wikis.
- Encuestas.
- Chat.
- Módulo de ejercicios.
- Módulo de calificación.
- Soporte multilenguaje.

- **WebCT**.

WebCT es un sistema comercial de aprendizaje virtual online usado principalmente por instituciones educativas [104]. Fue desarrollado en 1995 por la Universidad de Columbia Británica, en Canadá.

2.3. Cloud computing

2.3.1. Introducción

A continuación se procederá al desarrollo del apartado de Cloud Computing. Esta nueva tecnología empezó a despuntar en el 2008, cuando varios expertos en el tema lo tachaban como la nueva área en expansión, afianzándose en el 2009 al ser incluido dentro de las tendencias más importantes del año en el mundo tecnológico.

Pese a no ser una tecnología de nuevo desarrollo, ya que tiene más de dos años, sí es cierto que es ahora cuando más está despuntando, debido a que los avances computacionales permiten sacarle el máximo.

Consiste en permitir el uso de aplicaciones o servicios a través de Internet, ya sea para uso cotidiano por cualquier usuario desde su hogar o para las empresas. De hecho, su característica más importante es precisamente que estas aplicaciones son accedidas a través de la cloud, nube en inglés, metáfora de Internet. Se podría decir que es una evolución de la Web 2.0, en la que los datos de todos los usuarios y empresas podrían ser accedidos desde cualquier sitio y operar sin tener que instalar los dispositivos necesarios, con el coste que esto conlleva.

En la vida cotidiana, hay varios ejemplos de Cloud Computing a nivel de usuario normal, no de empresas. Uno de estos ejemplos es Google Docs, un programa en línea para la creación de documentos de manera colaborativa con los formatos de documentos más extendidos: procesador de textos, hojas de cálculo y un programa para la realización de presentaciones.

Aunque el uso de aplicaciones por Internet puede no parecer una novedad digna de tanta mención, el caso es que cloud computing es más que esto, proporciona no sólo las aplicaciones, si no los servidores donde ejecutar y almacenar la información para que éstos funcionen, así como el ancho de banda necesario para que se suplan todas las necesidades que puedan surgir por su uso. Pero no se trata simplemente de un conjunto de servidores y servicios para el uso del cliente, además, gestiona su propio empleo de los recursos disponibles para que se haga de la manera más eficiente, sin necesidad de gastar más de lo que se necesita y evitando posibles problemas causados por no disponer de los suficientes recursos.

IBM, en su documento "*Cloud Computing*" [28], define esta tecnología tanto como una plataforma como un tipo de aplicación: "*Una plataforma de Cloud computing dinámicamente suministra, configura, reconfigura y retira servidores según las necesidades. […] Cloud computing también describe aplicaciones que son extendidas para ser accesibles a través de Internet. Estas aplicaciones de la nube usan grandes almacenes de datos y poderosos servidores que alojan aplicaciones Web y servicios Web. Cualquiera con una conexión adecuada a Internet y un navegador estándar puede acceder a una aplicación de la nube*".

Por otra parte, el NIST (National Institute of Standards and Technology, Instituto Nacional de Estándares y Tecnología) define Cloud Computing como [29]: "*un modelo para permitir convenientemente y bajo demanda el acceso a través de la red para compartir un conjunto de recursos de computación configurables (por ejemplo redes, servidores, almacenamiento, aplicaciones y servicios) que pueden ser suministrados y liberados rápidamente con un mínimo esfuerzo de gestión o interacción del proveedor del servicio*".

La tecnología de Cloud Computing bebe de la influencia de otros modelos que han estado presentes en los últimos años como las novedades más influyentes entre las empresas:

- SaaS o Software como Servicio: Un modelo de distribución software a través de Internet.
- Utility Computing: Se define como el suministro de recursos computacionales, por ejemplo procesamiento y almacenamiento, como un servicio medible.
- Grid Computing: Es una tecnología que permite utilizar de forma coordinada todo tipo de recursos (almacenamiento, cómputo, aplicaciones) que no están sujetas a un control centralizado, si no que se hace de manera distribuida. Los recursos utilizados no tienen por qué ser homogéneos, pudiendo valerse de distintos sistemas operativos, arquitecturas, supercomputadores, etc. La conexión entre estos sistemas es mediante redes, normalmente a través de Internet. En grid computing la clave está en cómo conectar los distintos sistemas para llevar a cabo los objetivos propuestos, a diferencia de Cloud Computing, en el que el usuario no tiene conocimiento sobre cómo está dispuesta la infraestructura utilizada.

2.3.2. Funcionamiento y Uso de Cloud Computing

Lo que hace Cloud Computing es virtualizar los recursos que ofrece de manera que parezca que el cliente está utilizando los suyos propios cuando en realidad está accediendo a los mismos a través de Internet. Gracias a ello, puede [28]:

- Alojar diferentes tipos de trabajos, incluidos procesos batch o trabajos back-end y aplicaciones interactivas de cara al usuario.
- Permitir trabajos para ser implementados y escalados rápidamente a través de la adjudicación de máquinas, virtuales o físicas.
- Ofrecer un sistema de apoyo redundante, autorecuperación y un modelo de programación altamente escalable que permite la recuperación a partir de inevitables fallos de hardware y software.
- Controlar los recursos en tiempo real para permitir balancear la asignación de tareas cuando sea necesario.

Este último punto es el que puede ser el más interesante para las empresas, sobre todo aquellas que contraten un proveedor en el que sólo pagan por aquello que consumen. Esto es, si necesitan dos servidores para llevar a cabo su tarea, únicamente pagan por estos dos servidores. En el momento en el que necesiten más, dinámicamente se les asignarán los necesarios, ajustándose de nuevo el precio. De esta manera, en cualquier momento pueden contar con una gran cantidad de recursos de todo tipo, hardware, software, almacenamiento, procesamiento, etc., sin necesidad de tenerlos físicamente en propiedad, utilizando sólo aquello que necesitan, y gastando mucho menos de lo que les costaría si no utilizaran al proveedor.

En el artículo "*Cloud Computing, las TI como servicio*" [30], se propone el uso de Cloud Computing en la empresa, sobre todo en aquellas empresas pequeñas o las llamadas start-ups, que no tienen posibilidades, o no quieren invertir tanto dinero en grandes cantidades de recursos informáticos que sufraguen sus necesidades.

Juan Manuel Rebés se declara a este respecto así: "*este modelo está inspirado en la idea de disponer de infraestructuras tecnológicas de modo que los recursos informáticos sean compartidos dinámicamente, se encuentren virtualizados y resulten accesibles como un servicio. Aúna de esta manera gran parte de las nuevas tendencias de software como servicio, virtualización de recursos, redes grids e informática bajo demanda. En el modelo de Cloud Computing los grandes clusters de sistemas se enlazan entre sí para proporcionar servicios tecnológicos como si se tratasen de un único superordenador global*".

Daryl C. Plummer, analista de la empresa Gartner, dice así: "*La virtualización, la orientación al servicio e Internet han convergido para fomentar un fenómeno que permite a consumidores y negocios elegir cómo adquirir o suministrar servicios TI. […] Los servicios suministrados a través de la nube robustecerán una economía basada en la entrega y consumo de cualquier cosa, desde el almacenamiento a la computación, el vídeo o la gestión de las finanzas*".

Como se puede ver, la opinión general de los expertos es que el Cloud Computing en el ámbito empresarial abre una nueva época en la que el aspecto productor-consumidor toma una nueva perspectiva: ya no tiene por qué ser la propia empresa la que lleve a cabo todas las tareas informáticas, el departamento de TI puede ahorrarse mucho dinero y quedar reducido a la mínima expresión gracias al uso de una tercera empresa que aporte soluciones a las necesidades tecnológicas de la primera.

Aunque el movimiento no sea nuevo, es ahora cuando la mentalidad está cambiando, quizá por la necesidad de reducir gastos que se está extendiendo debido al momento económico por el que se está pasando. Cada vez hay más empresas que contratan los servicios tecnológicos tal y como se contrata la línea telefónica o el agua: servicios que llegan hasta el cliente sin tener que preocuparse por cómo sucede.

Este uso se está viendo aumentado debido a las grandes posibilidades que permite en los más variados escenarios [28]. El más extendido es el comentado a lo largo de este documento:

empresas que contratan la infraestructura tecnológica de su departamento de TI para no tener que hacer esa gran inversión ellos mismos.

Hay otros usos, sin embargo, menos extendidos y no por ello menos útiles. Aquellas empresas que decidan crear un mundo virtual pueden utilizar esta solución para conseguir las grandes cantidades de capacidad computacional que se necesitan, sobre todo si el mundo virtual creado es multi-usuario. Gracias a esta técnica se podrían dedicar los servidores necesarios dependiendo del número de usuarios conectados, además de proporcionar la capacidad de almacenamiento necesaria para guardar las instancias de los participantes y su situación dentro del entorno ficticio creado.

El mundo del E-business también puede verse beneficiado en la misma medida que los mundos virtuales. La escalabilidad que proporciona Cloud Computing se puede aprovechar para aumentar el número de servidores a disposición del cliente si el uso que se está haciendo del sistema es elevado.

2.3.3. Características de Cloud Computing

El rasgo principal del Cloud Computing es precisamente que toda su infraestructura está basada en el uso de Internet como medio de acceso y comunicación con los recursos ofrecidos, ya sean hardware o software. El hecho de que funcione así hace que el cliente no necesite saber cómo desempeña o reparte realmente el trabajo propuesto, sino que simplemente lo utilice y espere los resultados. Igualmente, el acceso a los servicios ofrecidos se puede hacer desde cualquier sitio, sin tener que estar sujeto a un emplazamiento concreto desde el que poder trabajar: basta con estar conectado a Internet.

Otro de los grandes rasgos importantes de este modelo es que el cliente sólo paga por lo que usa. Esto es así porque la utilización de los recursos proporcionados es medible, pudiendo así poner un precio, por ejemplo, a la capacidad de almacenamiento usada o al ancho de banda necesitado.

Los recursos se van reajustando según el cliente necesite en cada momento. Al ir asignando recursos según sea necesario, es posible que una misma aplicación acabe ejecutándose en varias máquinas o que una máquina lleve a cabo varias aplicaciones distintas. Esto es así porque se independiza las aplicaciones del hardware, se crean máquinas virtuales en las que las aplicaciones realizan sus tareas independientemente de la disposición física de éstas.

Es más, los recursos disponibles en la empresa proveedora son compartidos por todos los clientes que tienen, de esta forma se puede hacer el ajuste de la carga asignando más o menos recursos a cada uno dependiendo de cuánto necesiten en cada momento. Esto, sin embargo, no afecta a la seguridad, ningún cliente ve peligrada la confidencialidad de los datos ni su privacidad.

Para las empresas que quieren utilizar este tipo de servicios, es importante que confíen en el proveedor de los mismos. No se pueden permitir que, debido a problemas en la red, se queden sin poder satisfacer sus propias necesidades o que, simplemente, se les borren los datos por un error en el servidor. Para asegurarse de que se cumplen los requisitos necesarios para el buen funcionamiento y entendimiento entre la empresa solicitante de los servicios y el proveedor, se debe firmar un acuerdo entre ambas partes, denominado SLA (Service Level Agreement, o Acuerdo de Nivel de Servicio). En este documento se definirán las políticas que se van a seguir, como por ejemplo tiempos de respuesta, disponibilidad del servicio, etc.

Debido a estos requisitos que el proveedor debe cumplir frente a los clientes, sobre todo en lo que respecta a mantener la congruencia de los datos y a la fiabilidad de las aplicaciones, la información necesaria es almacenada redundantemente en backups que se utilizarían en caso de fallo.

Algo que se debe tener en cuenta, puesto que es una característica importante, es que estos sistemas son escalables. Esto quiere decir que el funcionamiento de los procesadores es predecible de acuerdo a una simple regla: un servidor maneja un determinado número de transacciones, luego dos servidores juntos ejecutarán justamente el doble de transacciones. Esto es importante para distribuir los recursos entre los clientes, sabiendo cuántas transacciones se tienen que llevar a cabo para un cliente, se sabe qué es lo que se le tiene que proporcionar.

2.3.4. Arquitectura de Cloud Computing

Esta tecnología tiene una arquitectura en capas, cada una de las cuales se trata a su vez de otra tecnología proveedora de servicios ya existente con anterioridad. Así pues, Cloud Computing es un cúmulo de tecnologías ya implantadas con anterioridad y que, todas ellas aunadas, proporcionan un servicio mayor a empresas y particulares.

El gráfico que muestra cómo está organizada esta arquitectura es el siguiente [32]:

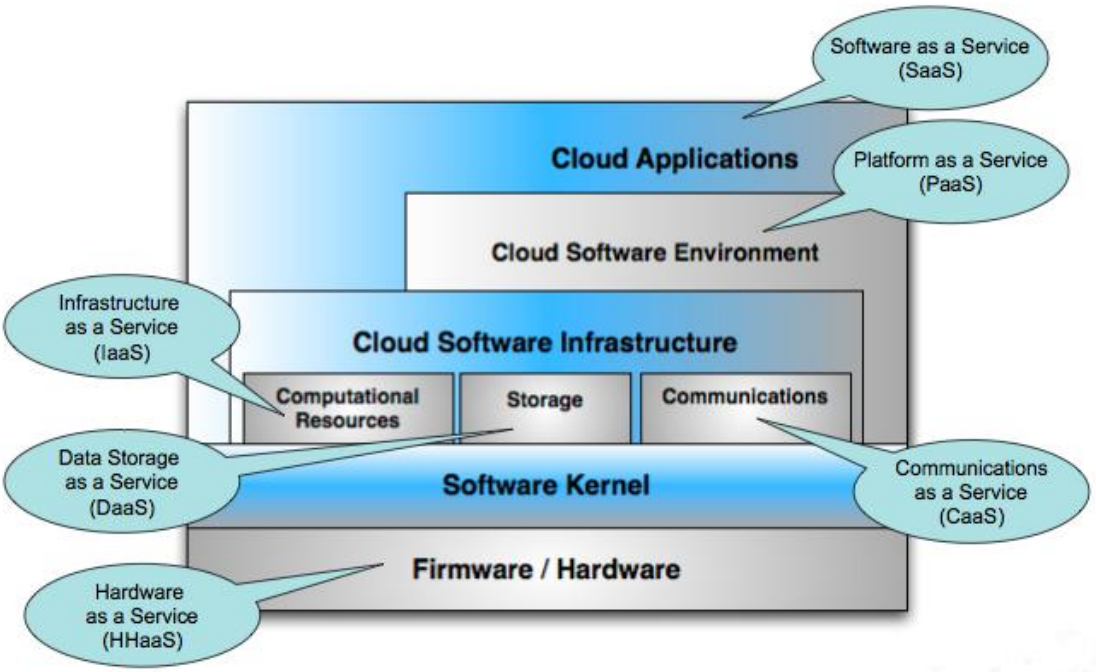

Imagen 18: Arquitectura del Cloud Computing

A continuación se detallará cada una de las capas:

- **Software as a Service (SaaS)**.

Este término empezó a circular a finales de los años noventa, aunque fue a partir del año 2000 cuando empezó a tener más auge entre el mundo empresarial y profesional. Es traducido como "Software como Servicio", y se trata de un modelo de distribución software: la empresa distribuidora aporta el servicio de mantenimiento, operación diaria y soporte del software solicitado por el cliente. Lo que esto quiere decir es que el cliente tiene las aplicaciones que necesita, la lógica de su negocio, en una tercera empresa a la que contrata. El servidor central, por tanto, se encontraría en la empresa proveedora y no en la del cliente.

El software es accedido a través de Internet, aunque no necesariamente tiene que ser a través de navegadores Web, puede ser una aplicación alojada en un servidor remoto. Los usuarios usan una misma aplicación y comparten los recursos, por lo que es evidente que dichos programas deben ser aptos para trabajar de forma concurrente con un número alto de usuarios.

En la imagen siguiente se puede ver un ejemplo de cómo funciona este modelo 0:

Imagen 19: Funcionamiento de SaaS

Los clientes, normalmente empresas, pagan por acceder a unos servicios o aplicaciones suministradas por una empresa externa, que a su vez puede comprar los programas necesarios para su funcionamiento a una tercera empresa.

Es fácil confundir este término con Cloud Computing, pues ambos parecen tener los mismos objetivos. Sin embargo, Cloud Computing se refiere al uso de los servicios tecnológicos a través de Internet, pudiendo ser éstos aplicaciones, almacenamiento, procesamiento, etc. SaaS simplemente ofrece el uso de software a través de la red.

Una aplicación SaaS puede encontrarse en uno de los cuatro niveles de madurez definidos [33]:

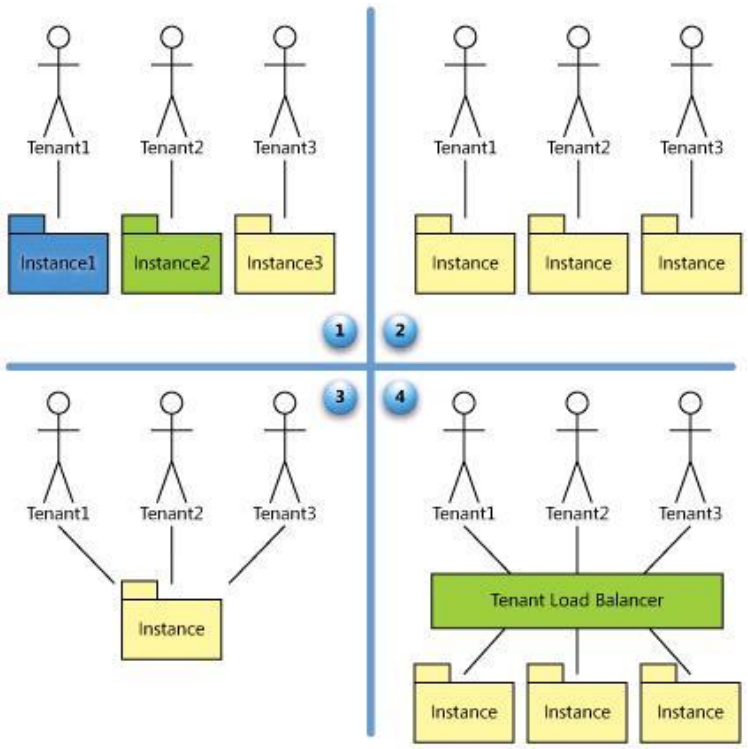

Imagen 20: Niveles de Madurez del SaaS

- Nivel 1: Modelo ASP (Application Service Provider). El cliente aloja el software o aplicación en un servidor externo. Cada uno de los usuarios

tiene su propia versión de la aplicación y ejecuta su instancia de la misma en el mismo servidor que la aloja.

- Nivel 2: Configurable. Cada usuario tiene su propia instancia de la aplicación, aisladas las unas de las otras, aunque todas ellas tienen el mismo código. Se facilita de este modo el mantenimiento del software.

- Nivel 3: Configurable, Multi-usuario. Se añade la capacidad de mantener una única instancia en el servidor para todos los clientes, aunque sigue personalizándose para cada uno de ellos. El control sobre el uso de la aplicación se hace mediante permisos que restringen o permiten el acceso a ciertas partes del software e información del mismo. Se consigue en este nivel facilitar aún más el mantenimiento al tenerse tan sólo una instancia, así como ahorrar en coste y espacio para disponer de almacenamiento suficiente para todas las instancias, como ocurría en los casos anteriores.

- Nivel 4: Configurable, Multi-usuario, Escalable. Se tiene un conjunto de instancias de la aplicación que se usan según las necesidades, atendiendo al número de clientes que estén usando el software. Así se hace que el sistema sea escalable a un número indeterminado de clientes evitando tener que rediseñarlo.

Las ventajas vienen dadas por la comodidad de no tener los sistemas alojados en el propio cliente, sino que es una tercera empresa la que carga con las dificultades de mantenimiento, actualizaciones, seguridad y control del funcionamiento del mismo. Además, puesto que tiene que satisfacer las necesidades del cliente para que éste siga contando con sus servicios, pone especial interés es atenderle y cumplir los requisitos de su contrato, así como en mantener la seguridad de los servicios disponibles. Las empresas proveedoras ofrecen a sus clientes, asimismo, la libertad de utilizar el sistema operativo que prefieran, sin poner limitaciones y ofreciendo la máxima flexibilidad posible.

Por supuesto, los inconvenientes también son fruto de no contar el cliente con sus propios sistemas alojados localmente. El tener los datos en una fuente remota hace que se dependa en todo momento de la conexión a Internet. Un fallo en la red, tanto del cliente como del servidor, puede ser fatal para la empresa, pues no podría disponer de su motor de trabajo.

La privacidad y seguridad de los datos también se ve amenazada, pues no deja de ser un posible blanco para ataques externos malintencionados.

- **Platform as a Service (PaaS)**.

El modelo PaaS, o “Plataforma como Servicio”, consiste en ofrecer la infraestructura necesaria al cliente para que éste pueda desarrollar e implantar sus

aplicaciones Web, normalmente aplicaciones SaaS, sin tener que contar con el software y los equipos necesarios para realizar dicho desarrollo.

PaaS incluye todas las facilidades al programador para prototipar, analizar, desarrollar, documentar y poner en marcha aplicaciones, todo en un solo proceso. PaaS da servicio de integración de la base de datos, seguridad, escalabilidad, almacenaje, copias de seguridad y versiones, habilitando la posibilidad realizar trabajos colaborativos [36].

Una buena forma de visualizar qué es PaaS es la siguiente [37]:

Imagen 21: Qué es PaaS

Como características, se pueden mencionar [37]:

- Herramientas de desarrollo multi-usuario: las herramientas proporcionadas por la empresa proveedora permiten múltiples usuarios para el desarrollo de las aplicaciones. Cada uno de los usuarios podrá tener, a su vez, varios proyectos abiertos en los que trabajar.
- Arquitectura multi-usuario: los servicios ofrecidos mediante PaaS aseguran que el sistema desarrollado podrá ser accedido por un número ilimitado de usuarios, garantizando que el sistema será escalable.
- Gestión integrada: se proporcionan herramientas de gestión integradas en la etapa del desarrollo.
- El pago del sistema se hace mediante facturas en las que se paga por lo que se usa.
- El desarrollo se hace a través de los servicios ofrecidos por la empresa proveedora, la cual proporciona herramientas de desarrollo que no es necesario que el cliente se instale en su propio equipo.
- El despliegue de la aplicación lo lleva a cabo el cliente a través de herramientas proporcionadas por la empresa proveedora, no es necesario contactar con ninguna persona intermediaria que despliegue el sistema.

Como en todos los modelos "as a Software", las ventajas son consecuencia de no tener que hacer la inversión de comprar equipos y software necesarios para realizar el desarrollo de las aplicaciones de este estilo. Es más, el proveedor de PaaS se encarga de costear las actualizaciones, parches, sistemas operativos, etc. Y, una vez más, el cliente sólo paga por el uso que hace, no por la infraestructura que es posible no utilice en su totalidad.

Como puntos en contra destaca la dependencia del cliente con el buen funcionamiento de Internet: si éste falla entonces él no podrá realizar sus aplicaciones. Esto genera desconfianza por parte de los clientes al utilizar sistemas remotos de este estilo, pues genera miedo a no poder acceder a sus datos o a que otros se introduzcan en ellos sin los permisos adecuados.

Además, los proveedores de PaaS ponen a disposición del cliente un software y hardware específico: arquitecturas, bases de datos, sistemas operativos, herramientas de desarrollo. Por muchas opciones que éste ofrezca, el cliente siempre dependerá de las posibilidades que se le planteen.

- **Infrastructure as a Service (IaaS)**.

La "Infraestructura como Servicio" es un modelo que proporciona la capacidad de cómputo a un cliente mediante sistemas virtualizados a través de Internet. Esta capacidad de cómputo incluye almacenamiento, hardware, servidores y equipamiento de redes. La empresa proveedora es la responsable de toda esta infraestructura y de que el funcionamiento sea el que el cliente necesite y que no se produzcan fallos de ningún tipo, incluyendo de seguridad [38].

El cliente paga por el uso que hace: por la cantidad de espacio que esté usando, el número de servidores que estén a su disposición, etc.

Las características principales de IaaS son:

- Un acuerdo de nivel de servicio (SLA) entre proveedor y cliente para acordar cuáles serán las condiciones del contrato entre ambos.
- Pago según el uso de las capacidades computacionales ofrecidas por la empresa proveedora.
- El entorno proporcionado por la empresa proveedora será en forma de máquinas virtuales.
- La empresa proveedora se encargará de proporcionar todo el hardware necesario para satisfacer las necesidades del cliente, incluyendo opciones para la escalabilidad de las aplicaciones.

- La empresa proveedora se encargará de proporcionar todo el software necesario para mantener las necesidades del cliente, como cortafuegos, balanceo de carga entre los servidores, sistemas operativos, etc.

- La empresa proveedora será capaz de asegurar al cliente una conectividad a Internet sin fallos, con copias de seguridad que garanticen la integridad de los datos.

➢ **Data Storage as a Service (DaaS)**.

"Almacenamiento de Datos como servicio" es una capa que se encarga de proporcionar la gestión y el mantenimiento completo de los datos manejados por los clientes. Trabaja conjuntamente con IaaS.

➢ **Communications as a Service (CaaS)**.

La capa de "Comunicaciones como Servicio" trabaja, igual que DaaS, en el mismo nivel que IaaS. Este modelo se encarga de proveer el equipamiento necesario de redes y la gestión de las comunicaciones, como el balanceo de carga.

➢ **Software Kernel**.

Esta capa gestiona la parte física del sistema. Controla los servidores a través de los Sistemas Operativos instalados, el software que permite la virtualización de las máquinas, la gestión de los clusters y del grid, etc.

➢ **Hardware as a Service (HaaS)**.

El "Hardware como Servicio" es la capa de más bajo nivel en el modelo de Cloud Computing. Trata la parte física de los elementos necesarios para trabajar a través de Internet, consistiendo éstos en centros con máquinas que proporcionan la computación, almacenamiento, servidores, etc.

2.3.5. Tipos de Cloud Computing

Según el NIST (National Institute of Standards and Technology) existen cuatro tipos de Cloud Computing [29]:

- Cloud pública: Estos servicios se caracterizan por estar disponibles para los clientes por medio de proveedores a través de Internet. Que sea público no siempre quiere decir que sea gratis, aunque es posible que así sea. De igual manera, el término público no implica que el acceso sea libre, ya que en la mayoría de los casos se requiere una autenticación para poder hacer uso de los servicios suministrados. La cloud pública provee un medio flexible y rentable para el desarrollo de soluciones.

- Cloud privada: Este servicio ofrece muchos de los beneficios de una cloud pública. La diferencia entre ambas reside es que en la privada los datos y procesamientos están gestionados dentro de la organización sin las restricciones del ancho de banda, seguridad y requisitos legales que puede comportar el uso de la pública. Este tipo de servicios, además, ofrece al cliente la posibilidad de controlar más la infraestructura proporcionada, mejorando la seguridad y la recuperación.

- Community Cloud: Este tipo de cloud está controlada y usada por un grupo de organizaciones que comparten los mismos intereses, como una misión común o requisitos de seguridad similares.

- Cloud híbrida: Esta cloud es una combinación entre la cloud pública y privada.

2.3.6. Ventajas y desventajas

Como todas las soluciones que se proponen, el Cloud Computing cuenta con puntos a favor y puntos en contra. Para saber si es conveniente instaurarlo hay que hacer un balance de estos pros y contras y estudiar cada caso individualmente. A continuación se pueden ver las principales ventajas y desventajas:

- Ventajas [31]:

  - Acceso a la información y los servicios desde cualquier lugar. No es necesario que el usuario se encuentre donde están alojados los servidores físicos, ni si quiera en su propia casa o empresa, podrá acceder a su información y a los servicios solicitados desde cualquier emplazamiento, pues éstos se encuentran en Internet.

  - Puesto que no todos los usuarios son iguales ni tienen las mismas necesidades, existen diferentes servicios que podrán contratar según sea la situación en la que se encuentren. Así se podrá contar con servicios gratuitos o de pago, adecuándose estos últimos, además, al uso que se haga de ellos, pagando por el número de servidores usados, aplicaciones ejecutadas, o tasa de subida o descarga.

  - Las empresas podrán contar con la facilidad de escalabilidad que aporta Cloud Computing. Pueden obtener un mayor número de recursos según lo necesiten sin tener que pagar por tenerlos físicamente en sus sucursales, con el lógico gasto de hardware, software y personal. Es más, si en determinada época del año o, incluso, momento del día, el tráfico es muy bajo, no es necesario que tengan en sus oficinas servidores sin ninguna carga de trabajo, sino que de

esta forma no tendrían ni que pagar por tener los servidores en la empresa proveedora de los mismos.

- Los usuarios no tienen una cantidad de recursos determinada, si no que comparten todas las capacidades de la empresa proveedora. Gracias a esto, se puede balancear la carga de cada uno según la actividad que tenga en cada momento, evitando la escasez de recursos incluso cuando se están haciendo tareas de actualización o mantenimiento.

- El uso de copias de seguridad aporta cierta confianza y seguridad frente a las posibles pérdidas de información por fallos en el sistema.

➢ Desventajas [30]:

- Hacer una SLA que realmente se cumpla y que satisfaga tanto a cliente y proveedor no resulta fácil, debido a que esta tecnología no ha terminado de consolidarse.

- Aún no se ha extendido lo suficiente para que pueda considerarse como una nueva forma de entender la empresa, o al menos el departamento de TI, por lo que aún se genera desconfianza respecto al éxito que se puede tener.

- La seguridad y privacidad de los datos es una de las mayores preocupaciones de las empresas, incluso cuando esta información la tienen almacenada dentro de su propia red. Con Cloud Computing, estos datos se guardan en servidores que pueden estar en cualquier parte del mundo y, además, que comparten varios clientes. El riesgo, por lo tanto, aumenta casi exponencialmente. Esto hace que las medidas de seguridad tengan que verse incrementadas y reforzadas, de manera que el cliente quede seguro de que no se va a corromper la integridad de sus datos, así como de que se cumplirá la Ley Orgánica de Protección de Datos.

- Que la información esté en una tercera empresa supondría un peligro debido a los problemas de accesibilidad que pueden surgir teniendo en cuenta que dicho acceso se hace a través de la red y que ésta, por tanto, puede fallar. La productividad de una empresa que utilice Cloud Computing, en un determinado momento, puede verse afectada si Internet no les funciona o si la empresa proveedora de los servicios tiene problemas con sus propios servidores, lo que podría acarrear una pérdida importante de tiempo y dinero.

- Es necesario que se adapte la forma de distribución en algunos casos, como son las licencias de software. Una aplicación que sea utilizada mediante Cloud Computing funciona sobre varios servidores, siendo este número variable según las necesidades del cliente, por lo que no se puede pagar como se ha venido haciendo hasta ahora por el número de versiones utilizadas.

- Se comentó que un servidor podría tener múltiples aplicaciones ejecutándose en él y que una aplicación podría estar en varios servidores. Para que la última opción sea posible, es necesario que se programen aplicaciones que puedan funcionar bajo esta condición.

El Cloud Computing cuenta con muchos defensores que lo tachan como la nueva forma de utilizar los recursos TI dentro de las empresas. Hugh Macleod dice así: "*El Cloud Computing es la verdadera batalla importante en este momento en la escena tecnológica: las compañías que dominen la nube serán los verdaderos actores del futuro, con esquemas de contratación muy importantes debido a la misma naturaleza de la actividad*".

El ahorro de dinero por parte de las empresas consumidoras está siendo, de igual modo, uno de los frentes a favor con más fuerza. Xabier Ormazábal, responsable de preventa de *Salesforce.com* comenta: "*El acceso y consumo de los servicios TI a través del Cloud Computing permite desplegar soluciones con una gran rapidez, debido a que las empresas ya no necesitan realizar grandes inversiones ni abordar proyectos de puesta en marcha de complejos sistemas de hardware y software en sus propias instalaciones. En definitiva, el modelo cloud facilita a las empresas de todo tamaño y sector focalizar sus recursos en optimizar sus procesos, liberándolos del mantenimiento, actualización y amortización de grandes inversiones tecnológicas en sistemas, que con frecuencia son menos eficientes y están infrautilizados dentro de cada organización*".

Ormazábal, por otra parte, resalta la gran capacidad de adaptación de estos sistemas en cualquier tipo de empresa: "*este tipo de soluciones está indicado para empresas de cualquier tamaño. Para las más pequeñas este procedimiento de gestión empresarial les da la posibilidad de acceder a las ventajas que ofrecen las tecnologías pero sin realizar una fuerte inversión en infraestructuras que no pueden afrontar. Por otro lado, las grandes compañías buscan, cada vez más, un valor añadido y una mayor efectividad en las soluciones que incorporan. Y a través del Cloud Computing pueden conseguirlo. Debido al número de usuarios en sus despliegue las grandes empresas son con frecuencia los mayores beneficiarios*".

Sin embargo, no todo el mundo está a favor de esta tecnología, como todas las novedades tiene también detractores. Para algunos, como Richard Stallman, el Cloud Computing no es lo que las grandes empresas intentan vender, sino una forma más de negocio para alimentar su ya abundante cartera de clientes dentro del software propietario y de pago: "*Una de las razones por las que no debes usar aplicaciones Web para tus tareas de TI, es que pierdes el control. Tú debes estar en condiciones de realizar tus propias tareas en tu propio PC, en un programa amante de la libertad. Si usas un programa propiedad de un proveedor, o un servidor Web de otra persona, entonces quedas indefenso… El Cloud Computing es una trampa*".

2.3.7. Ejemplos de Cloud Computing

- **Google Apps**.

Google Apps es un servicio proporcionado por Google. Pone a disposición de los clientes aplicaciones como el servidor de correo GMail, la herramienta de conversación instantánea Google Talk, Google Calendar y la herramienta de trabajo colaborativo Google Docs, que aporta un editor de textos, hojas de cálculo y un programa para realizar presentaciones.

- **Google App Engine**.

Google App Engine, cuya primera versión es del año 2008, es una plataforma para desarrollar y alojar aplicaciones Web en centros de procesamiento de datos gestionados por el propio Google. Actualmente soporta los lenguajes de programación Python y Java, aunque Google ha anunciado que está trabajando para que se puedan utilizar más lenguajes.

Este sistema posee algunas restricciones, entre las que se cuenta que los desarrolladores tienen acceso únicamente de lectura al sistema de archivos de App Engine, que se ejecuta sólo con llamadas HTTP, que el almacenamiento es limitado y que las aplicaciones Java no pueden crear threads.

- **Amazon Web Services**.

Amazon Web Services (AWS) es una colección de servicios vía Web. La mayoría de estos servicios están destinados a desarrolladores que puedan encontrar aquí una ayuda a la hora de realizar sus soluciones informáticas.

Algunos de estos servicios son:

- Amazon Elastic Compute Cloud (EC2): A través de este servicio, lanzado en su primera versión en 2006, se pueden alquilar máquinas en las que el cliente hace funcionar sus aplicaciones. Ofrece un servicio escalable, permitiendo al cliente gestionar las máquinas virtuales que utilizará para lanzar el software deseado. El cliente puede crear, lanzar y eliminar instancias en el servidor según sea necesario, pagando por el número de instancias y servidores activos. La última versión consta del año 2008, e incluye Service Level Agreement (SLA) para acordar con el cliente las necesidades que ofrecerán, una consola para la gestión de AWS y un plan para balancear la carga, autoescalable y con monitorización a través de Internet.

- Amazon Simple Storage Service (Amazon S3): Amazon S3 ofrece un servicio online para el almacenamiento de datos. Provee espacio ilimitado

a través de una sencilla interfaz gráfica. Como reclamo publicitario, Amazon asegura que dicha empresa utiliza este mismo servicio de almacenamiento para su propia red de e-commerce.

- Amazon Simple DB: Este servicio ofrece una base de datos distribuida escrita en Erlang. Se paga por almacenamiento, transferencia de datos y tasa de trasferencia a través de Internet.

- **Azure de Microsoft**.

Microsoft Windows Azure es una plataforma de Cloud Computing que provee una amplia variedad de servicios para que los desarrolladores puedan crear sus aplicaciones y lanzarlas en la red. Consiste en una especie se sistema operativo llamado Windows Azure que permite la ejecución de aplicaciones y provee herramientas para el desarrollo, gestión y almacenamiento de éstas, pudiendo utilizarse individualmente o todas juntas.

2.4. Arquitectura de software

2.4.1. Introducción

Cuando se va a desarrollar un producto software, la cantidad de personas implicadas es inmensamente grande: cliente, analistas, diseñadores, técnicos, programadores, etc. Además de contar con posibles incorporaciones tardías o que recojan el relevo, por ejemplo, para el mantenimiento del nuevo sistema.

Es necesario, por tanto, que cuando se esté hablando del nuevo software haya un total entendimiento entre todas las partes, evitando así malentendidos que pueden acabar en retrasos en el tiempo, pérdidas de dinero y pérdida de confianza por parte del cliente, lo que no debería permitirse.

En los años sesenta, Edsger Dijkstra introdujo el concepto de arquitectura del software, afirmando que ésta importaba y que era crucial elegir la estructura correcta. Sin embargo, hasta los noventa no se tomó realmente como algo necesario, convirtiéndose en una medida que evita, o disminuye en gran medida, los problemas anteriormente comentados.

Existen numerosas definiciones de arquitectura de software, y ninguna está reconocida oficialmente. De hecho, existen páginas que se han dedicado a reunir las definiciones existentes acerca del tema, como por ejemplo algunas definiciones modernas [39] o las encontradas en la bibliografía sobre esta cuestión [40]. Algunas de estas definiciones son:

- Definición oficial del IEEE, IEEE Std 1471-2000.

"*La Arquitectura del Software es la organización fundamental de un sistema formada por sus componentes, las relaciones entre ellos y el contexto en el que se implantarán, y los principios que orientan su diseño y evolución.*"

Este estándar también explica algunos términos de esta definición:

"*Un sistema es una colección de componentes organizada para cumplir una función o conjunto de funciones. El término sistema abarca aplicaciones individuales, sistemas en el sentido tradicional, subsistemas, sistemas de sistemas, líneas de productos, familias de productos, empresas enteras, y otras agregaciones de interés. Un sistema existe para satisfacer una o más misiones en su entorno.*"

"*El entorno, o contexto, determina el marco y las circunstancias del desarrollo, operaciones, política, y otras influencias sobre ese sistema.*"

- Cita del libro "*Software Architecture in Practice*". [41].

"*La arquitectura software de un programa o sistema de computación es la estructura o estructuras del sistema, que comprenden elementos software, las propiedades externas visibles de esos elementos y las relaciones entre ellos.*

[..]

[Las implicaciones de esta definición son:]

Primero, la arquitectura define elementos. La arquitectura expresa información sobre cómo los elementos se relacionan entre sí. Esto quiere decir que la arquitectura específicamente omite cierta información sobre los elementos que no se refiere a su interacción. […]

Segundo, la definición deja claro que los sistemas pueden y deben constar de más de una estructura y que ninguna estructura es irrefutablemente la única arquitectura. […]

Tercero, la definición implica que todo sistema software tiene una arquitectura porque todo sistema puede ser mostrado como una composición de elementos y relaciones entre ellos. En el caso más trivial, un sistema es un único elemento – una poco interesante y probablemente poco útil arquitectura, pero una arquitectura a pesar de todo. […]

Cuarto, el comportamiento de cada elemento es parte de la arquitectura en la medida que ese comportamiento puede ser observado o discernido desde el punto de vista de otro elemento. Este comportamiento es lo que permite a los elementos interactuar entre ellos, lo cual es claramente parte de la arquitectura. Esto no significa

que el comportamiento exacto de cada elemento tenga que ser documentado en todas las circunstancias; pero hasta el punto que un comportamiento de un elemento influya en cómo otro elemento debe ser escrito para interactuar con él o incluye en la aceptabilidad de un sistema completo, este comportamiento es parte de la arquitectura."

- [Lane, 90].

"*Arquitectura de software es el estudio de la estructura a gran escala y del comportamiento de un sistema software. Aspectos importantes de una arquitectura de software incluyen la división de las funciones entre los módulos, el significado de las comunicaciones entre los módulos, y la representación de la información compartida.*"

- [Perry, 92].

"*Distinguimos tres clases diferentes de elementos arquitectónicos: elementos de procesamiento; elementos de datos; y elementos de conexión. Los elementos de procesamiento son aquellos componentes que facilitan la transformación en los elementos de datos; los elementos de datos son aquellos que contienen la información que es usada y transformada; los elementos de conexión (que a veces pueden ser de procesamiento o de datos, o ambos) son el pegamento que mantiene las diferentes piezas de la arquitectura unidas.*"

Estas definiciones se pueden resumir en que una arquitectura es una especificación de alto nivel de un sistema software, en el que es primordial la representación de los elementos de dicho sistema y las relaciones existentes entre los mismos, así como su comportamiento. Gracias a esta especificación, todos los interesados en el producto pueden trabajar bajo una idea común, permitiéndoles alcanzar sus objetivos de una manera más sencilla.

Se puede resumir su propósito en los siguientes puntos [42]:

- Definir los módulos principales.
- Definir las responsabilidades que tendrá cada uno de estos módulos.
- Definir la interacción que existirá entre dichos módulos.
- Control y flujo de datos:
  - Secuenciación de la información.
  - Protocolos de interacción y comunicación.
  - Ubicación en el hardware.

Según Mary Shaw y David Garlan, en el "First International Workshop on Architectures for Software Systems", celebrado en 1995, existen cuatro tipo de modelos:

- Modelos estructurales: La arquitectura software está compuesta de componentes, las conexiones entre esos componentes, y normalmente, otros aspectos como la configuración, el estilo, restricciones, semánticas, propiedades, necesidades de los stakeholders, etc.

- Modelos de framework: Prestan especial atención en la (normalmente única) estructura coherente del sistema completo, en vez de centrarse en su composición.

- Modelos dinámicos: Se centran principalmente en el comportamiento de los sistemas. Con dinámico se puede referir a los cambios en la configuración del sistema, o a los cambios en los datos, etc.

- Modelos de proceso: Ponen su énfasis en la construcción de la arquitectura, y los pasos o el proceso involucrado en esa construcción.

Para decidir qué arquitectura es la que va a diseñarse para el sistema, es necesario que se reúnan los requisitos que debe cumplir el software, tanto los funcionales como los no funcionales, siendo ejemplos de estos últimos la mantenibilidad, flexibilidad, etc.

2.4.2. Representación de la arquitectura de software

La descripción de la arquitectura de software suele hacerse mediante la representación de lo que se llaman vistas, una forma de ver el diseño desde las diferentes apreciaciones de los interesados, o stakeholders. Cada una de estas vistas define o reproduce una parte de la arquitectura. Las vistas pueden solaparse entre ellas, siendo necesario que exista coherencia entre las mismas, pues al fin y al cabo representan el mismo sistema software.

Es importante, sin embargo, que cada una de las vistas, independientemente de cómo se representen, vengan acompañadas de una descripción textual en la que se explique el por qué de las decisiones tomadas respecto a la arquitectura seleccionada o el propio diseño.

De estas vistas, las más comunes son:

- Vista conceptual.

  La vista conceptual es usada para representar los requisitos funcionales y la vista que los clientes del negocio tienen de la aplicación, describiendo el modelo de negocio que la arquitectura debe cubrir [43]. Dicho con otras palabras, mediante esta vista se puede comprobar qué es lo que debe hacer el sistema a desarrollar, cuáles son las funcionalidades de las que el usuario final podrá hacer uso.

  Este tipo de vistas se suelen plasmar en Diagramas de Casos de Uso, Diagramas de Actividad, etc.

- Vista lógica.

  La vista lógica se encarga de representar los elementos del sistema y sus relaciones. No se tendrán en cuenta aspectos más técnicos, ni la futura implementación del sistema [43]. Este modelo suele dividir los elementos en paquetes, agrupándolos en función de distintos modelos ya estandarizados.

- Vista física.

  La vista física especifica cómo se va a ejecutar la aplicación, distribuyendo el procesamiento entre los equipos implicados en la ejecución del sistema, incluyendo los servicios y los procesos base. [43].

- Vista de implementación.

  En esta vista se describe cómo se implementan los distintos elementos físicos, agrupándolos en subsistemas, cada uno de ellos con sus propias capas y jerarquías, junto con las dependencias entre ellos [43].

Existen más vistas incluidas dentro de marcos arquitectónicos desarrollados por diferentes empresas u organismos. Este tema se tratará en el punto siguiente 2.4.3 Frameworks arquitectónicos.

Hay numerosas formas de describir estas vistas. Siempre suele hacerse uso del lenguaje natural, aunque sea como medio para explicar cualquier otro tipo de modelización. Lo que tratan de explicar estos lenguajes de representación son los elementos principales del sistema, y las relaciones y comunicaciones entre ellos.

Dentro del gremio de los arquitectos de software, sin embargo, se ha llegado a un consenso informal para utilizar la notación UML (Unified Modeling Language), la cual se explica de manera más detallada en el punto 2.4.5 UML (Unified Modeling Language).

Todas las vistas deben cumplir una serie de características para que puedan desempeñar su función:

- Tiene que describir el área deseada en detalle. No debe tener ambigüedades, pues el objetivo es que pueda entenderlo una tercera persona.

- El diseño debe seguir los estándares escogidos rigurosamente, de manera que cualquier persona pueda entender el modelo sin tener que preguntar al arquitecto original.

- No se debe sobrecargar un diagrama con demasiada información, pues puede confundir al destinatario. Es mejor realizar varios diagramas con la información necesaria en cada uno de ellos.

- La idea es comunicar al receptor o receptores del diseño la idea que se tiene para la construcción del sistema. Es una buena técnica para recibir un feedback de los interesados y así comprobar que el trabajo está bien realizado.

2.4.3. Frameworks arquitectónicos

Existen varios frameworks reconocidos que agrupan, de una manera u otra, un conjunto de vistas para la representación de la arquitectura de software. Uno de los más famosos es el framework "4+1", aunque como se puede ver en el cuadro siguiente, hay una gran variedad diferentes de ellos [44].

RM-ODP	4+1	[BRJ 99]	Microsoft
Empresa	Lógica	Diseño	Lógica
Información	Proceso	Proceso	Conceptual
Computación	Física	Implementación	Física
Ingeniería	Desarrollo	Despliegue	
Tecnología	Casos de uso	Casos de uso	

Tabla 1: Vistas en los marcos de referencia

- **RM-ODP** (Reference Model of Open Distributing Procesing)

El Modelo de Referencia para Procesamiento Distribuido Abierto especifica un marco para grandes sistemas distribuidos. Define cinco vistas para un sistema y su entorno, que no corresponden a etapas de proceso de desarrollo o refinamiento. Estas vistas son empresa, información, computación, ingeniería y tecnología.

- **4+1**.

Este modelo, de Philippe Kruchten, está vinculado al Rational Unified Process (RUP). Define cuatro vistas diferentes:

- Vista lógica: describe el modelo de objetos.
- Vista de proceso: muestra la concurrencia y sincronía de los procesos.
- Vista física: muestra la ubicación del software en el hardware.
- Vista de desarrollo: describe la organización del entorno de desarrollo.
- Una quinta vista con una selección de casos de uso o escenarios, elaboradas a partir de las vistas anteriores.

- **[BRJ 99]**.

Modelo propuesto por Grady Booch, James Rumbaugh e Ivar Jacobson. Definen cinco vistas interrelacionadas que conforman la arquitectura de software. Las vistas que la componen son:

- Vista de diseño: Comprende las clases, interfaces y colaboraciones que forman el vocabulario del problema y su solución.
- Vista de proceso: Conforman los hilos y procesos que forman los mecanismos de sincronización y concurrencia.
- Vista de implementación: Incluye los componentes y archivos sobre el sistema físico.
- Vista de despliegue: Comprende los nodos que forman la topología de hardware sobre la que se ejecuta el sistema.
- Vista de casos de uso: Muestra el sistema como lo perciben los usuarios, analistas y encargados de las pruebas.

- **Microsoft**.

Microsoft define cuatro vistas, llamadas ocasionalmente arquitecturas: negocios, aplicación, información y tecnología. Cada arquitectura, a su vez, se descompone en vistas, que son:

- Vista lógica: Define los componentes funcionales y su relación en el sistema.
- Vista conceptual: Cercana a la semántica de negocios y a la percepción de los usuarios no técnicos.
- Vista física: Ilustra los componentes específicos de una implementación y sus relaciones.

2.4.4. Arquitecturas más comunes

Como se mencionó anteriormente, la arquitectura del software se viene especificando desde los años noventa, cuando los arquitectos y desarrolladores se dieron cuenta de que se necesitaba una forma de representar el sistema a un nivel de detalle alto, para el correcto desarrollo posterior del mismo. Desde entonces, se han estudiado diferentes arquitecturas, de tal forma que cuando se construya un nuevo sistema, no se tenga que partir de cero a la hora de definir su propia arquitectura. Estos modelos, son patrones ya estudiados y de probado funcionamiento y efectividad, y se pueden adaptar a las condiciones de determinados

sistemas. Evidentemente, es posible hacer variaciones para cada software diferente, aunque en su base son iguales.

Los más comunes y usados son:

- Arquitectura cliente – servidor [48].

Esta arquitectura se caracteriza por tener dos partes claramente diferenciadas: una que provee una serie de servicios (el servidor), y una serie de procesos que actúan como cliente para ese servidor.

Los distintos componentes no tienen por qué ejecutarse en la misma máquina, siendo habitual que se comuniquen entre ellos a través de la red. La parte del cliente se ejecuta en la máquina del usuario, para optimizar su uso, mientras que el servidor suele estar en una máquina remota.

Los servidores pueden tener dos funciones no necesariamente excluyentes:

- Servir como centro de la base de datos de la aplicación, alojando los archivos necesarios para los clientes, permitiendo su consulta, modificación, borrado, etc.
- Funcionar como servidores de aplicaciones, alojando distintos programas que pueden ser ejecutados desde la máquina del cliente.
- También pueden realizar ambas funciones.

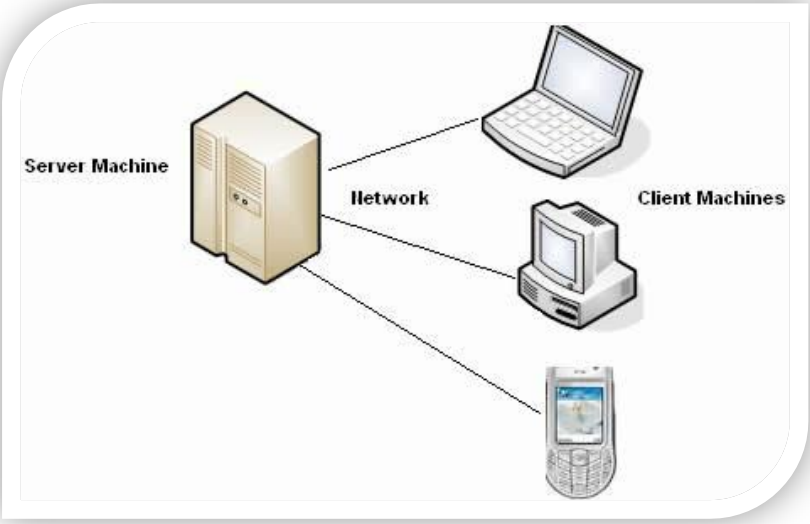

Imagen 22: Estilos arquitectónicos: cliente - servidor

- Arquitectura en pipes y filters [47].

Esta arquitectura se identifica porque cada componente tiene un conjunto de entradas y salidas. El componente lee un flujo de datos en la entrada, lo transforma, y produce otro flujo de datos en la salida, entregando una instancia completa del resultado en un orden estándar. Los componentes se llaman filtros debido a la modificación que hace sobre los datos, mientras que los conectores entre esos filtros se llaman tuberías, o en inglés "pipes".

Los filtros son entidades independientes que no tienen conocimiento de otros elementos de la arquitectura. No saben dónde dirigen los flujos de datos, simplemente conocen el tratamiento que deben darle.

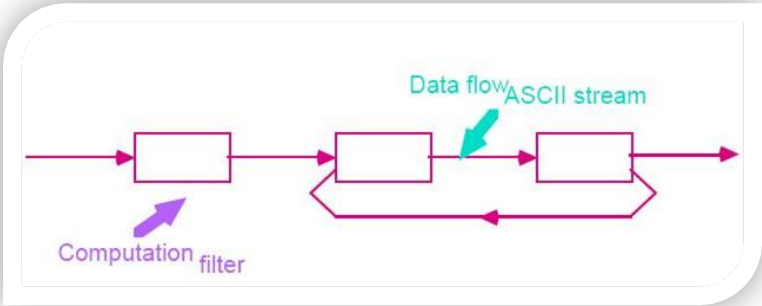

Imagen 23: Estilos arquitectónicos: pipes and filters

Uno de los ejemplos más extendidos que utilizan este tipo de arquitecturas son los sistemas Unix, al igual que la programación funcional y los sistemas distribuidos.

Algunas de las ventajas de utilizar este modelo son:

- Permiten al diseñador comprender fácilmente el conjunto global de entradas y salidas del sistema.
- Los filtros soportan la reutilización, cualquier pareja de filtros puede ser conectada, siempre que estén de acuerdo en los datos que se transmiten entre ellos.
- Los sistemas pueden ser fácilmente mantenidos y mejorados, pudiendo añadir o quitar filtros, así como modificar los ya existentes.
- Soportan la ejecución concurrente al estar los filtros implementados como entidades independientes que pueden ejecutarse en paralelo con otros filtros.

Este sistema también tiene algunas desventajas:

- En algunas ocasiones, este modelo induce a una ejecución secuencial. Esto sucede porque al diseñarlos se puede provocar que se necesite cierta transformación de los datos antes de pasar al siguiente filtro.

➢ Arquitectura basada en eventos [47].

Esta técnica se basa en contestar a llamadas implícitas de los procedimientos y funciones del sistema, sin tener que hacerlo directamente. Para que funcione, un componente anuncia uno o más eventos. Los demás componentes se registran en los eventos que les interesan, aquellos ante los que deben responder, asociando éstos a un procedimiento determinado. Cuando el evento registrado es anunciado, es el propio sistema el que llama al procedimiento y lo ejecuta.

Los componentes de este modelo son módulos cuyas interfaces proporcionan tanto una colección de procedimientos como un conjunto de eventos. Los procedimientos se pueden llamar de un modo tradicional, mediante llamadas explícitas, aunque en este caso, un componente también puede registrar algunos de sus procedimientos con un evento del sistema.

Aquellos elementos que notifican la ocurrencia de un evento no saben qué componentes van a responder, por lo que no se puede saber cuál va a ser el resultado del propio evento.

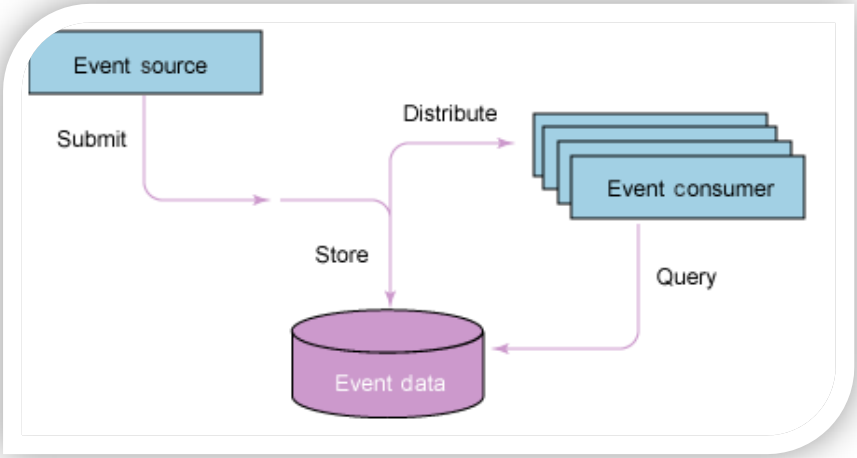

Imagen 24: Estilos arquitectónicos: basado en eventos

Sistemas que se aseguran de la consistencia de las bases de datos y sus restricciones, aplicaciones con interfaces de usuario, etc., son sistemas que usan este tipo de arquitecturas.

Una de las mayores ventajas que aporta la programación basada en eventos es que es fácil de reutilizar, pues resulta sencillo introducir componentes que aporten nuevos procedimientos y eventos. Además, también se pueden modificar unos componentes por otros, mejorando sin mucho esfuerzo los ya existentes.

La más clara desventaja es la pérdida del control al no hacer las llamadas a las funciones de manera explícita. Es más, puesto que no se sabe qué componentes van a responder al evento, no se sabe si ha habido alguno que no haya realizado su función y no ha invocado al procedimiento correspondiente.

- Arquitectura en repositorio [47].

En el estilo en repositorio hay dos tipos de componentes: una estructura central de datos que representa el estado actual, y una colección de componentes independientes que operan con dicha estructura central.

Hay dos clases básicas de repositorio: si las transacciones son las que hacen que se desencadene el proceso a ejecutar, se trata de una base de datos tradicional; si por el contrario, es el estado de la estructura de datos central el que provoca la ejecución de un proceso u otro, entonces el repositorio se llama "*blackboard*" o pizarra. Este último tipo tiene tres partes diferenciables:

- Las fuentes del conocimiento: Parcelas independientes donde se tiene el conocimiento. Estas fuentes sólo interactúan con la *blackboard*.
- La estructura de datos blackboard. Es donde se encuentra el estado de los datos. Las fuentes del conocimiento pueden hacer cambios sobre ella.
- Control: Actúa según el estado de la blackboard.

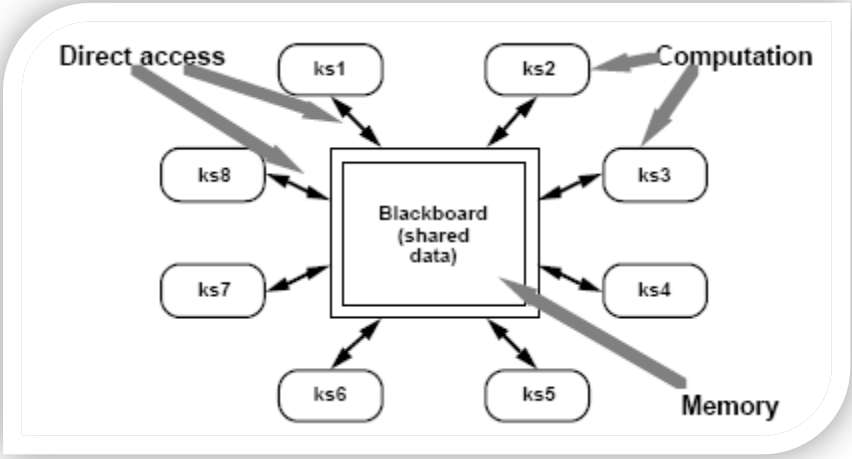

Imagen 25: Estilos arquitectónicos: repositorio blackboard

- Sistemas en capas [47].

Un sistema en capas está organizado jerárquicamente. Cada una de las capas provee un servicio a la capa superior y funciona como cliente con la capa inmediatamente inferior.

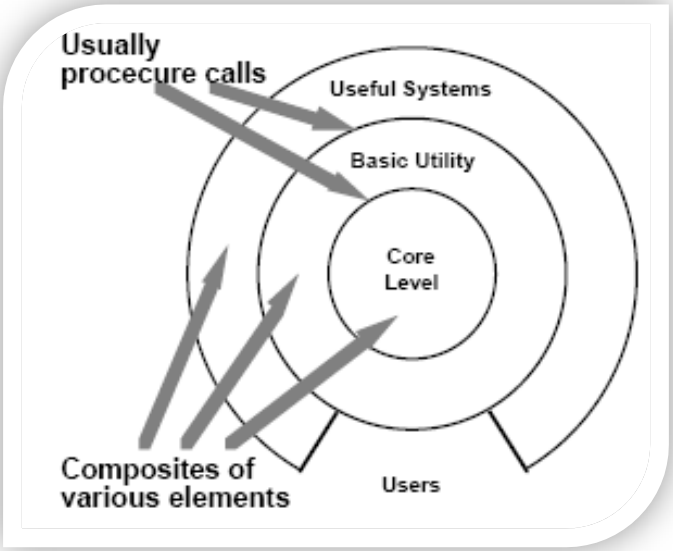

Imagen 26: Estilos arquitectónicos: sistema en capas

El ejemplo por excelencia de este tipo de arquitecturas son los protocolos de comunicación basados en capas, como el protocolo TCP/IP.

Como ventajas cabe destacar:

- Soportan un diseño basado en niveles de abstracción, lo que proporciona la posibilidad de resolver problemas complejos como una secuencia de pasos.
- Es fácilmente mejorable y mantenible, pues el cambio en una de las capas, como mucho, afecta a las otras dos con las que mantiene un contacto directo.
- Las capas pueden ser intercambiables para ofrecer distintos servicios. Una opción es definir un estándar en cuanto a las entradas y las salidas, de tal forma que las capas puedan modificarse sin tener que afectar al resto.

El problema de este modelo es que no todos los sistemas se adaptan a esta arquitectura. Además, es complicado encontrar el número correcto de niveles, pues tiene que ser lo suficientemente grande para dividir el trabajo, pero lo suficientemente pequeño para no sobrecargar el sistema en demasiadas operaciones.

- Modelo – Vista – Controlador [49], [50].

Este modelo consiste en la división de la aplicación en tres capas, cada una con una función:

- Modelo: Es el componente que representa la información del mundo real y encapsula los datos y el comportamiento que son independientes de la presentación. Es el responsable de realizar las consultas a la base de datos, ejecutar los cálculos de los procesos de negocio y de procesar las órdenes que le dé el usuario.

- Vista: Es el componente que se encarga de mostrar la información contenida en el modelo al usuario. Se puede tener varias vistas para cada tipo de usuario de la aplicación.

- Controlador: Sirve como una conexión lógica entre la interacción del usuario y los servicios de negocio disponibles en el modelo. Las órdenes le llegan desde la vista y traspasa esa información al modelo para que la procese. Posteriormente reenvía la respuesta de éste último de nuevo a la vista para que el usuario pueda apreciar los cambios acontecidos.

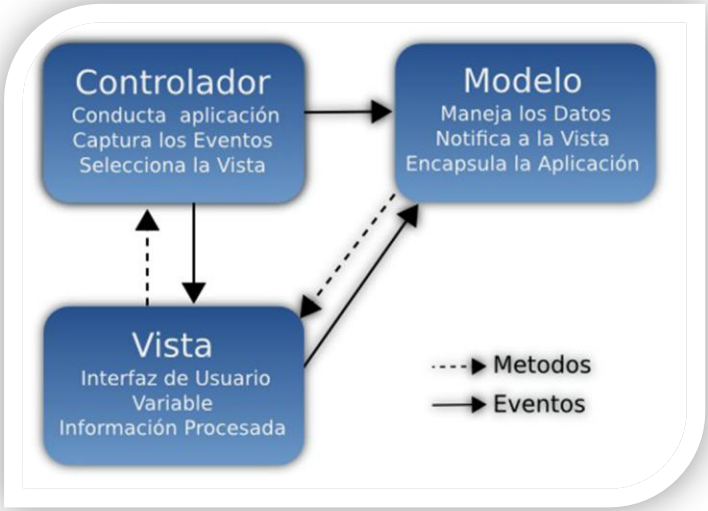

Imagen 27: Estilos arquitectónicos: Modelo Vista Controlador

2.4.5. UML (Unified Modeling Language)

El lenguaje de modelado UML es, a día de hoy, el más extendido y utilizado para la representación de los diseños de los sistemas software, desde diseños de alto nivel, como la arquitectura software, a diseños de más bajo nivel, como diagramas de clases.

En este apartado se pretende hacer una breve introducción a este lenguaje, mencionando los elementos básicos a partir de los cuales se pueden realizar los diagramas necesarios.

Los elementos básicos de las representaciones son [51]:

- Clases. Una clase es una agrupación de datos (variables o campos) y de funciones (métodos) que operan sobre esos datos. La clase es el patrón o modelo para crear objetos. Su representación es la siguiente:

Imagen 28: Representación en UML de una clase

- Casos de uso: Cada caso de uso proporciona uno o más escenarios que indican cómo debería interactuar el sistema con el usuario o con otro sistema para conseguir un objetivo específico. Su representación es:

Imagen 29: Representación en UML de un caso de uso

- Componentes: Es un grupo de objetos o partes más pequeñas que tienen relación entre sí y que se combinan para dar un servicio. En UML se pueden ver de la siguiente manera:

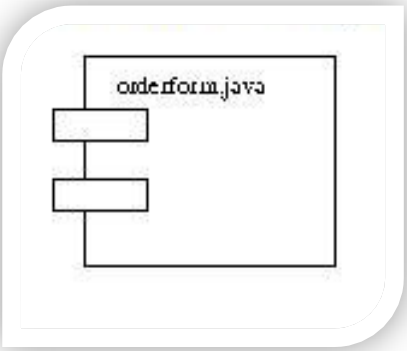

Imagen 30: Representación en UML de un componente

- Nodos: Un nodo es un conjunto de componentes. Se representa mediante:

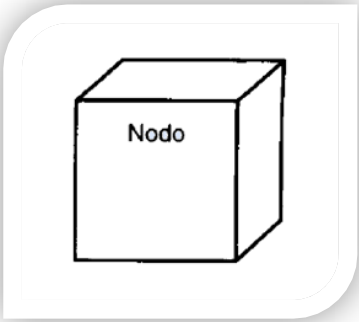

Imagen 31: Representación en UML de un nodo

- Paquetes: Es un grupo de elementos del modelo. Pueden tener otros paquetes, componentes, etc. Se representa:

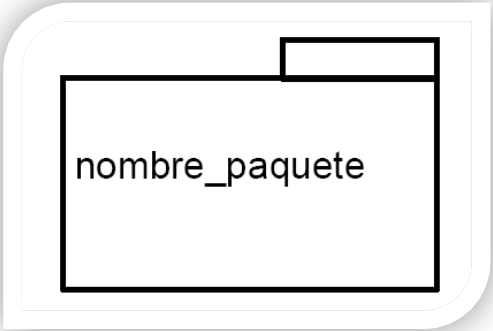

Imagen 32: Representación en UML de un paquete

Las conexiones entre los elementos pueden ser [51]:

- Dependencia: Es una relación semántica entre dos elementos, en la cual un cambio a un elemento (el elemento independiente) puede afectar a la semántica del otro elemento (el dependiente).

Imagen 33: Representación en UML de una dependencia

- Asociación: Es una relación estructural que describe un conjunto de enlaces, los cuales son conexiones entre objetos. La agregación es un tipo especial de asociación, que representa una relación estructural entre un todo y sus partes.

Imagen 34: Representación en UML de una asociación

- Generalización: Es una relación de especialización-generalización en la cual los objetos del elemento especializado (el hijo) pueden sustituir a los objetos del elemento general (el padre). De esta forma el hijo comparte la estructura y el comportamiento del padre.

Imagen 35: Representación en UML de una generalización

- Realización: Es una relación semántica entre clasificadores, en donde un clasificador especifica un contrato que otro clasificador garantiza que cumplirá. Se pueden encontrar relaciones de realización: entre interfaces y las clases o componentes que las realizan. Semánticamente la realización es una mezcla entre dependencia y generalización.

Imagen 36: Representación en UML de una realización

Los elementos y relaciones se agrupan en diagramas que representan diferentes aspectos del sistema:

- Diagrama de secuencia [53]: Muestran la interacción entre elementos, detallando de forma explícita la secuencia de estímulos ordenada temporalmente. Se utilizan para describir los distintos escenarios derivados de los casos de uso.

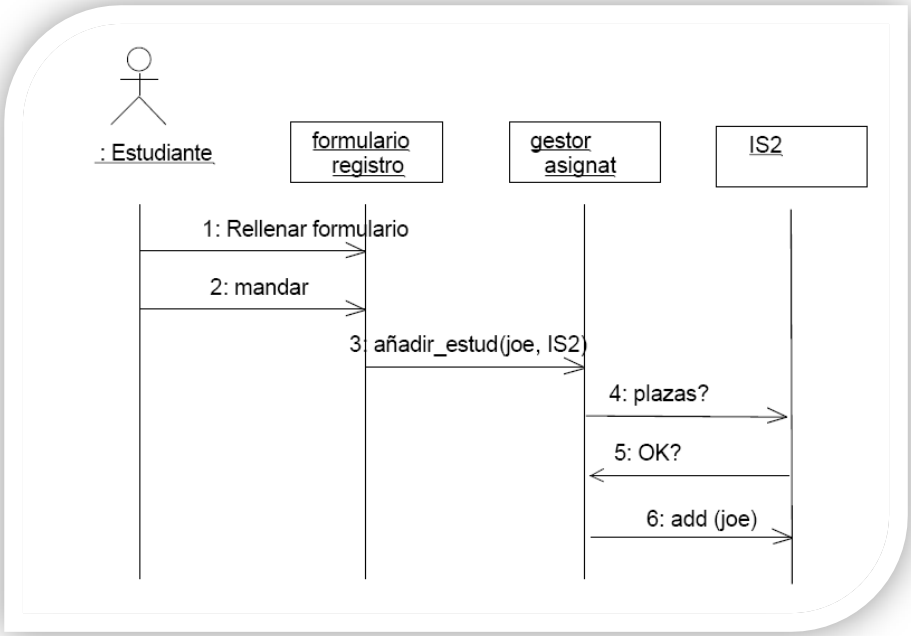

Imagen 37: Diagrama de secuencia

- Diagrama de clases [51]: Presenta las clases, junto con sus atributos, operaciones, interfaces y relaciones. Se presenta igualmente cómo se agrupan las clases en los distintos componentes y paquetes.

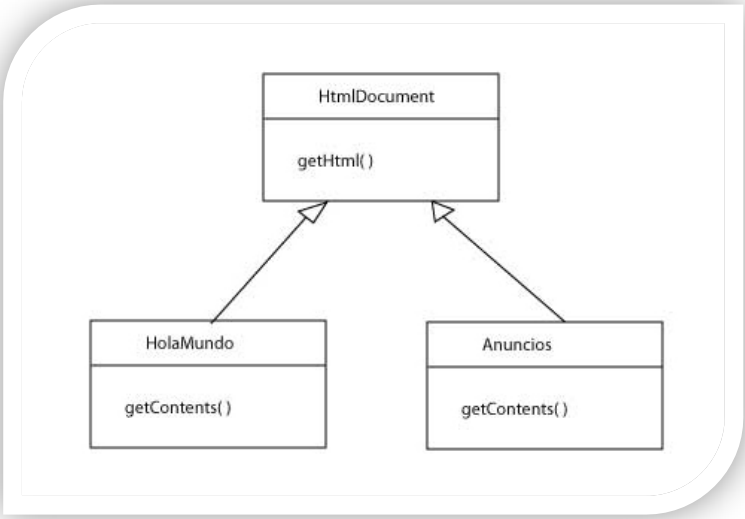

Imagen 38: Diagrama de clases

- Diagrama de objetos [51]: Muestra instancias de clases (objetos) con valores en sus atributos y relaciones.

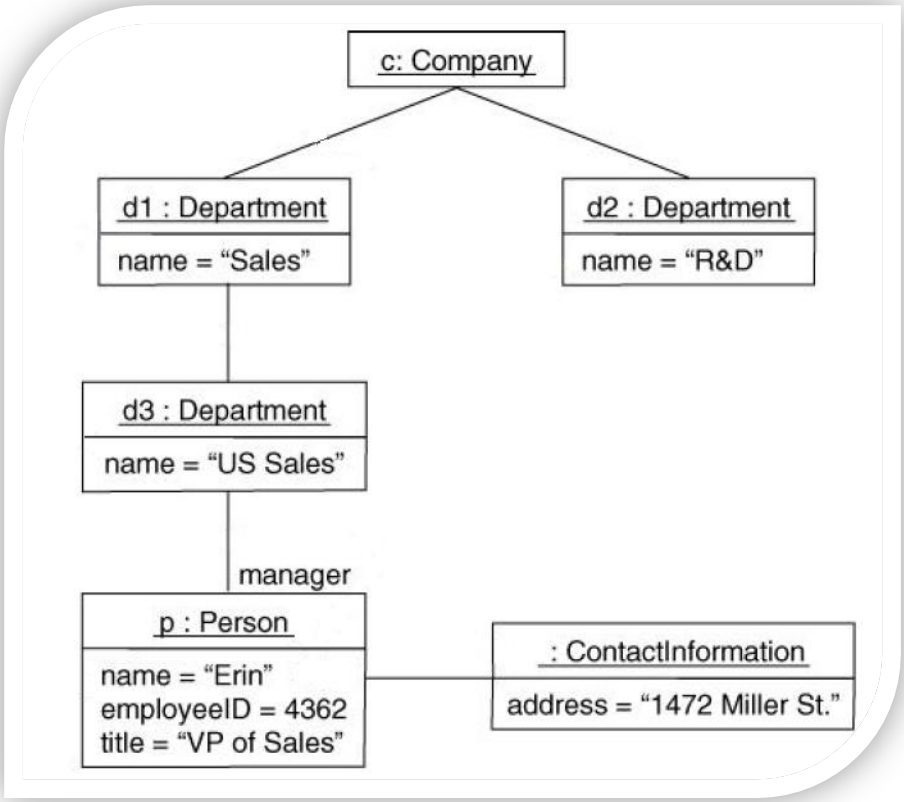

Imagen 39: Diagrama de objetos

- Diagrama de casos de uso [53]: Captura la funcionalidad del sistema vista por los usuarios.

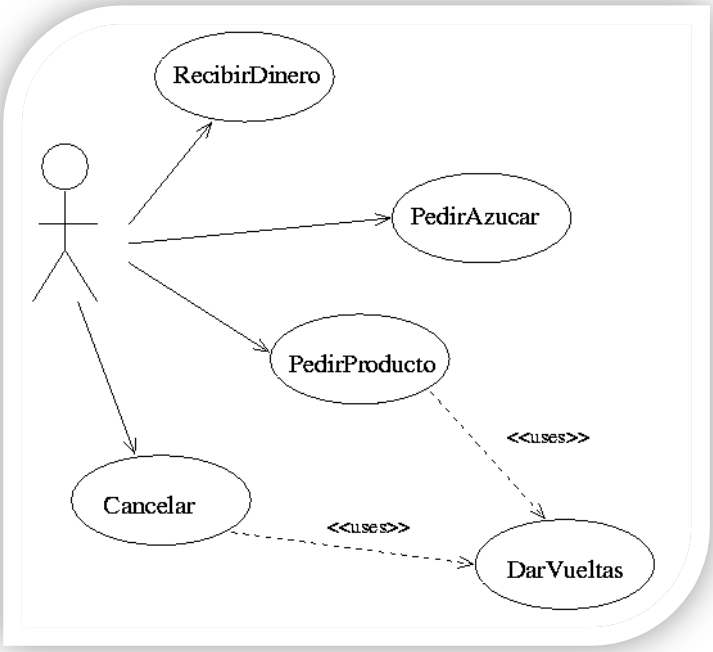

Imagen 40: Diagrama de casos de uso

- Diagrama de interacción [51]: Comprende los diagramas de secuencia y colaboración. Presenta objetos y relaciones entre ellos desde el punto de vista dinámico.

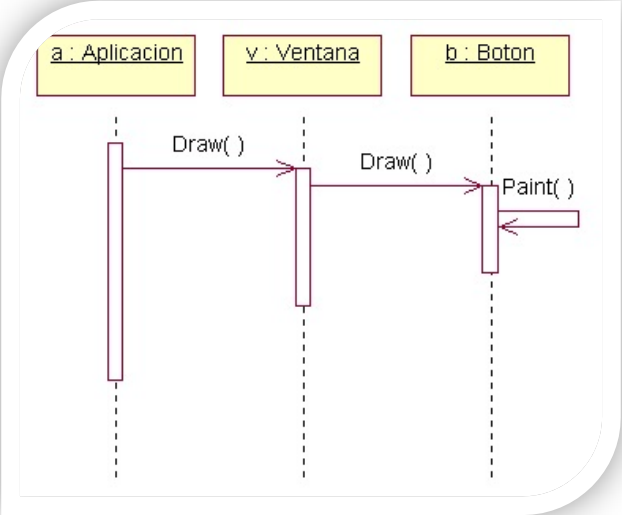

Imagen 41: Diagrama de interacción

- Diagrama de estado [51]: Representa los posibles estados, eventos y transiciones entre las clases u objetos.

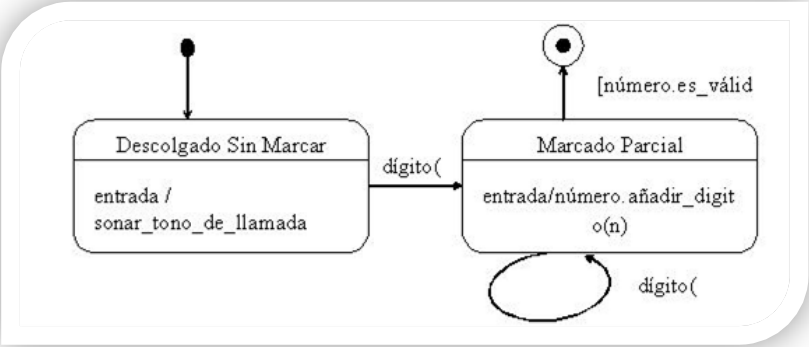

Imagen 42: Diagrama de estados

- Diagrama de componentes [51]: Muestra la organización y dependencia entre componentes físicos.

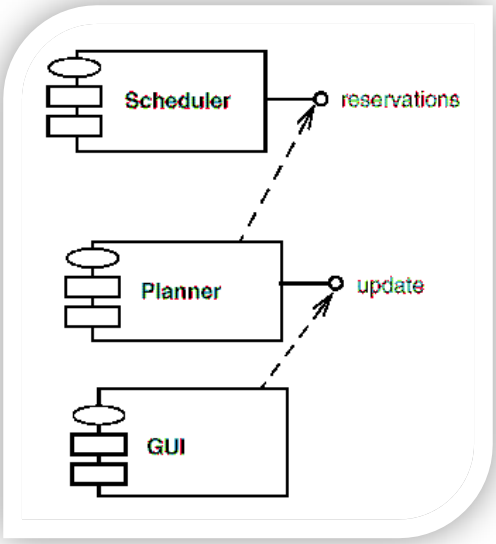

Imagen 43: Diagrama de componentes

- Diagrama de despliegue [51]: Muestra la distribución y comunicación de los componentes en los dispositivos hardware.

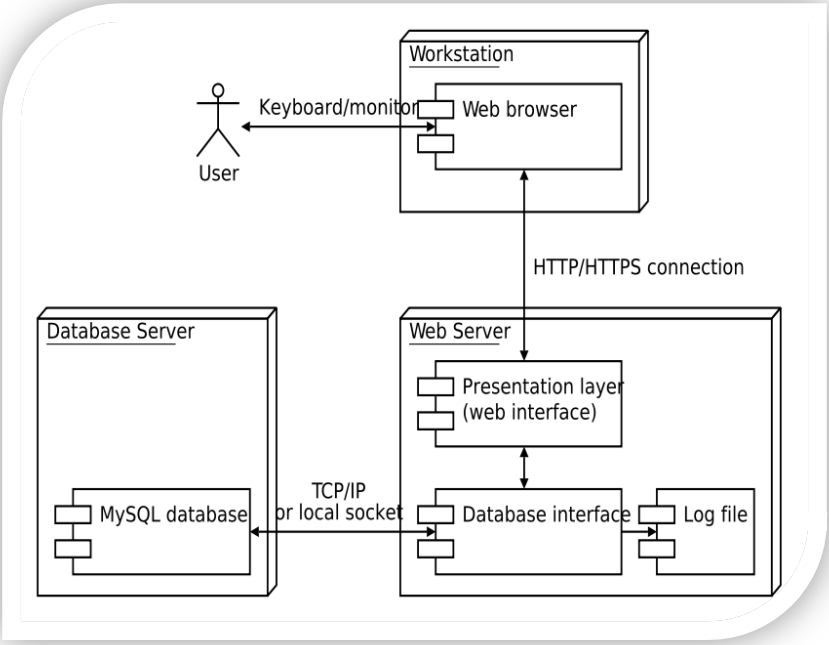

Imagen 44: Diagrama de despliegue

2.4.6. Ventajas de la Arquitectura Software

Desde que se empezaron a realizar los diseños de arquitectura software para el desarrollo de sistemas informáticos, se han podido ver las numerosas ventajas que proporciona. Paul

Clements y Linda Northrop en 1996 [45], y David Garlan en el año 2000 [46], escribieron sendos libros en los que exponían los beneficios de la arquitectura software. Algunas de sus conclusiones se presentan aquí [44]:

- Gracias a la arquitectura de software se expone cuál es el diseño de alto nivel del sistema, justificando textualmente las medidas tomadas. Así, los stakeholders implicados en el proceso podrán comprender, en mayor o menor medida, el proceso y sus decisiones, pudiendo compartir impresiones y comunicarse entre sí.

- La realización de la arquitectura software del sistema requiere que se medite sobre el diseño en una etapa temprana del proceso, evitando que se siga adelante si no están de acuerdo con el mismo todos los interesados.

- Al poder observar las diferentes vistas del sistema, se comprueban cuáles son las interfaces y conexiones que se necesitan a la hora de desarrollar el software, identificando así las restricciones y dependencias existentes.

- Una arquitectura determinada se puede reutilizar en sistemas que tengan objetivos similares. Así, se tienen una serie de patrones arquitectónicos que pueden adaptarse a cualquier caso particular que se pueda presentar.

- Al tener una visión de lo que será el sistema, se pueden establecer las zonas menos estables y que posiblemente estarán abiertas a una evolución del software, desarrollando dichas partes adecuadas a tal fin.

- La arquitectura de software permite estudiar si el análisis se ha hecho correctamente o si falta algún detalle no contemplado anteriormente, ahorrando errores posteriores que acarren dificultades mayores.

- La realización del diseño arquitectónico en primer lugar es una prueba más de la viabilidad del software, comprendiendo mejor de esta forma los requisitos, las estrategias de implementación y los riesgos potenciales.

2.5. Ontologías

2.5.1. Introducción

El término ontología apareció originalmente en la filosofía como una disciplina que estudia la naturaleza de la realidad y de lo que existe: qué es, cómo es y cómo es posible. En informática, una ontología es una especie de gran esquema conceptual en el que se recogen los términos y relaciones de un dominio concreto, así como ciertas restricciones y características propias de los términos dentro de ese dominio. Estas propiedades son las que dotan a las ontologías de un carácter semántico, dando un determinado significado a los

términos dependiendo del marco en el que se engloben: no es lo mismo hablar de descarga en el entorno de Internet que en el de los transportistas.

Los filósofos utilizaban las ontologías como medio para representar el mundo. Aquellos dedicados a la informática, sin embargo, las empezaron a usar en Inteligencia Artificial a mediados de los años setenta, pues intentaban crear una estructura con la que se pudiera conseguir un razonamiento automático: representaban el conocimiento acerca de un determinado dominio, y mediante sus reglas, se podían llegar a ciertas conclusiones de manera automática.

Una de las primeras definiciones oficiales surgió a principios de los años noventa, cuando Tom Gruber dijo que "*una ontología es una especificación explícita de una conceptualización compartida*". Sin embargo, como en casi todos los aspectos tecnológicos, existen numerosas definiciones, encontrando documentos que resumen algunas de ellas [60]:

- Según Neches [57]: "*Una ontología define los términos y las relaciones básicas para la comprensión de un área, así como las reglas para combinar los términos para definir las extensiones del vocabulario*".

- Según Guerrero y Lozano [58]: "*Las ontologías son construcciones que estructuran contenidos explícitos y que son capaces de codificar las reglas implícitas de una parte de la realidad, pese a trabajar con declaraciones explícitas independientes del fin y del dominio de la aplicación*".

- Según Quin y Palin [59]: "*Las ontologías muestran un alto nivel de especificación, un alto grado de flexibilidad, permiten fórmulas de distribución y reutilización, y están dispuestas para acomodar términos descriptivos variables*".

A pesar de la variedad de definiciones, sí se ha llegado a un consenso en cuanto a los elementos que componen las ontologías [54]:

- Clases o Conceptos: son las ideas a formalizar. Pueden ser clases de objetos, métodos, planes, estrategias, etc.

- Relaciones: representan las interacciones entre las clases o conceptos.

- Funciones: son relaciones donde se identifican elementos mediante el cálculo de una función.

- Instancia: son los objetos de una determinada clase.

- Axiomas: son teoremas declarados sobre relaciones que deben cumplir los elementos de una ontología. Gracias a éstos, y a la herencia de conceptos, se pueden realizar inferencias sobre los conceptos, generando el conocimiento de dominio.

Noy y McGuinness declaran como principales objetivos de las ontologías los siguientes [61]:

- Compartir la forma de entender la estructura de la información entre personas o agentes software. Esto facilitará la extracción y recuperación de la información en páginas Web.
- Permitir la reutilización del conocimiento sobre un dominio.
- Permitir hacer explícitos los supuestos de un dominio.
- Permitir analizar el contenido de un dominio concreto.

Según Francisco Javier Honrubia [54], el beneficio de las ontologías está claro:

- Proporcionan una forma de representar y compartir el conocimiento utilizando un vocabulario común.
- Permiten usar un formato de intercambio de conocimiento.
- Proporcionan un protocolo específico de comunicación.
- Permiten una reutilización del conocimiento, pues pueden ser incorporadas dentro de otras ontologías para aumentar el dominio representado.

Evidentemente no todo son facilidades y ventajas, existen problemas aún no solucionados [54], como la imposibilidad de construir ontologías grandes que abarquen un gran número de dominios, la subjetividad a la que están sujetas, pues quedan siempre desde el punto de vista del diseñador, y la falta de estándares para su construcción y codificación, lo que hace que su uso no sea tan extendido como pudiera ser.

2.5.2. Clasificación de ontologías

Existen diferentes maneras de clasificar las ontologías, dependiendo del criterio que se siga. Van Heist, por una parte, divide las ontologías según la cantidad y tipo de conceptualización. La división queda así [55]:

- Ontologías terminológicas: especifican los términos que son usados para representar conocimiento en el universo del discurso. Suelen ser usadas para unificar vocabulario de un dominio determinado.
- Ontologías de información: especifican la estructura de almacenamiento de bases de datos. Ofrece un marco para el almacenamiento estandarizado de información.

- Ontologías de modelado del conocimiento: especifican conceptualizaciones del conocimiento. Contienen una rica estructura interna y suelen estar ajustadas al uso particular del conocimiento que describen.

Guarino, por su parte, propone una clasificación basada en el alcance de aplicabilidad de la ontología. Así, diferencia los siguientes tipos [56]:

- Ontologías de dominio: ontologías específicas para un dominio concreto. Están destinadas a describir conceptos generales tales como el espacio, el tiempo, la materia, la acción, etc.

- Ontologías de tareas: representan las tareas que son susceptibles de realizar en un dominio concreto.

- Ontologías generales: van a representar los datos generales y no específicos de un dominio. Describen los conceptos conforme a un campo determinado o a unas tareas concretas.

2.5.3. Construcción de una ontología

El proceso de construcción de una ontología no está estandarizado, pese a que hay una rama de la ingeniería del conocimiento, la ingeniería de las ontologías, que se dedica a estudiar las mejores metodologías para ello. Aún así, Honrubia reconoce cuatro pasos que se deben seguir si se desea obtener una buena ontología [54]:

1. Identificar el propósito y el alcance.

   Durante esta fase se debe analizar cuál va a ser el punto de vista del modelo de la ontología, estudiándolo dentro del contexto de la aplicación. Este último describe el dominio, cuáles son los objetos de interés que deben representarse y cuál es la finalidad última de la ontología. Mediante el punto de vista se especifica qué tipo de modelo se va a usar, dinámico, estático, etc.

2. Construcción de la ontología.

   Este paso describe la propia construcción de la ontología. Tiene, a su vez, varias divisiones:

   - Captura de los objetos que van a representarse.

   - Codificación y representación de la conceptualización en un lenguaje formal de ontologías.

- Integración de las ontologías existentes, si es que se va a hacer uso de alguna ontología ya construida previamente.

3. Evaluación.

   En esta fase se debe valorar la ontología construida, teniendo en cuenta otros aspectos como la posible futura reutilización de la misma y si cubre todos los aspectos necesarios.

4. Documentación y reutilización.

   La documentación no es un paso final, si no que se debe hacer paralelamente a la construcción de la ontología, justificando las decisiones tomadas, los resultados de la evaluación, la captura del conocimiento, etc.

Honrubia, sin embargo, no se queda simplemente en la enumeración de estos pasos, sino que aconseja algunos principios a tener en cuenta [54]:

- Claridad y objetividad: se debe dar el significado de la ontología y sus términos de manera objetiva y mediante lenguaje natural para su mayor comprensión.
- Completitud: las definiciones han de ser expresadas en términos necesarios y suficientes.
- Máxima extensibilidad monótona: las especializaciones o generalizaciones deben ser incluidas en la ontología de tal forma que no se requiera una división de las definiciones preexistentes.
- Principio de distinción ontológica: las clases deben ser disjuntas.
- Diversificación: se han de diversificar las jerarquías incluidas para aumentar la potencia de los mecanismos de herencia múltiple.
- Modularidad: para disminuir el acoplamiento entre los módulos.
- Estandarización: siempre que sea posible se deben estandarizar los nombres.
- Minimización de la distancia semántica entre conceptos emparentados: conceptos similares estarán agrupados y representados utilizando las mismas primitivas.

Por otra parte, Noy y McGuinness proponen otra metodología distinta [61]:

1. Determinar el dominio y el alcance.
2. Considerar la reutilización de ontologías existentes.
3. Enumerar los términos importantes.

4. Definir las clases y su jerarquía.

5. Definir las propiedades de las clases.

6. Crear las instancias.

2.5.4. Las ontologías y la Web Semántica

La Web Semántica es el nuevo objetivo de los diseñadores de páginas Web. Consiste en dotar de significado a las páginas incluyendo información que indique cuál es su temática, los recursos que ofrece, etc. La expansión de esta nueva forma de entender la red provocará que sea más fácil intercambiar datos entre páginas, buscarlas, etc.

Un ejemplo: si ahora se realiza cualquier búsqueda en Internet, muchos de los resultados expuestos no tienen nada que ver con el significado de la consulta “Los diez mandamientos”. Es muy posible que gran parte de los resultados tengan que ver con la película de este mismo nombre en lugar de recuperar los propios mandamientos. Si se incluyera significado semántico a las páginas, se podría orientar la consulta a la manera de entender el dominio por parte del usuario.

La forma más común hoy en día de definir los recursos de la Web es mediante la inclusión de metadatos en las páginas. El objetivo de los metadatos es estructurar la información y enumerar los recursos que ofrece. Sin embargo, para conferir de semántica a la página es necesario incluir aún más información, de ahí el uso de las ontologías.

Gracias a ellas, los agentes no sólo encontrarán la información de manera más precisa, sino que podrán realizar inferencias sobre los recursos para ofrecer un mejor resultado al usuario. Se potenciará, sobre todo, la búsqueda y recuperación de información y las aplicaciones de comercio electrónico.

Los metadatos en formato RDF (ver el apartado 2.1.8.4 RDF) ya incluyen cierta información en las páginas, aunque para que el concepto de Web semántica se haga efectivo en su más amplio sentido es necesario que se profundice más en ese tipo de lenguajes. Para tal fin, existen algunas herramientas de ontologías ya desarrolladas, como *Protégé* o *WebOnto*, que permiten realizar anotaciones en las páginas con leguajes de marcado propios. Claro, que además de utilizar el lenguaje, es necesario que las aplicaciones Web cuenten con agentes informáticos capaces de reconocer esos lenguajes y su significado.

2.5.5. El uso de las ontologías para la enseñanza

Ya que las ontologías recogen conocimiento sobre un dominio, y permiten realizar razonamiento e inferencias automáticas sobre el mismo, uno de los posibles usos que pueden tener está en el campo de la enseñanza. Miguel Rodríguez ha propuesto en su artículo [63] lo que sería una ontología general para tal fin. Aquí se resumirá esa propuesta.

Según Rodríguez, el modelo consta de tres dominios:

- Dominio conceptual.

  Recoge los conceptos del dominio de la materia, así como sus relaciones estructurales y dependencias. Las entidades de este dominio son:

  - Concepto: el elemento de conocimiento básico para el razonamiento en el ámbito de la materia. Por ejemplo, predicado, variable libre, etc.
  - Actividad: es una acción que se realiza sobre un concepto. Por ejemplo, evaluar un predicado.

  Las relaciones del dominio son:

  - Relaciones estructurales: aquellas que clasifican las entidades del dominio en base a:
    - tipo-de: representan la jerarquía de clases. Proporcionan la respuesta a preguntas “Cuáles son los tipos de X”.
    - parte-de: representa la composición. Dan respuesta a preguntas como “Qué partes componen X”.
  - Relaciones dependientes del dominio del conocimiento:
    - induce: indica que, si existe un determinado elemento, es como consecuencia de la existencia de otro.
    - produce: el resultado de realizar una actividad sobre uno o varios conceptos, produce como consecuencia otro.
    - se realiza sobre: establece la relación entre la actividad y el concepto sobre el que se aplica.

- Dominio instruccional.

  Describe los objetos instruccionales asociados a los elementos del dominio conceptual. Éstos son aquellos que sirven para enseñar la materia, como problema, solución, etc. Las entidades que lo componen son:

- Ejemplo: una instancia de un concepto o de una actividad con el objetivo de facilitar la comprensión de éste.
- Error: un fallo que sea frecuente en la comprensión de alguno de los elementos del dominio.
- Pista: es una entidad que orienta hacia la solución de un problema.
- Explicación: complementa la información que se haya proporcionado sobre algún elemento del dominio, en general sobre una solución.
- Problema: el enunciado de un ejercicio que pone a prueba el conocimiento y grado de comprensión sobre uno o varios conceptos.
- Solución: los pasos asociados a la resolución de un problema.
- Pregunta/Respuesta: son entidades diferentes pero asociadas a un elemento expositivo con una cuestión acerca de éste y eventualmente con una respuesta.

Las relaciones son:

- es solución: enlaza una entidad problema con su solución.
- explica: aclara otros elementos, como una solución o un error.
- responde a: una respuesta responde a una pregunta.

➢ Dominio didáctico.

Clasifica los elementos y relaciones de los dominios anteriores teniendo en cuenta sus propiedades pedagógicas, incorporando nuevos atributos y relaciones. Así, por ejemplo, un problema puede ser fácil o difícil, o es requisito indispensable saber un enunciado A antes que otro B. No añaden nuevas entidades. Aquí se definen:

- Atributos didácticos: se definen sobre las entidades de los otros dominios. Son:
  - Grado de dificultad: se añade a las entidades expositivas y explicativas, indicando cuán difícil es la comprensión del elemento.
  - Nivel de adquisición: se incluye dentro de las entidades de tipo exploratorio que indica qué papel juega el problema en referencia a los conceptos que involucra. Algunos valores podrían ser comprender, aplicar, conocer, etc.

- Relaciones didácticas: establecen las dependencias teniendo en cuenta el proceso educativo. Son:
    - prerrequisito: se asocian conceptos y actividades en una sucesión que permiten establecer los órdenes de dependencia de conocimientos.
    - involucra: un problema involucra para su desarrollo uno o varios conceptos.
    - ejercita: un problema ejercita uno o varios conceptos.
    - ilustra: un ejemplo ilustra uno o más conceptos.
    - tiene faq: un elemento del dominio conceptual o instruccional está asociado con un elemento de tipo pregunta mediante esta relación.

En la figura siguiente se ve gráficamente cómo casan los diferentes dominios entre sí:

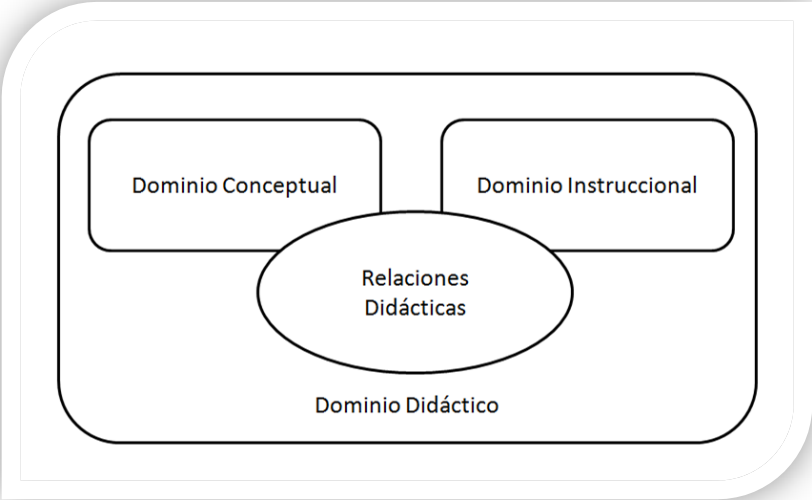

Imagen 45: Dominios que modelizan una materia de estudio

2.5.6. Lenguajes de representación de ontologías

Para la representación de las ontologías se utilizan lenguajes específicos en los que se puedan describir todos los componentes de las mismas. Puesto que no hay un estándar que defina dicho lenguaje, han ido surgiendo varios, pudiendo elegir cualquiera de ellos a la hora de construir una ontología.

Para elegir un lenguaje u otro, Honrubia da una serie de requisitos que deben cumplirse [54]:

- Debe poseer una sintaxis bien definida para ser capaz de poder leer las ontologías que se definan mediante este lenguaje.
- Debe tener semánticas bien definidas, para poder ser capaz de comprender las ontologías.
- Debe tener suficiente expresividad para poder capturar varias ontologías.
- Debe ser fácilmente mapeable desde / hacia otros lenguajes ontológicos.
- Debe ser eficiente a la hora de realizar razonamiento.

Algunos de los lenguajes más extendidos son:

- **OWL**. [64]

OWL viene de Ontology Web Language, y es uno de los lenguajes de marcado utilizados para la realización de las ontologías. Está basado en RDF y su estructura es igual al formato XML.

Existen tres variantes del mismo:

- OWL Lite: Es el sublenguaje menos expresivo, destinado a aquellos usuarios que únicamente necesiten una clasificación jerárquica de términos y restricciones simples. Comparado con RDF, añade restricciones de rango local, restricciones existenciales, restricciones de cardinalidad simple y varios tipos de propiedades (inversa, transitiva y simétrica).
- OWL DL: Comparada con OWL Lite, añade soporte total a la negación, disyunción, restricciones de cardinalidad, enumeraciones y restricciones de valor. El elemento "DL" viene por su semejanza a un lenguaje expresivo de lógica de descripciones. Asegura que todas sus conclusiones son computables y resolubles en un tiempo finito.
- OWL Full: Mientras que OWL Lite y OWL DL imponen restricciones al uso de vocabulario y el uso de declaraciones RDF, OWL Full no tiene tales restricciones. Por ello, OWL Full permite tanto la especificación de clases-como-instancias como el uso de construcciones del lenguaje. Sin embargo, este subtipo no asegura que se pueda obtener un razonamiento completo para cada característica.

Un ejemplo de cómo se vería un fragmento de una ontología en este lenguaje es el siguiente:

```
<rdf:RDF
 xml:base="http://cohse.semanticweb.org/ontologies/people">
<owl:Ontology rdf:about=""/>
 <owl:Class rdf:about="#white+van+man">
  <rdfs:label>white van man</rdfs:label>
   <rdfs:comment>
    A white van man is a man who drives a white van.
   </rdfs:comment>
   <owl:equivalentClass>
    <owl:Class>
     <owl:intersectionOf rdf:parseType="Collection">
     <owl:Class rdf:about="#man"/>
     <owl:Restriction>
      <owl:onProperty rdf:resource="#drives"/>
      <owl:someValuesFrom>
       <owl:Class>
        <owl:intersectionOf rdf:parseType="Collection">
         <owl:Class rdf:about="#white+thing"/>
         <owl:Class rdf:about="#van"/>
        </owl:intersectionOf>
       </owl:Class>
      </owl:someValuesFrom>
     </owl:Restriction>
     </owl:intersectionOf>
    </owl:Class>
   </owl:equivalentClass>
 </owl:Class>
</rdf:RDF>
```

- **OIL**. [54]

OIL es el acrónimo de Ontology Inference Layer. Unifica tres aspectos: semánticas formales y soporte de razonamiento, primitivas de modelado y un estándar para intercambiar notaciones sintácticas.

Con OIL se pueden definir las clases, slots, que son atributos sobre las clases, y axiomas. Las clases pueden estar relacionadas unas con otras, o por medio de una relación de herencia. Los slots sirven, por una parte, para dar información sobre cómo las instancias de las clases interactúan unas con otras y, por otro, para representar restricciones.

Un ejemplo sería:

- Definición de slots y propiedades:

```
Slot-def eats
        Inverse is-eaten-by
Slot-def has-part
        Inverse is-part-of
        Properties transitive
Slot-def comes-from
Slot-def age
        Range (min 0)
        Properties functional
Slot-def weight
        Range (min 0)
        Properties functional
Slot-def colour
        Range string
        Properties functional
```

- Definición de clases e instancias:

```
Class-def animal
Class-def plant
Disjoint animal plant
Class-def tree
        Subclass-of plant
Class-def branch
        Slot-constraint is-part-of
              Has-value tree
Class-def leaf
        Slot-constraint is-part-of
              Has-value branch
Class-def defined carnivore
        Subclass-of animal
        Slot-constraint eats
              Value-type animal
Class-def defined herbivore
        Subclass-of animal
        Slot-constraint eats
              Value-type (plant or
              (slot-constraint is-part-of has-value plant))
disjoint carnivore herbivore
```

- **DAML**. [54]

DAML es el acrónimo de DARPA Agent Mark-up Language. Diseñado por Jim Hendler, está cuidadosamente elaborado para que cumpla con los planes de la Web Semántica, basado en RDF.

Su sucesor es DAML+OIL, que mezcla ambos lenguajes. Un ejemplo sería [69]:

```
<rdf:RDF>
 <daml:Ontology rdf:about="">
  <daml:versionInfo>
   $Id: daml+oil-ex.daml,v 1.9 2001/05/03 16:38:38 mdean Exp $
  </daml:versionInfo>
  <rdfs:comment>
   An example ontology, with data types taken from XML Schema
  </rdfs:comment>
  <daml:imports
   rdf:resource="http://www.daml.org/2001/03/daml+oil"/>
 </daml:Ontology>
 <daml:Class rdf:ID="Animal">
  <rdfs:label>Animal</rdfs:label>
  <rdfs:comment>
   This class of animals is illustrative of a number of
   ontological idioms.
  </rdfs:comment>
 </daml:Class>
 <daml:Class rdf:ID="Male">
  <rdfs:subClassOf rdf:resource="#Animal"/>
 </daml:Class>
 <daml:Class rdf:ID="Female">
  <rdfs:subClassOf rdf:resource="#Animal"/>
  <daml:disjointWith rdf:resource="#Male"/>
 </daml:Class>
[…]
 <daml:ObjectProperty rdf:ID="hasParent">
  <rdfs:domain rdf:resource="#Animal"/>
  <rdfs:range rdf:resource="#Animal"/>
 </daml:ObjectProperty>
[…]
 <Person rdf:ID="Santa">
  <rdfs:comment>
   Santa is an instance of Person. Santa has two pieces of
    associatedData, one of which is the real number 3.14159 and
   the other of which is the string "3.14159". We may be able to
   infer a   logical inconsistency (because Persons can have at
   most 1 item of   associatedData, and a value cannot be both
   a string and a real number).
  </rdfs:comment>
  <associatedData>
   <xsd:real rdf:value="3.14159"/>
  </associatedData>
  <associatedData>
```

```
    <xsd:string rdf:value="3.14159"/>
   </associatedData>
  </Person>
</rdf:RDF>
```

2.5.7. Herramientas para la construcción de ontologías

Las herramientas relacionadas con las ontologías son de lo más variado, ofreciendo cada una de ellas diferentes posibilidades. Gómez-Pérez divide dichas herramientas en los siguientes tipos [62]:

- Herramientas de desarrollo de ontologías: herramientas tanto para la construcción de nuevas ontologías como para la reutilización de las ya existentes. Algunas de las funciones que aportan son la edición y consulta de la ontología, así como la exportación e importación de éstas, la visualización en distintos formatos gráficos, etc.
- Herramientas de la fusión y de la integración de las ontologías: pretenden facilitar la unión de dos o más ontologías sobre el mismo dominio, combinándolas e integrándolas.
- Herramientas de evaluación de ontologías: son herramientas que pretenden mantener un mínimo de calidad tanto de las ontologías como de las tecnologías relacionadas con éstas.
- Herramientas basadas en la anotación: estas herramientas permiten la adición de informaciones y datos por parte del propio usuario.
- Herramientas de almacenaje y de preguntas: son herramientas que pretenden facilitar el uso de la propia ontología.
- Herramientas de aprendizaje: se utilizan semi-automáticamente para construir ontologías a partir del lenguaje natural.

2.5.8. Ejemplos de ontologías

El área de aplicación de las ontologías es inmensa, desde la ingeniería del conocimiento para la representación del mismo y el razonamiento sobre éste, al procesamiento del lenguaje natural, pasando, por supuesto, por la recuperación de información y gestión del conocimiento, ambas involucradas en la Web Semántica.

Francisco Javier Honrubia hace una original propuesta de aplicación: un gestor de ontologías [54]. Describe dicha aplicación dividiéndola en los siguientes módulos:

- Módulo de visualización: para poder ver la ontología en alguno de los lenguajes que se definan.
- Catálogo de ontologías: almacena las ontologías en diferentes lenguajes, permitiendo conocer de qué trata la ontología, sus clases, relaciones, etc.
- Módulo de comparación de ontologías: este módulo permite comparar dos ontologías, ya estén en el mismo lenguaje o no, comprobando cuáles son los datos coincidentes y cuáles son las diferencias entre ambas.
- Módulo de creación de ontologías: mediante este módulo se podrá crear una ontología en el lenguaje que el usuario desee, facilitando la adición de nuevas clases, relaciones, etc., y almacena la ontología final en el catálogo.
- Módulo de captura de requisitos: aquí se permitirá realizar una trazabilidad entre los requisitos del sistema que se recojan a una ontología en un determinado lenguaje.
- Módulo de gestión: módulo opcional que permitirá la gestión e interacción del resto de módulos.

Cualquiera puede construirse una ontología del dominio que desee, aunque es posible encontrar algunas ya elaboradas. Algunos ejemplos son las siguientes:

- **BioPAX**.

  BioPax es un grupo cuyo objetivo es la construcción y el desarrollo de una ontología destinada al intercambio de información sobre datos biológicos. Este grupo se fundó en el año 2002 por Chris Hogue, Peter Karp, y Chris Sander. Actualmente hay varias versiones y varias bases de datos, descargables desde su página [65].

- **Dublin Core**. [66].

  Dublin Core es lenguaje para introducir metadatos basado en los modelos RDF y XML. Fue elaborado por DCMI (Dublin Core Metadata Initiative). Normalmente son usados en páginas Web, describiendo los recursos existentes en las mismas, aunque organizaciones educativas y bibliotecas también los utilizan para mantener organizados los recursos y la información de los que disponen.

  Un ejemplo de Dublin Core sería:

```
<rdf:RDF
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:dc="http://purl.org/dc/elements/1.1/">
  <rdf:Description
  rdf:about="http://media.example.com/audio/guide.ra">
   <dc:creator>Rose Bush</dc:creator>
   <dc:title>A Guide to Growing Roses</dc:title>
   <dc:description>
    Describes process for planting and nurturing different
    kinds of rose bushes.
   </dc:description>
   <dc:date>2001-01-20</dc:date>
  </rdf:Description>
</rdf:RDF>
```

- **Gellish English Dictionary**. [67]

El Gellish English Dictionary es una ontología que incluye un diccionario y una taxonomía de términos relacionados con la lengua inglesa.

- **WordNet**.

WordNet es una base de datos léxico-conceptual del inglés estructurada en forma de red semántica, es decir, compuesta de unidades léxicas y relaciones entre ellas. La característica principal de WordNet es que un concepto o término viene definido por el conjunto de formas léxicas que, en un contexto apropiado, sirven para representarlo en el lenguaje.

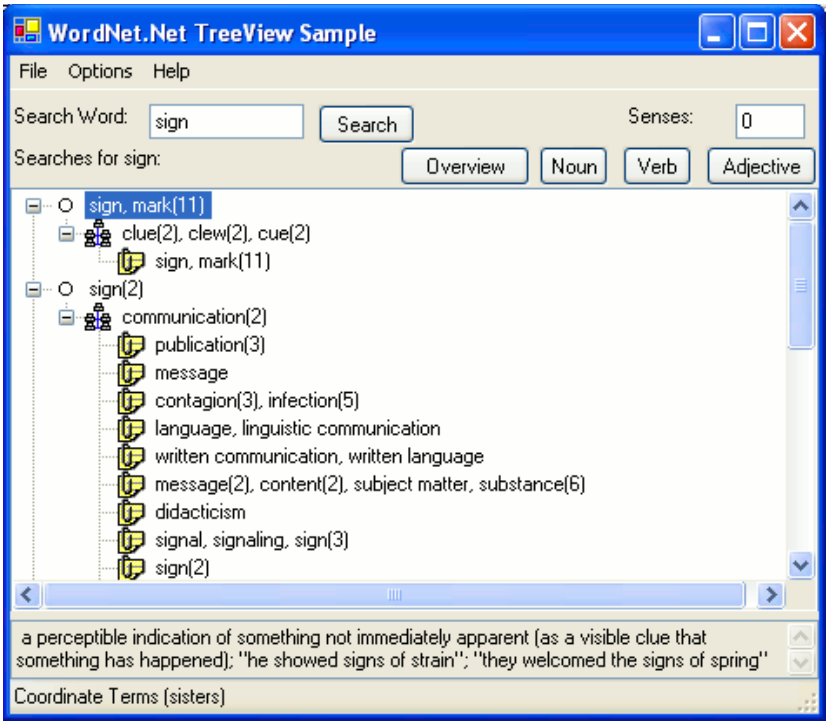

Imagen 46: Captura de WordNet

2.6. Técnicas de enseñanza

2.6.1. Modelos de enseñanza

Si bien es cierto que la información sobre las tecnologías asociadas a un proyecto informático es esencial, en este caso no hay que restarle importancia a las técnicas de aprendizaje al ser el objetivo impartir una materia lectiva, en concreto Arquitectura de Software.

Para comprender en su totalidad la enseñanza y sus implicaciones, es correcto comenzar con la teoría asociada a la misma, en concreto con los diferentes modelos de enseñanza y de procesamiento de la información [94].

Los modelos cognitivos se basan en la idea del aprendizaje a través de la manipulación de símbolos mediante determinadas reglas, de tal forma, que la mente interactúa con éstos y no con su significado. Así, la mente funcionaría correctamente cuando los símbolos representan de forma correcta la realidad externa, y el procesamiento de la información dentro del sistema lleva a una solución adecuada del problema que se ha presentado. Algunos de los modelos cognitivos son:

- Pensamiento inductivo. Hilda Taba, mayor representante de este tipo de procesamiento, analiza el pensamiento tanto desde el punto de vista psicológico,

que atiende al pensamiento, como lógico, que comprende el producto y el contenido del pensamiento.

- Organización intelectual. Ausubel defiende este tipo de procesamiento para que los profesores transmitan la mayor cantidad de información del modo más significativo y eficaz posible, relacionando cuerpos de conocimiento.

- Formación de conceptos básicos. Se basa en la categorización y está desarrollada por Bruner y sus colaboradores. Este tipo de procesamiento hace equivalentes cosas que son diferentes, definiendo nuevas categorías y formando conceptos, reduciendo así la complejidad del medio y el aprendizaje de conceptos nuevos, pues se pueden relacionar con otros ya conocidos.

- Memorización. Es la propia retención de datos. Lorayne y Lucas han construido un modelo para aumentar la atención sobre lo que hay que aprender, los sentidos implicados en dicha atención, y las asociaciones que se realizan entre el nuevo material y el ya aprendido.

- Método de descubrimiento. Consiste en ayudar a los alumnos a desarrollar la disciplina intelectual y las habilidades necesarias para plantear cuestiones y buscar respuestas resultantes de la curiosidad.

De igual forma que hay que desarrollar las capacidades cognitivas, también es necesario desarrollar el individuo, teniendo en cuenta distintos aspectos afectivos y sociales de la vida. Los modelos para tal fin son los siguientes:

- Modelo de enseñanza no directiva. En este modelo, el educador debe ponerse en el lado del alumno, respetando la capacidad de éste de plantear sus problemas y formular soluciones.

- Sinéctica, desarrollo de la creatividad. William Gordon basa la sinéctica en que la creatividad es importante en cualquier actividad diaria y de ocio, así como para las relaciones sociales, en que se puede enseñar al individuo a potenciar su creatividad, en que ésta es semejante en todos los campos, y en que el pensamiento creativo individual y grupal son muy semejantes.

- Modelo de desarrollo de la conciencia: William Schutz cree que uno de los principales obstáculos para la realización y la alegría en las relaciones personales es la incapacidad de la gente de ser consciente de sus propias necesidades y sentimientos, siendo necesario que la gente se libere de sus emociones. Así, ha desarrollado una serie de actividades de aprendizaje que facilitan respuestas emotivas de los sujetos.

- Modelo del grupo de aula. Se basa en la idea de William Glasser de que los humanos necesitan autoestimación para desarrollarse satisfactoriamente en grupo.

Los modelos de interacción social son:

- Modelo de juego de roles. Trata los problemas mediante la acción. Se plantea el problema, se representa y se discute, haciendo que unos alumnos jueguen los papeles y otros observen.

- Modelo del trabajo en grupo como proceso democrático. John Dewey y Herbert Thelen dicen que el hombre es un ser social. Mediante la negociación, los alumnos aprenden a dominar los conocimientos comprometiéndose en la solución de problemas sociales.

Uno de los conjuntos de modelos más importantes son los modelos conductistas, los cuales insisten en cambiar el comportamiento visible del sujeto y no su estructura psicológica. Con ese fin, se dan los medios para llegar al comportamiento esperado y verificar su obtención, aunque nada garantiza que el comportamiento externo se corresponda con el mental. Algunos de estos modelos son:

- Método del refuerzo. Establece relación entre estímulo y respuesta. Se aporta al alumno una gratificación cuando el trabajo o la respuesta es realizada correctamente.

- Método de autocontrol mediante el conocimiento operante. La parte del refuerzo positivo queda en manos del interesado, teniendo éste procedimientos de autogratificación. Una razón para seguir este método es que en ocasiones, el medio no proporciona incentivos al ritmo necesario para que las condiciones actuales del sujeto produzcan un comportamiento nuevo.

- Modelo de reducción de estrés. Si un sujeto acumula excesivos puntos de estrés en un período determinado, se produce un trastorno o enfermedad. Para evitar estas situaciones, se ha demostrado que es útil que el sujeto hable consigo mismo dándose consignas para relajarse y se refuerce por su comportamiento.

Otro modelo es el constructivismo, el cual afirma que el conocimiento de todas las cosas es un proceso mental del individuo, que se desarrolla de manera interna conforme el individuo interactúa con su entorno. El constructivismo social sostiene que el ser humano aprende gracias a encontrarse en un entorno social, lo que le permite la comprensión del conocimiento y la cognición. Siguiendo esta línea de razonamiento, el ambiente de aprendizaje más óptimo es aquel donde existe una interacción dinámica entre los instructores, los alumnos y las actividades que proveen oportunidades para los alumnos de crear su propia verdad, gracias a la interacción con los otros.

Así, un profesor constructivista es simplemente un mediador entre el conocimiento y el aprendizaje de los estudiantes, que comparte sus experiencias en una actividad conjunta de construcción de los conocimientos, preocupándose por formar alumnos autodidactas.

Paul Ernest (1991) resume los principios del constructivismo social de la siguiente manera:

- El conocimiento no se recibe pasivamente sino que es construido activamente por el sujeto cognitivo.
- Las teorías personales que resultan de la organización experimental del mundo, deben calzar las restricciones impuestas por la realidad física y social.
- Esto se logra a través de un ciclo de Teoría - Predicción - Prueba - Error - Rectificación - Teoría.
- Esto da paso a las teorías socialmente aceptadas del mundo y los patrones sociales así como las reglas de uso del lenguaje.
- El constructivismo social es la reflexión que hacen aquellos que están en la posición de enseñar a los demás, como ellos enseñan, y la información que muestran a los otros.

Seitzinger (2006) se apoya en el constructivismo social y aporta algunas características que debe tener dicho tipo de aprendizaje.

- Activo y manipulable: son los estudiantes quienes interactúan y exploran, dándoles oportunidad de concienciar el resultado de su manipulación del aprendizaje.
- Constructivo y reflexivo: el estudiante asimila los nuevos conocimientos y los relaciona con los previos, lo cual lleva a la reflexión de su aprendizaje.
- Intencional: el estudiante es el que se propone alcanzar unas determinadas metas, controlándose a sí mismo hasta que lo logra.
- Auténtico, retador y contextualizado: el estudiante sitúa su aprendizaje en situaciones reales, preparándole para futuros retos.
- Cooperativo, colaborativo y conversacional: fomenta la interacción entre estudiantes para discutir problemas, aclarar dudas y compartir ideas.

En la siguiente imagen se puede observar gráficamente algunos de los modelos de enseñanza comentados y su aplicación.

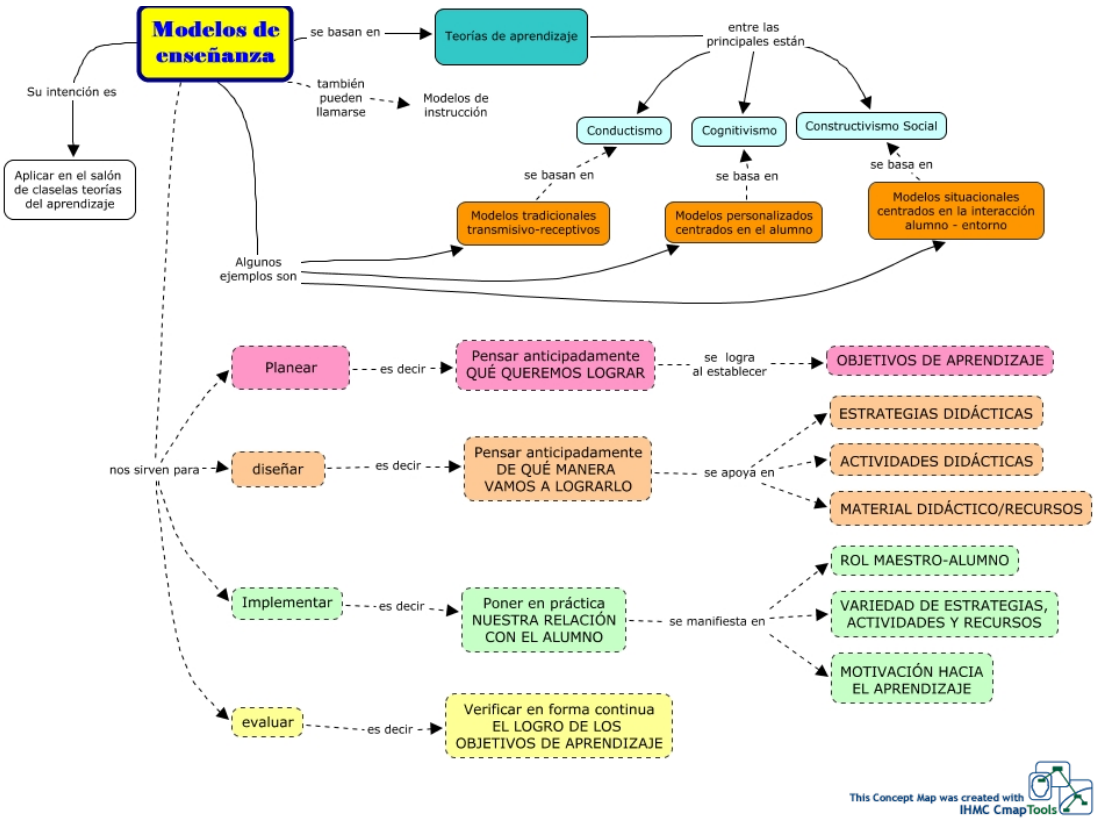

Imagen 47: Modelos de enseñanza

Tan importante como es el proceso de adquisición del conocimiento es el hecho de evaluar cómo de bien se ha realizado esta primera tarea. Hay dos tipos de paradigmas para la evaluación de los conocimientos:

- Paradigma positivista.

Augusto Comte plantea que los conocimientos pasan por tres estados teóricos distintos [99]:

- Teleológico: donde la mente busca las causas y principios de las cosas, lo más profundo y lejano e inasequible.
- Metafísico: es una etapa intermedia entre el estado teológico y el positivo. (…) la mente que se lanzaba a lo lejano, se va acercando paso a paso a las cosas, y así como en el estado anterior los poderes se resumían en el concepto de Dios, aquí es la Naturaleza la gran entidad que lo constituye.
- Positivo o real, que es el definitivo. En él la imaginación queda subordinada a la observación. La mente humana se atiene a las cosas. El positivismo busca sólo hechos y sus leyes, no causas ni principios de las esencias o sustancias. Todo esto es inaccesible. El positivismo se atiene a lo positivo, a lo que está puesto o dado, es la filosofía del dato.

En definitiva, lo que interesa es la información que el alumno ha adquirido, y evaluarla.

- Paradigma Alternativo o Comprensivista.

Consiste en comprender procesos más que en medir resultados. Se entiende la evaluación como una ayuda para el aprendizaje, donde se toma en cuenta los significados que los alumnos otorgan al proceso de aprendizaje, sus expectativas, intereses, necesidades, actitudes, estilos de aprendizaje y su manera de interactuar con sus compañeros, el docente y su entorno [100].

2.6.2. Técnicas y estrategias del aprendizaje

Existen numerosos trabajos que hablan de las mejores técnicas o prácticas para que el aprendizaje de los alumnos sea óptimo, y no sea una simple memorización de la teoría que sea olvidada en un breve periodo de tiempo.

María Cristina Cicarelli ha escrito un documento en el que habla de estas técnicas y estrategias [90]. En él pone la sensibilización y atención como principales medios de conseguir que el alumno adquiera correctamente los conocimientos de la materia. Según ella, de este contexto forman parte la motivación, la emoción y las actitudes, poniendo especial énfasis en la primera de éstas, puesto que "*mediante la motivación el alumno se abre activamente hacia los datos del input informativo para interpretarlos, procesarlos e integrarlos en las redes informativas ya existentes*".

En cuanto a la motivación, propone los siguientes modelos:

- Motivación intrínseca.

Es aquella por la cual se realiza una actividad por el simple placer de realizarla, sin obtener un incentivo externo. En este tipo de motivación existe un alto grado de estimulación que proviene del propio alumno, una curiosidad sobre la materia, lo que le influye positivamente para el aprendizaje.

Las estrategias para conseguir esta motivación son el desafío, la curiosidad, el control y la fantasía.

- Motivación de logro.

Utiliza el deseo de triunfar o alcanzar un nivel de excelencia. Aquellas personas con este tipo de motivación consiguen más logros que aquellos que carecen de la misma, incluso cuando fracasan, pues lo atribuyen a una falta de esfuerzo por su parte y no se desmotivan.

Entre las técnicas para incrementar esta motivación se encuentran aumentar la necesidad de rendimiento, disminuir el temor al fracaso y aumentar la probabilidad esperada de éxito.

- Atribución casual y expectativas de éxito.

Se refiere a las causas que la gente suele dar para el éxito o fracaso sufrido, como la capacidad, la suerte, el esfuerzo o la dificultad de la tarea. Estas atribuciones afectan a las expectativas de éxito, a las reacciones emocionales y a la persistencia en tareas relacionadas con el rendimiento.

Para procurar que el alumno se desmotive por la perspectiva de falta de éxito, es necesario comenzar con actividades que requieren poco esfuerzo para conseguir realizarlas correctamente, e ir incrementando la dificultad según éste se sienta más seguro y confiado. Es necesario, sobre todo en niños, hacerles saber que están consiguiendo alcanzar sus objetivos gracias a su capacidad y a su esfuerzo.

- Orientación a la meta.

Hay dos tipos, orientada a la tarea y orientada al yo. En la primera, el aprendizaje es valorado por sí mismo, la atención está centrada en la tarea y en las estrategias necesarias para dominarla. En el segundo caso, se utiliza el aprendizaje para sentirse más inteligente o para no parecer torpe.

- Estrategias de refuerzo.

Se utilizan los refuerzos para evitar los efectos negativos o desfavorables, sobre todo cuando los resultados no son satisfactorios. El alumno debe saber autoevaluarse y autoreforzarse.

Otro documento interesante sobre las estrategias y técnicas de aprendizaje es el escrito por Francisco de Asís Martín [91]. En él hace un extenso análisis de las fases de aprendizaje y de las técnicas en cada una de ellas, atendiendo principalmente a los expertos Pedro Hernández y Jesús Beltrán.

El autor pone énfasis en la necesidad de ejercitar los conocimientos para que éstos queden en la memoria del alumno por sí mismas y no porque lo memorice sin entender el significado real de las palabras que está estudiando. Diferencia tres fases, las cuales se subdividen en procesos, y éstos en estrategias que se entrenan mediante diferentes técnicas:

- Fase de recepción.

Consiste en interesar al alumno por lo que tiene que hacer. Aparte de la labor del docente para motivar al estudiante, explicando de forma clara la lección, este último también tiene que actuar para adquirir el conocimiento transmitido, incluso en esta fase: deben estar preparados y receptivos para captar la información que les van a transmitir. Se pueden distinguir dos procesos cognitivos diferentes:

- Procesos de sensibilización.

  - Estrategias motivaciones: hay dos tipos de motivaciones, intrínsecas y extrínsecas. La primera es la explicada anteriormente, por la tanto, es deducible que con la motivación extrínseca el alumno obtiene algún tipo de beneficio externo para prestar atención a lo que se le está explicando. En la imagen siguiente se pueden ver las técnicas propuestas para cada una las motivaciones:

Estrategias motivacionales	
INTRÍNSECAS	**EXTRÍNSECAS**
• Desafío • Adelanto del éxito • Actitud de éxito y logro • Atribución de causalidad • Orientación a la tarea • Relaciones personales • Búsqueda de apoyo • Biblioterapia	• Refuerzo • Reforzamiento positivo • Autocontrol de Kanfer • Autoinstrucciones • Presencia del sancionador • Economía de Fichas • Moldeado y modelado • Encadenamiento

Imagen 48: Técnicas de estrategias motivacionales

  - Estrategias actitudinales: se centran en la beneficiosa actitud positiva ante un problema, siempre y cuando también sea realista, que permita afrontar las dificultades del aprendizaje. Hay dos técnicas en este caso. En primer lugar, la orientación al problema, que se refiere a la posición relativa que adopta el alumno al enfrentarse al estudio, fomentando la confianza en sí mismo y en la posibilidad de poder y querer solucionar los problemas que se le planteen, imponiéndoles para ello la necesidad de consultar otras fuentes, como otros compañeros, profesores, libros de consulta, etc. Por otro lado, se tiene la actitud motivacional, que consiste en crear un calendario de actividades y tener la clara intención de seguirlo. En ambas se pretende que el alumno sea capaz de enfrentarse a los problemas que se le planteen, buscando ayuda si es necesaria, con una actitud positiva y sin postergar la resolución del mismo, alejando los pensamientos negativos que puedan desmotivarle.

- Procesos atencionales. Es necesario que además de mantenerse motivado y sensibilizado, el alumno sea capaz de mantener el interés sobre aquello que se le está impartiendo.

  - Estrategias de atención. Hay distintos tipos de atención según el momento: atención global o comprensiva, donde la atención del

alumno puede variar según él lo decida; atención selectiva, que hace referencia a la capacidad del sujeto para centrarse en una parte del mensaje ignorando el resto; y atención sostenida, donde la atención se mantiene a lo largo de un periodo completo. En la siguiente imagen se pueden comprobar una serie de técnicas para cada una de las estrategias nombradas.

ESTRATEGIAS ATENCIONALES		
ATENCIÓN GLOBAL	**ATENCIÓN SELECTIVA**	**ATENCIÓN SOSTENIDA**
1. Exploración de la estructura de los datos	1. Evaluación 2. Entrenamiento 3. Fragmentación y combinación	1. Evaluación 2. Entrenamiento

Imagen 49: Técnicas de estrategias atencionales

- Fase de transformación.

En esta fase el alumno interioriza la lección aprendida para adaptarla a su manera de pensar, de forma que se le quede en la memoria para utilizar dicha información cuando sea necesario. Implica dos procesos:

- Comprensión. Es el proceso de seleccionar y organizar la información que ha debido ser, previamente, entendida. Además de entender una lección, el alumno debe comprenderla, lo cual tiene que ver con el hecho de asimilarla. Tiene dos estrategias:
  - Estrategia de selección. Según Beltrán, el proceso de adquisición del conocimiento comienza con la selección o codificación selectiva mediante la cual se logra la incorporación del material informativo de interés para el alumno. Una vez se ha producido la explicación, que puede incluir declaraciones, ejemplos, etc., el receptor, es decir, el estudiante, debe hacer la comprensión, seleccionado los términos esenciales y significativos del mensaje emitido por el profesor. Entre las técnicas de selección están la exploración, notas marginales, subrayado, esquemas, toma de apuntes, selección de libros, etc.
  - Estrategias de organización. Es el segundo paso en la comprensión. El alumno debe organizar lo aprendido de alguna forma, donde aparezca una jerarquización conceptual. Para ello, por una parte se deben relacionar los nuevos materiales entre sí; y por otra, hay que conexionarlos con los que ya tiene el alumno. Entre las técnicas de organización están resumen, red semántica, árbol organizado, mapa semántico, mapa heurístico, etc.

- Retención y almacenamiento. Una vez la información ha sido comprendida, es necesario un proceso de retención, para asegurarse de que las lecciones quedan en la mente del alumno.

    - Estrategias de elaboración. Según Hernández, la elaboración es el proceso a través del cual la mente, de modo personal y subjetivo, se aplica de forma activa y constructiva sobre una información determinada, logrando nuevas informaciones o productos distintos de los expuestos explícitamente en esa información. En la siguiente imagen se pueden observar las técnicas propuestas por los dos expertos mencionados en el artículo:

PROCESO DE RETENCIÓN	
ELABORACIÓN: P. HERNÁNDEZ	**ELABORACIÓN: J. BELTRÁN**
1. Toma de notas - en clase - conferencia - a partir de un texto **2.** Visión previa **3.** Elaboración de memorización **4.** Elaboración de comprensión y consolidación - elaboración de inferencias - desarrollar inferencias y hacerse preguntas - calentamiento de imagen **5.** Valoración **6.** Ampliación	**1.** Interrogación elaborativa **2.** Metáforas y analogías **3.** Procedimientos nemotécnicos **4.** Toma de apuntes o notas **5.** Organizadores previos **6.** La imagen **7.** La activación del esquema

Imagen 50: Técnicas de la estrategia de elaboración

    - Estrategia de repetición. Estas estrategias pretenden mejorar el proceso mediante el cual las personas retienen una información en la memoria a largo plazo. Consiste en pronunciar, nombrar o decir de forma repetida aquella información que se desea memorizar. Hay dos formas de repetición: de mantenimiento, que se refiere al reciclado de la información para mantenerlo activo en la memoria a corto plazo; y la elaboración, en la que la nueva información se relaciona con otra más antigua con el fin de recordarla mejor. A continuación se ven las técnicas para el proceso de retención mediante la estrategia de repetición.

PROCESO DE RETENCIÓN	
REPETICION SEGÚN PEDRO HERNÁNDEZ	**REPETICIÓN SEGÚN JESÚS BELTRÁN**
1. Nemotecnia de repetición. **2.** Nemotecnias con conexión motivacional **3.** Nemotecnias de asociación: superficial y profunda **4.** Integración sintáctica **5.** Interrelación semántica **6.** Asociación analógica **7.** Conexión con imágenes visuales **8.** Encadenamiento de imágenes. **9.** Método loci **10.** Palabras pegadas. **11.** Procedimental-vivencial	**1.** Reenunciado verbal **2.** Reenunciado substancial **3.** Repetición Verbal **4.** Repetición substancial **5.** Reenunciado más detallado. **6.** Referencia implícita.

Imagen 51: Técnicas de la estrategia de repetición

- Fase de recuperación y transferencia de la información.

Esta última fase hace referencia al hecho de utilizar lo aprendido, comprendido y memorizado en las fases anteriores, volcando los conocimientos adquiridos y aplicándolos para resolver problemas reales.

- Procesos de evocación. Participan en la recuperación de la información y consisten en acceder a la información almacenada en la memoria a largo plazo permitiendo su activación y posible utilización consciente. Entre las técnicas están la búsqueda autónoma, de huella, de elección y de reconocimiento.

- Procesos de generalización y transferencia. Este proceso es a través del cual la información recuperada es aplicada a otras situaciones distintas de la situación original en la que se aprendió. Hay dos tipos: hacer la transferencia entre situaciones muy parecidas en las que no se requiere ninguna búsqueda ni abstracción por parte del alumno; y presentar situaciones completamente distintas, de áreas diferentes, para que el estudiante tenga que emplear una generalización abstracta a la hora de resolver el problema. Algunas técnicas son:

  - Incremento de elementos idénticos: facilita la generalización del aprendizaje de un tipo de contenido a otro que comparten elementos idénticos.

  - Identificación de los principios generales: ofrece al estudiante la posibilidad de identificar los principios generales que rigen la realización satisfactoria de las tareas originales.

  - Variabilidad estimular: incrementa el mantenimiento y transferencia de lo aprendido, haciendo que se dé la conducta aprendida en una gran variedad de situaciones distintas.

- Incremento de la variabilidad de la respuesta: eleva al máximo la disponibilidad de la respuesta para que sea aplicada en múltiples ocasiones.

- Procesos cognitivos de comunicación. Se refieren a la fase en la que el alumno demuestra lo que ha aprendido. Se demostrará un verdadero aprendizaje cuando el alumno aplique los conocimientos tras un proceso de reflexión y deducción, resolviendo los problemas propuestos aunque no se dé el mismo caso que en el problema original, y no cuando repita la lección de memoria sin entender, en muchas ocasiones, lo que significa.

Es un hecho que el conocimiento cada vez se transmite más a través de medios electrónicos, por lo que la captación del mismo a través de imágenes o mapas conceptuales se hace más importante, siendo ésta, además, una manera de acercarse a la forma en la que la mente humana recupera la propia información del individuo [97].

Este tipo de método es especialmente útil para:

- Aprendizaje visual.

Este tipo de aprendizaje es especialmente útil para enseñar las habilidades del pensamiento. Los métodos gráficos ayudan al estudiante a pensar con claridad, a elaborar, organizar y priorizar la nueva información, además de estimular el pensamiento creativo y el pensamiento crítico, puesto que transmiten tanto la información básica como las relaciones, estructuras, modelos y características que de otra forma no son evidentes. Entre los instrumentos de este tipo de aprendizaje están los mapas conceptuales y las simulaciones en ordenador.

Las técnicas de aprendizaje visual que ayudan a los estudiantes son:

- Depurar el pensamiento: la representación hace que se vea explícitamente cómo se relacionan los conceptos.

- Reforzar la comprensión: interactuar con los mapas conceptuales representando lo que se ha aprendido hace que dicho conocimiento se interiorice más profundamente y se mantenga más tiempo en la memoria.

- Integrar nuevo conocimiento: se relaciona el conocimiento nuevo con el ya existente, favoreciendo su comprensión y permitiendo retenerlo mejor en la memoria.

- Identificar errores conceptuales e incomprensiones: un mapa elaborado por el estudiante permite al profesor identificar aquellos conceptos o áreas que no han quedado claros o que son erróneos.

- Aprendizaje Activo.

Los mapas conceptuales activos se refieren a aquellos mapas conceptuales con los que se puede interactuar, de manera que se potencia la interacción cognitiva. Hay diversos modos de desarrollar este aprendizaje activo:

- Crear y/o analizar los mapas y bases de conocimiento, individualmente o en grupos.
- Adaptar y desarrollar mapas hechos por otros.
- Asociar documentos, incluso multimedia, a los conceptos.
- Interactuar con los conceptos, sus instancias y sus relaciones.
- Categorizar conceptos y relaciones.
- Buscar en el mapa del conocimiento (o en todos los mapas a la vez) para localizar conceptos y descripciones interesantes.
- Interrogar las bases de conocimiento con varios tipos de preguntas, obteniendo respuestas, también a voz (un diálogo cognitivo).
- Ejecutar los recorridos semánticos, evidenciando las pistas cognitivas del mapa y trayectos de pensamiento.
- Crear los propios recorridos semánticos, comentándolos.
- Analizar las descripciones, los textos y los documentos asociados.

- Aprendizaje colaborativo.

El aprendizaje colaborativo es un punto importante dentro de la enseñanza, pues existen estudios que han afirmado que la satisfacción de los estudiantes es superior en los ambientes colaborativos. En dichos entornos, los estudiantes resuelven problemas, responden preguntas, formulan preguntas propias, discuten, explican, debaten o hacen brainstorming. Así, pueden alcanzar un nivel más profundo y permanente de comprensión.

Por último, comentar las ideas de Carles Dorado [98], quien propone que los principales procedimientos estratégicos son:

- Análisis y discusión de casos.

Estos procesos tienen como objetivo recoger las ideas que se han hecho manifiestas tras la exposición de un caso y analizarlas y valorarlas bajo un proceso explícito concreto.

Los procedimientos de análisis y discusión de casos se pueden enfocar hacia:

- Análisis de los propios procesos: se basa en la comparación, discusión y evaluación, valorando aciertos y errores, de los procesos cognitivos seguidos para tomar decisiones.
- Análisis de las actividades de otras personas: promueve el intercambio de sugerencias, la comparación, la crítica, la interrogación desde múltiples visiones, etc.
- Casos de pensamiento simulados: se basan en la presentación de un problema donde diferentes personajes juegan su role, explicando sus procesos de pensamiento a fin de optar por diferentes soluciones.

➢ Imitación de modelos.

Hay distintos tipos de modelos:

- Modelo experto: una persona experta en una determinada disciplina o actividad es capaz de optimizar su funcionamiento en relación al trabajo y dar una acción eficaz y consistente. Estas acciones son pautas en sí mismas, pero si el experto hace público sus procesos de decisión y valoración, esta situación puede aprovecharse como forma de aprendizaje.
- Otros modelos: los profesores y alumnos a menudo actúan como modelos de pensamiento y acción, cuando explican el Qué, el Cómo y el Por Qué de lo que hacen.

➢ Procedimientos de interrogación.

Tiene como finalidad la optimización del proceso cognitivo, centrándose en la reflexión y la consciencia. La estrategia puede tener dos visiones:

- Guías de interrogación: El objetivo es interiorizar el conocimiento que acabe en autonomía y posibilidad de transferir dicho conocimiento. Con esta intención se entra en una dinámica de formulación constante y programada de interrogantes.
- Obtención de información: el objetivo estratégico viene definido por la obtención de un conjunto de datos o procedimientos que aporten nuevos elementos de aprendizaje significativo.

2.6.3. Aprendizaje en mundos virtuales

Todo lo aquí mencionado no está especialmente diseñado para el aprendizaje en mundos virtuales, que es el caso propuesto en este proyecto, aunque sí es cierto que es perfectamente aplicable. El proceso de aprendizaje sigue las mismas fases ya sea en un entorno real como en uno virtual: la motivación, el proceso de comprensión, y la utilización de esos conocimientos en casos prácticos han de realizarse indistintamente.

Por otra parte, también es cierto que cuando se trata de mundos no reales, hay algunas cosas más a tener en cuenta. Según el documento sobre las "Comunidades Virtuales de Aprendizaje como herramienta didáctica para el apoyo de la labor docente" [92], una comunidad virtual deberá contemplar:

- Un espacio destinado a la publicación de proyectos, producciones y actividades que girarán en torno al ámbito de la comunidad.
- Tener hipervínculos o direcciones relacionadas con la temática de la comunidad que sirvan para orientar a docentes y alumnos en la búsqueda de materiales adicionales.
- Contemplar direcciones que contengan recursos de información relevante para el área que se está trabajando.
- Un calendario de actividades en donde se especifiquen las fechas en que se desarrollarán foros de discusión, entregas de trabajos, etc.

Igualmente, resalta la importancia de la colaboración y comunicación entre profesor y alumno, siendo el primero más que un simple comunicador de una lección, además debe ser partícipe en el proceso de aprendizaje del alumno, animándole a participar en actividades dinámicas y cooperativas, tanto con él mismo como con otros compañeros, y a revisar información adicional que posibilite la interiorización de la materia estudiada con mayor facilidad. Ha quedado demostrado que, incluso en personas adultas, no funciona el aprendizaje simplemente aportando textos científicos o libros especializados, si no que es necesaria una figura superior que guíe y ayude en el proceso.

En el artículo "*Las estrategias de enseñanza en entornos virtuales*" [93], se hace referencia a las herramientas que deben usar los profesores en dichos mundos basándose en la investigación a través de varios entornos de enseñanza encontrados en la Web. Estas herramientas son:

- El correo electrónico para mantener contacto entre profesor y alumno, de manera asíncrona, con un fin de consulta y tutoría.

- Recursos externos que se pueden encontrar en la Web, aportando fuentes adicionales a las que se pueden encontrar en el programa formativo del entorno virtual.

- Videoconferencias, que aportan un feedback instantáneo, manteniendo un cierto contacto visual entre los participantes.

- Chats de conversación en tiempo real que funciona igual que las conferencias, aunque sin la imagen.

- Glosario de términos para mantener una referencia global acerca de la materia, ayudando de esta forma a la comprensión de la misma por parte del alumno.

- Índice de materiales del curso, organizando así los contenidos.

- Calendario de eventos, que ayuda a los alumnos a mantener un orden en el estudio de la materia, teniendo constancia de los hitos importantes que puedan producirse.

- Preguntas de autocomprobación que se corrigen instantáneamente, con las que el alumno podrá analizar si está realizando correctamente la interiorización de los conocimientos ofrecidos.

- Test y exámenes, que servirán al profesor para evaluar si el alumno tiene los conocimientos necesarios para pasar el curso.

Igualmente, en este artículo se presentan las que podrían ser las nuevas estrategias de enseñanza:

- Mayor exposición, prestando especial atención a que sea lo más clara y concisa posible, dando menos importancia al aprendizaje experimental.

- Mayores fuentes de información, aportando otras que potencien el aprendizaje creativo y la capacidad de síntesis.

- Uso de hipertextos, actividades de búsqueda, dinámicas de grupos online, etc. Con todas estas actividades los alumnos pueden descubrir nuevos conceptos y construirlos significativamente.

En el artículo "*Adecuación de un modelo de enseñanza superior en modelo de enseñanza virtual*" [96] se habla de crear un sistema que apoye la enseñanza proporcionada por las Universidades, dividiendo dicho espacio en las distintas carreras, cursos, etc. ofertados. En dicho artículo, se pone de manifiesto la importancia de que alumno y profesor desempeñen correctamente sus nuevos roles, siendo éstos diferentes que en la enseñanza tradicional. El primero debe tener en cuenta que la responsabilidad del aprendizaje en estos casos es mayor, pues dispone de la plataforma todo el día, y es él mismo quien determina sus propios horarios de acuerdo a sus necesidades. El profesor, en cambio, necesita planificar el trabajo con mucha

antelación y una cantidad y diversidad mayor de materiales e instrumentos educativos que debe preparar o seleccionar con mucho cuidado, aprovechándose de que las nuevas tecnologías le proporcionan mecanismos asíncronos, síncronos y de autoformación.

Según dice en el propio artículo: "*Un modelo definido así (con la estructura organizacional de las instituciones universitarias) no se puede encuadrar en aquellos preestablecidos como el constructivista (Fairstein, 2001), Montessori (Yaglis, 1989), activo (Becerra, Gónez & Salinas, 2002), ecléctico, tradicional; porque al existir una gran libertad para el docente en la educación superior universitaria es probable que la naturaleza de cada asignatura obligue a combinar estos modelos o diseñar modelos ad-hoc, teniendo como objetivo fundamental lograr el mayor grado de aprendizaje por parte de los estudiantes.*"

3. HERRAMIENTAS

3.1. Mashup

3.1.1. Yahoo! Pipes

La empresa Yahoo! sacó al mercado en febrero de 2007 una aplicación web para la creación de Mashups. Estos mashups se llaman Pipes, y recopilan feeds de otras páginas, agregándolos y modificándolos. Además, es posible compartir los Pipes creados entre usuarios registrados [70].

Según indica la información facilitada en la página de la firma, los servicios más populares que ofrece son:

- Crear tu feed personalizado combinando otros feeds en uno sólo, ordenándolos, filtrándolos y traduciéndolos.
- Codificar geográficamente tu feed favorito y mostrar los elementos en un mapa interactivo.
- Mezclar tus feeds favoritos y usar el "Pipe" para potenciar una nueva aplicación.
- Construir páginas de búsqueda personalizadas imposibles de hacer con motores de búsqueda corrientes.
- Incluir widgets y badges en tu web site.
- Leer la salida de cualquier Pipe en RSS, JSON y otros formatos.

Como característica interesante cabe destacar la capacidad de filtrado y traducción de los feeds recopilados, pudiendo así mostrarlos en la nueva página web como el usuario desee, de una manera más personalizada. Otro dato a tener en cuenta, es que los Pipes creados pueden ser reutilizados para crear nuevos Pipes, ya sean los originales del propio usuario o de cualquiera de las personas registradas en la página.

3.1.2. IBM Mashup Center

IBM ha diseñado una herramienta para la creación de soluciones en el mundo empresarial a través de mashups, con las capacidades de seguridad y administración que requiere un departamento de TI. Combina las herramientas, también de IBM, IBM Lotus Mashups e IBM InfoSphere MashupHub [71].

Esta herramienta proporciona un entorno de mashup ligero, donde "las organizaciones pueden desbloquear y transformar Web e información personal y departamental en consumibles o activos agregables a mashups, incluyendo la información de feeds y widgets. Estos activos pueden ser recopilados dinámicamente en nuevas aplicaciones que conducen a los retos empresariales diarios". En definitiva, IBM ofrece mejorar la productividad de las empresas gracias a los mashups a través de esta herramienta con la que podrán crear sus propias aplicaciones personalizadas.

Las capacidades que ofrece son:

- Acceder a diversas fuentes de información, incluyendo bases de datos, información de escritorio y de los departamentos, y sitios Web, tanto externos como internos. Se pueden crear feeds a través de una sencilla interfaz que usa el método de point-and-click (apuntar y clicar). En cuanto a aspectos técnicos, permite añadir documentación técnica y especificar permisos de seguridad. Además, provee de los conectores más comunes para los sistemas más extendidos en las empresas, como SAP o Microsoft Excel, pudiendo los usuarios crear sus propios conectores gracias a un plug-in.

- Crear widgets dinámicos sin necesidad de escribir código gracias al programa incluido en IBM Mashup Center, Lotus Widget Factory. Además, se puede hacer uso de IBM WebSphere sMash, una aplicación donde los desarrolladores pueden extender las habilidades de los widgets introduciendo características de la Web 2.0.

- Transformar la información recibida de los feeds, creando otros nuevos. Para ello, se utilizará una herramienta a través del navegador gracias a la cual se podrá analizar, mezclar, fusionar, agrupar, ordenar, anotar, filtrar y transformar los feeds de gran variedad de formas, creando una única vista de información totalmente diferente. Soporta funciones avanzadas como expresiones regulares y codificación. Los feeds pueden ser publicados en un gran número de formatos además de los habituales ATOM, RSS y XML.

- Descubrir y compartir recursos, publicando lo mashups, widgets y feeds creados en un catálogo centralizado accesible por todos los usuarios. De esta manera, también se puede aumentar la publicidad gracias a la reutilización de los recursos más útiles.

- Rápido montaje de mashups a través de la herramienta. Los usuarios pueden, rápida y fácilmente, crear mashups arrastrando y soltando widgets en la página. La aplicación incluye widgets ya prefabricados que soportan un amplio abanico de opciones de visualización, como gráficos y tablas. Adicionalmente, en IBM Mashup Catalog, los usuarios pueden encontrar más widgets, entre ellos miles de Google Gadgets.

3.1.3. JackBe Presto

La empresa JackBe® ha desarrollado JackBe Presto, un paquete de herramientas para la creación de mashups que permite capacitar a las organizaciones para crear, personificar y colaborar a través de mashups empresariales permitiendo tomar decisiones más rápidamente y con mejores resultados [72].

Existen tres productos dentro de la familia Presto:

- Presto Enterprise Mashup Server.

  Proporciona un servicio virtualizado que resuelve el problema principal en los negocios: acceder a servicios de información, sin importar su localización, y conectarlos conjuntamente para proporcionar respuestas.

- Presto Mashup Composers.

  Dentro de Presto Mashup Composers, se pueden encontrar:

  - Presto Mashlets. Esta herramienta proporciona mashups dinámicos a la empresa a través de widgets. Una vez el mashlet ha sido creado, puede usarse en cualquier sitio: en un portal de una compañía, en un blog, en una microaplicación para compartir otros mashlets, etc.

  - Presto Wires. Esta herramienta está destinada para TI y usuarios empresariales y está diseñada para un desarrollo rápido del mashup. Los usuarios recogen y combinan datos de diversas fuentes para la toma de decisiones en una pequeña fracción de tiempo, consiguiendo una imagen completa de la situación financiera, operacional o del mercado.

  - Presto Mashup Studio. Es un plug-in de Eclipse que provee a los programadores de Java completo control para diseñar, probar y lanzar mashups.

- Presto Mashup Connectors.

  JackBe ha creado una serie de conectores entre portales, ESBs, infraestructura empresarial como SOA, Oracle, SharePoint y Microsoft Excel. Otros conectores incluyen APIs para vincular el mashup con código en Java, JavaScript y C#.

3.1.4. Comparativa

	Yahoo! Pipes	IBM Mashup Center	JackBe Presto
Conocimientos informáticos	Usuario normal	Usuario normal	Usuario normal
Interfaz	Sencilla	Sencilla	Sencilla
Precio	Gratis. Necesario registrarse	2,913.92 € (Licencia para 20 usuarios + Subscripción + Servicio al cliente 12 meses) 493.00 € (Licencia para 1 usuario + Subscripción + Servicio al cliente 12 meses)	Gratis. Necesario registrarse para obtener la licencia
Uso de feeds	Sí	Sí	Sí
Uso de APIs	No	Sí	Sí
Uso de Screen Scraping	No	No	Sí, mediante código programado
Acceso a herramientas externas	No	Sí, sobre todo herramientas empresariales, bases de datos, etc.	Sí

Tabla 2: Comparativa entre herramientas de mashups

3.2. Entornos virtuales

3.2.1. Multiverse

La plataforma Multiverse ofrece la tecnología necesaria para que equipos de desarrolladores puedan crear juegos online multijugador; mundos virtuales para el entretenimiento, negocios o educación; y juegos que puedan agregarse a redes sociales o portales de juegos [81].

Lo que ofrece Multiverse es un conjunto de herramientas para llevar esto a cabo, como Flash, 3ds Max, Maya, Java, Python y muchas más, dando la posibilidad de crear mundos virtuales por un precio razonable. Las posibilidades de pago también son variadas, permitiendo obtener licencias flexibles según el uso o incluso no pagar nada por adelantado. En cuanto a la propiedad intelectual, el equipo de desarrollo tiene todos los derechos sobre el mundo a construir.

Se puede usar la plataforma gratis siempre que sea para un uso no comercial, teniendo la opción alojar el mundo en la red de Multiverse. Si, en cambio, se desea comercializar el entorno virtual, la plataforma ofrece dos opciones: ingresos compartidos o pagar por adelantado.

La forma de trabajar también puede hacerse según convenga al cliente: todas las funcionalidades de la plataforma pueden descargarse (servidor, cliente y herramientas) de manera gratuita, y construir al ritmo deseado el mundo virtual. Mientras que se esté trabajando en la construcción, o utilizando el mundo sin obtener ganancias monetarias, no hay que pagar nada. En el momento en el que se cobre a los usuarios por acceder al juego o por poner anuncios, la compañía Multiverse y el equipo compartirían los beneficios obtenidos. En este caso, el mundo virtual estaría alojado dentro de la red Multiverse, recogiendo éstos los beneficios y quedándose con un 10% del total, además de los gastos bancarios asociados a las transacciones.

Multiverse ofrece diferentes grados de soporte o ayudas según lo que el cliente desee:

- Básico.

  Es el nivel inicial y simplemente ofrece acceso a los foros de desarrolladores y a la Wiki del Desarrollador en Multiverse. No conlleva coste alguno.

- Nivel Premium de Plata.

  Permite el acceso a los foros privados, dotando a los mensajes propuestos en los mismos de cierta prioridad y garantizando una respuesta rápida por un equipo experto en el tema. Ofrece también actualizaciones, y acceso a una herramienta de CRM para avisar de posibles errores, problemas o comentarios. El precio es de 1.000$/mes.

- Nivel Premium de Oro.

  Además de las opciones incluidas en el nivel de plata, se garantiza una hora semanal con el equipo técnico para hablar sobre posibles mejoras, revisar algunas cuestiones, etc., y acceso vía e-mail al equipo de desarrollo de apoyo. En definitiva, un seguimiento completo y accesibilidad total al equipo de apoyo proporcionado por Multiverse. El coste es de 50.000$/año.

- Nivel Premium de Platino.

  Este nivel mejora las condiciones aportadas en el nivel de oro, como aumentar a dos horas semanales las revisiones por teléfono, comunicación directa con el equipo de apoyo y prioridad para ayudar a solventar los posibles problemas que puedan surgir. El precio es 10.000$/mes o 100.000$/año.

Asimismo, se pueden contratar servicios de consultoría esporádicamente pagando 2.500$ por día más gastos, o consultoría para la estrategia de diseño por 2.500$ al día.

3.2.2. Project Wonderland

Project Wonderland es una herramienta open source y en Java para crear mundos virtuales colaborativos en 3D. Ha sido desarrollada por Sun Microsystems Laboratories y la comunidad open-source. Actualmente la versión que existe es la 0.4, sacada en agosto de 2008, aunque se está trabajando en la versión 0.5 [82].

Los usuarios que utilicen estos mundos podrán comunicarse con alta fidelidad a través de audio, compartir aplicaciones y documentos y realizar negocios reales. Es completamente extensible, pues los desarrolladores y artistas gráficos pueden ampliar su funcionalidad para crear nuevos mundos y nuevas características en un mundo ya existente, o incluso nuevos comportamientos para los objetos o avatares.

Wonderland garantiza que provee un entorno robusto en cuanto a seguridad, escalabilidad, fiabilidad y funcionalidad en el que las organizaciones pueden confiar. En cuanto al precio, la plataforma es completamente gratis.

Como Wonderland está basado en Java, es necesario tener instalado Java Runtime Environment (JRE). Para utilizarlo no es necesario descargarse el software cliente, aunque sí que es posible, si no que basta con conectarse a un mundo disponible públicamente alojado en internet a través de su URL. En el caso de querer bajarse la aplicación a un entorno local, el espacio varía dependiendo de la plataforma, siendo 134 MB para Windows, 132 MB para Mac OS X, 139 MB para Solaris y 147 MB para Linux.

3.2.3. VRSpace

VRSpace es un software comunitario en 3D, modular y multiplataforma. Algunas de las características que incluye son chat, tanto con otras personas como con bots, aplicaciones, mensajería, plug-ins, persistencia en archivos de texto o bases de datos, un editor de mundos, etc. [83].

La utilidad de estos mundos queda en manos del desarrollador: juegos, negocios, enseñanza, etc.

VRSpace se compone de:

- VRSpace SDK o Space Development Kit. Es un conjunto de herramientas para construir modelos 3D, montar y ver mundos desarrollados en VRML.
- VRSpace Worlds es una colección de mundos en VRLM gratis construidos por los miembros de VRSpace.

VRSpace es un software libre, por lo que se puede redistribuir y modificar. Sus componentes son el servidor, plug-ins para añadir funcionalidad a la plataforma, el cliente y las utilidades para hacer más sencillo el proceso de elaboración de un entorno tridimensional.

Las tecnologías utilizadas son muy variadas con el fin de lograr su objetivo: una experiencia en 3D multiusuario. Todo lo que VRSpace ha utilizado es gratis y open source.

3.2.4. Vizard

Vizard es una herramienta para construir contenido 3D interactivo con capacidades de realidad virtual. Está diseñado para hacer una rápida creación de un prototipo, proveyendo los recursos necesarios para que dicho prototipo se convierta en realidad sin tener en cuenta la dificultad [84].

Con Vizard se puede:

- Construir mundos instantáneamente.
- Completar largos proyectos en poco tiempo.
- Importar recursos 3D y multimedia.
- Conectar directamente con hardware de realidad virtual.
- Comprobar la potencia y simplicidad de Python.
- Utilizar APIs gratis para crear efectos personalizados.
- Controlar en tiempo real los proyectos.
- Aprovecharse de los beneficios de una comunidad de usuarios con diez años de experiencia.

Vizard garantiza soportar aparatos de realidad virtual, gafas de LCD, proyectores, sonido 3D, guantes, etc.

Algunas de las características más destacables son:

- Formatos de modelos 3D extensibles.
- Herramientas para crear personajes humanos.
- Convertir los scripts en ejecutables, pudiendo utilizarse en cualquier ordenador.
- Motion Capture para captar los movimientos de los personajes en tiempo real.

- Plug-in para aceptar cámaras de video.
- Interfaz de usuario visual, sin necesidad de programar el contenido del mundo.
- Soporte para casi todos los dispositivos de realizad virtual.
- Páginas HTML embebidas para mostrar instantáneamente texto e imágenes.
- Acepta numerosos dispositivos como ratón, teclado, gamepads, joysticks, volantes, etc.
- Multiusuario.
- Acepta plug-ins para aumentar las funcionalidades.

Existen varias ediciones: Lite, Developer y Enterprise. La primera es ideal para realizar pruebas y ver cómo funciona la herramienta. La segunda ofrece un aspecto más profesional del mundo creado. La última aporta dos opciones importantes: la capacidad de ejecutar el mundo en un cluster y la posibilidad de crear un ejecutable.

Aquí se muestra un cuadro con las diferencias entre cada una de ellas.

Edition	Lite	Dev	Enterprise
Integrated development environment	✓	✓	✓
Community access (forum)	✓	✓	✓
Avatar support	✓	✓	✓
All VR components	✓	✓	✓
Fullscreen mode	X*	✓	✓
Unbranded viewer	X	✓	✓
Commercial license	X	✓	✓
Support/Maintenance (12% of list price)	X	✓	✓
CD ROM	X	✓	✓
Plug-in SDK	X	✓	✓
Publish as executable	X*	✓	✓
Publish as executable (protected)	X	X	✓
Royalty free non-commercial executables	X	✓	✓
Royalty free commercial executables	X	X	✓
Cluster support	X*	X*	✓
Live Characters plugin available	X	X	✓

* Available in restricted mode

Imagen 52: Comparativa entre las ediciones de Vizard

Existen numerosos precios dependiendo de las ediciones, licencias, etc., habiendo incluso una versión de evaluación gratuita. La edición Lite son 49$. La edición de desarrolladores (Development) va desde 4.004$ la más barata a 9.480$ la más cara. Por último, la edición

Enterprise abarca desde 6.404$ hasta 15.170$. Además es posible adquirir plug-ins, soporte, diseños, etc.

3.2.5. HeroEngine

HeroEngine ayuda en el proceso de construir juegos multijugador masivos online (MMO). Para ello, permite la colaboración en tiempo real de todo el equipo de desarrollo, diseñadores, constructores del mundo, productores, clientes, etc. [85].

Las ventajas que en la propia página se especifican son:

- Desarrollo colaborativo en vivo, todo el mundo trabaja online independientemente de su localización geográfica.
- Crear el juego en tiempo real obteniendo un feedback instantáneo del trabajo realizado.
- Sin asunciones ni restricciones.
- Acceso completo a "Hero's Journey® Reference", un juego para usarlo como referencia y ejemplo.
- Middleware integrado.

Las características más destacables son:

- La herramienta de diseño facilitada es simple de usar, con un panel de control que será accesible para realizar cualquier diseño dentro del mundo: añadir objetos, personajes, etc.
- Todo el sistema está integrado en un entorno de trabajo colaborativo.
- HeroEngine incluye, además, una herramienta para la gestión del proyecto. Se pueden mandar notas entre los integrantes del proyecto, gestionar las tareas, etc.
- Contiene un sistema de animación y de caracterización de los personajes.
- Dispone de servidores escalables y adaptables a las necesidades del cliente, incluyendo la opción de distribuir los procesos en un cluster.

Las distintas licencias se ajustan al cliente y a sus necesidades de financiación y distribución.

- Full Worldwide Royalty-Free. Esta licencia incluye cliente, servidor, conjunto de herramientas, código fuente, una copia del juego "Hero's Journey" y documentación. El mundo virtual será distribuido por todo el mundo.

- Regional. Incluye lo mismo que la anterior, aunque está especialmente diseñada para equipos que quieran distribuir el mundo virtual en una única y determinada región.

- Royalty Bearing. Mantiene los costes iniciales bajos. Puede distribuirse por todo el mundo o por una región.

3.2.6. Comparativa

	Multiverse	Project Wonderland	VRSpace	Vizard	HeroEngine
Precio	Gratis / Beneficios compartidos / Pago por adelantado a negociar	Gratis	Gratis	Variable, desde 49$ hasta 15.170$	Variable. Contactar con la empresa
Multiusuario	Sí	Sí	Sí	Sí	Sí
Alojamiento en sus servidores	Sí, excepto si se paga por adelantado	Sí	Sí	No	Sí
Soporte técnico	Sí. Varios niveles con distintos precios	Mediante foros	Foro, e-mails	Sí. Varios niveles con distintos precios	Sí
VOIP	Sin especificar	Sí	No	Sí	Sin especificar
Mensajería instantánea	Sí	Sí	Sí	Sí	Sí
Hardware de Realidad Virtual	No	No	No	Sí	No
Herramientas proporcionadas	Maya, 3ds Max, Blender, XSI y Milkshape	Maya, Blender, Wings 3D, Sketchup, 3D Studio Max, Lightwave, Art of Illusion	Wings3D, Blender	Sin especificar	Herramientas propias de HeroEngine

Plataforma del cliente	Windows Vista, Windows XP (x86 y x64), Mac, Linux	Linux, Windows XP, Solaris x86, Mac OSX	Windows, Linux, Mac	Windows 2000, XP	Windows, Linux
Plataforma del servidor	Linux Fedora Core 4 recomendado. Soporta Windows Vista, Windows XP (x86 y x64)	Linux, Solaris	Windows, Linux, Mac	Sin especificar	Windows, Linux
Lenguajes	Java, Python	Java	VRML, Java	Python	Sin especificar
Espacio en el cliente	Alrededor de 10 MB	Entorno web: no es necesario. Aplicación en el cliente: desde 132 MB hasta 147 MB	Sin especificar	85 MB	Sin especificar
Almacenamiento	MySQL	Sin especificar	Archivos de texto y Bases de Datos SQL	Sin especificar	Base de Datos, con la información replicada

Tabla 3: Comparativa entre herramientas de mundos virtuales

3.3. Cloud computing

3.3.1. Google App Engine

Según la propia página de Google: “App Engine ofrece una completa pila de desarrollo que emplea tecnologías habituales para crear y alojar aplicaciones Web.” Entre las ventajas que ofrece es que el cliente puede dotar a su aplicación web con el nombre de domino que desee, o bien utilizar un nombre dentro del dominio gratuito que Google proporciona [86].

Los lenguajes que acepta son Java, o cualquier lenguaje interpretable por JVM, como JavaScript o Ruby, y Python. Gracias al entorno en tiempo de ejecución de ambos lenguajes, Google asegura que las aplicaciones se ejecutarán de manera rápida, segura y sin interferencias con otras aplicaciones del sistema.

En cuanto a las tasas, sólo se paga por lo que se utiliza, sin costes de configuración. Todos los recursos, como el almacenamiento y el ancho de banda, se miden por gigabytes, facturándose según unas determinadas tasas. Es posible, asimismo, poner una cantidad máxima de recursos en función del dinero que el cliente esté dispuesto a gastarse.

App Engine incluye las siguientes funciones:

- Servidor Web dinámico, compatible con las tecnologías más comunes.
- Almacenamiento permanente, pudiendo realizar sobre él consultas, ordenar los datos y realizar transacciones.
- Escalado automático y balanceo de carga.
- API para autenticar usuarios y enviar correo a través de las cuentas de Google.
- Un entorno de desarrollo local que simula App Engine en el equipo del cliente.
- Tareas programadas para activar eventos en momentos determinados y en intervalos regulares.

Esta herramienta proporciona los siguientes servicios:

- Extracción de URL: permite a las aplicaciones acceder a recursos de internet mediante su dirección.
- Correo: se pueden enviar mensajes de correo electrónico, utilizando la infraestructura de Google.
- Memcache: proporciona a la aplicación el servicio de memoria caché de alto rendimiento, accesible desde varias instancias de la aplicación.
- Manipulación de imágenes: se pueden recortar, rotar o ajustar el tamaño de imágenes.
- Tareas programadas: permite programar tareas que se van a ejecutar en intervalos regulares.

Crear una aplicación en App Engine no conlleva coste alguno, siempre y cuando el espacio necesario no exceda los 500 MB y hasta cinco millones de visitas mensuales. En cualquier momento se puede activar la facturación, establecer un presupuesto diario máximo y asignar ese presupuesto a los distintos recursos según el interés del cliente. Es posible registrar hasta diez aplicaciones por desarrollador.

Como apunte, comentar que Google App Engine utiliza como almacén de datos el almacenamiento de objetos sin esquema a través del API JDO [113]. Éste proporciona una interfaz en la que se almacenan los objetos con datos en los mismos como si se guardaran en una base de datos tradicional, pudiendo hacer consultas y transacciones. Se puede utilizar

cualquier tipo de base de datos sin necesidad de utilizar un código concreto, ya sean relacionales, jerárquicas o de objetos [114].

Lo único que hace falta para que funcione son añadir las librerías, proporcionadas por el propio Google, y el archivo jdoconfig.xml, archivo de configuración, cuyo contenido también lo suministra Google en la página sobre JDO [114].

3.3.2. Azure

Windows Azure ofrece una plataforma intuitiva, fiable y potente para la creación de aplicaciones y servicios Web [87]. Está comprometida con Windows Azure, un sistema operativo que funciona como un servicio, SQL Azure para el almacenamiento de datos como una base de datos relacional, y servicios .Net. Su arquitectura a través de .Net permite a los desarrolladores elegir si construir aplicaciones Web que funcionen sobre PCs, servidores, o soluciones híbridas.

Todos los servicios de Cloud Computing que ofrece pueden usarse tanto conjuntamente como por separado. Permiten:

- A los desarrolladores usar sus habilidades para desarrollar aplicaciones Web.
- A los integradores de sistemas proveer servicios a sus clientes a un coste más efectivo.
- A empresas de cualquier tamaño rápidas respuestas a los cambios de negocio necesarios.

Los componentes de la plataforma son:

- Windows® Azure™ para el hospedaje y la administración de servicios, almacenamiento escalable de bajo nivel, computación y redes.
- Microsoft SQL Services para una gran variedad de servicios y reportes de bases de datos.
- Microsoft .NET Services, que son implementaciones basadas en servicios de conceptos .NET Framework familiares, como flujo de trabajo y control de acceso.
- Live Services para brindar a los usuarios una forma consistente de almacenar, compartir y sincronizar documentos, fotografías, archivos e información a través de sus PCs, teléfonos, aplicaciones de PC y sitios web.
- Microsoft SharePoint y Dynamics CRM Services para contenido empresarial, colaboración y desarrollo de soluciones rápidas en la nube.

El servicio gratuito incluye 50GB de almacenamiento y un ancho de banda de 20GB por día. Para unas condiciones mejores, es necesario pagar una cuota, dependiendo la cantidad de ésta según el uso que se haga de los recursos de Azure.

3.3.3. Amazon EC2

Amazon Elastic Compute Cloud (Amazon EC2) es un servicio Web que proporciona capacidad de cómputo de una manera flexible, ofreciendo una computación escalable a los desarrolladores [88].

Gracias a su interfaz, el cliente puede configurar la capacidad sin esfuerzo, permitiendo realizar un control completo de los recursos. Además, reduce el tiempo necesario para iniciar nuevas instancias del servidor, haciendo que se puedan crear o destruir rápidamente para adaptarse a las necesidades del cliente. Asimismo, ofrece a los desarrolladores herramientas para la construcción de sus aplicaciones.

Amazon presenta un entorno en el que alojar aplicaciones en una gran variedad de sistemas operativos, personalizando las opciones de dicho entorno y gestionando los permisos de acceso.

Las características de Amazon EC2 más destacables son:

- Elasticidad, dando la posibilidad al cliente de incrementar o disminuir la capacidad en minutos, pudiendo tener hasta miles de instancias en el servidor a un mismo tiempo.
- El cliente tiene el control absoluto sobre sus instancias, pudiendo acceder a ellas con permisos totales. Las instancias pueden ser reiniciadas remotamente usando APIs.
- Amazon proporciona otros servicios que pueden ser usados conjuntamente con EC2: Amazon Simple Storage Service (S3), Amazon SimpleDB y Amazon Simple Queue Service (Amazon SQS). De esta manera, se proporciona una solución completa a los desarrolladores.
- Amazon EC2 es un entorno altamente fiable.
- Amazon EC2 proporciona numerosos mecanismos de seguridad, como firewalls configurables que controlan la comunicación de fuera a las instancias y entre ellas, aislamiento de las instancias, etc.
- Se paga únicamente por lo que se consume, pudiendo ser de dos maneras: utilizando instancias según sea necesario, o pagando una cantidad determinada de éstas.

El precio varía dependiendo de la modalidad elegida y de si se usa sobre Linux o sobre Windows, siendo esto último más caro en general. Los precios son los siguientes:

United States | Europe

Standard On-Demand Instances	Linux/UNIX Usage	Windows Usage
Small (Default)	$0.11 per hour	$0.135 per hour
Large	$0.44 per hour	$0.54 per hour
Extra Large	$0.88 per hour	$1.08 per hour
High CPU On-Demand Instances	**Linux/UNIX Usage**	**Windows Usage**
Medium	$0.22 per hour	$0.32 per hour
Extra Large	$0.88 per hour	$1.28 per hour

United States | Europe

Linux/UNIX	One-time Fee		
Standard Reserved Instances	**1 yr Term**	**3 yr Term**	**Usage**
Small (Default)	$227.50	$350	$0.04 per hour
Large	$910	$1400	$0.16 per hour
Extra Large	$1820	$2800	$0.32 per hour
High CPU Reserved Instances	**1 yr Term**	**3 yr Term**	**Usage**
Medium	$455	$700	$0.08 per hour
Extra Large	$1820	$2800	$0.32 per hour

Imagen 53: Precios Amazon EC2

Además, se paga por la transferencia de datos y por el resto de servicios (S3, SQS, etc.) que se deseen.

3.3.4. Comparativa

Esta comparativa ha sido construida con los datos apartados en las páginas de cada servicio, así como del artículo de Dmitry Sotnikov [89].

	Google App Engine	Azure	Amazon EC2
Arquitectura	El cliente escribe su aplicación Web en Java o Python con una serie de limitaciones específicas proporcionadas por Google	El cliente provee el código .NET, el cual Microsoft despliega en un Windows 2008 virtual de acuerdo a las especificaciones del entorno del cliente	Permite subir las imágenes de la máquina virtual XEN a la infraestructura y da APIs para instanciarlas y gestionarlas

Balanceo de carga	Sí	Sí	Sí
Almacenamiento	Bases de datos y APIs de almacenamiento de datos	Aplicaciones de almacenamiento y servicios SQL	S3 y SimpleDB
Cola de mensajes para la comunicación	No	Colas en el almacenamiento de Azure	Simple Queue Service (SQS)
Integración con otros servicios	Sí, con los servicios existentes de Google: autenticación, mail, calendario, contactos, documentos, imágenes, YouTube	Con los llamados servicio .Net: control de acceso, contactos, mail, mapas. No integrados completamente	No
Vinculado con el centro de datos del vendedor	Sí	Sí	Sí
Herramientas de desarrollo	Sí, herramientas básicas de edición, simulación y despliegue. Lenguaje limitado a Java y Python	Sí, integración con Visual Studio. Soporta todos los lenguajes .Net	No aplicable
Precio	Cómputo: 0.10$/hora Almacenamiento: 0.15$/GB/mes Ancho de banda: 0.10$ entrada / 0.12$ salida / GB	Cómputo: 0.12$/hora Almacenamiento: 0.15$/GB/mes Transacciones: 0.01$ / 10K Ancho de banda: 0.10$ entrada / 0.15$ salida / GB	Transferencia de entrada: 0.10$/GB Transferencia de salida: desde 0.17$/GB/mes hasta 0.10$/GB/mes dependiendo de la cantidad transferida

Tabla 4: Comparativa entre servicios de Cloud Computing

3.4. Ontologías

3.4.1. Protégé

Protégé es una herramienta desarrollada en los laboratorios de SMI (Stanford Medical Informatics) de la Universidad de Stanford. Es una plataforma de código abierto, gratuita, que provee un conjunto de herramientas de escritorio para construir modelos de un dominio y aplicaciones basadas en el conocimiento a través de ontologías [73].

Permite modelar el conocimiento creando, visualizando y manipulando ontologías en varios formatos y lenguajes (OWL, OIL, RDF, Prolog, FLogic). Además, posee un editor central donde se pueden adjuntar una serie de plug-ins para extender la funcionalidad de la herramienta.

La plataforma Protégé soporta dos vías principales para modelar ontologías:

- Protégé-Frames.

  Este editor permite a los usuarios construir ontologías que están basadas en marcos, según el protocolo OKBC (Open Knowledge Base Connectivity). En este modelo, una ontología consiste en un conjunto de clases organizadas en jerarquías para representar el dominio de los conceptos, un conjunto de slots o propiedades asociadas a las clases para describir sus características y relaciones, y un conjunto de instancias de esas clases.

  Proporciona un editor con una completa interfaz de usuario y un servidor de conocimiento para ayudar a los usuarios en la construcción y almacenamiento de ontologías basadas en marcos, personalizando la estructura de los datos e introduciendo las instancias.

  Las capacidades de esta aplicación son:

  - Un amplio conjunto de elementos en la interfaz que pueden ser personalizados para permitir al usuario modelar el conocimiento e introducir datos de manera sencilla.
  - Una arquitectura basada en plug-ins que puede ser extensible gracias a la adición de elementos personalizados, como componentes gráficos, imágenes, vídeos, varias formas de almacenamiento (RDF, XML, HTML, bases de datos) y herramientas adicionales que complementan al editor.
  - Una API basada en Java que permite a los plug-ins y otras aplicaciones acceder y visualizar ontologías creadas con este editor.

- Protégé-OWL.

  El editor Protégé-OWL es una extensión de Protégé que permite a los usuarios construir ontologías para la Web Semántica, en particular en el lenguaje OWL.

  Las características de Protégé-OWL son:

  - Cargar y guardar ontologías construidas en OWL y en RDF.
  - Editar y visualizar clases, propiedades y reglas en el lenguaje SWRL.
  - Definir características lógicas de las clases como expresiones de OWL.

- Ejecutar razonadores como una descripción lógica de clasificadores.
- Editar instancias de OWL para la Web Semántica.

3.4.2. Ontolingua

Ontolingua es una aplicación Web de libre acceso creada por el KSL (Knowledge Software Laboratory), de la Universidad de Stanford, en 1997, que permite crear, editar, modificar y utilizar ontologías realizadas en diversos lenguajes (KIF, Ontolingua, OKBC, LOOM, etc.). Está compuesta por varias herramientas, siendo la principal el editor de ontologías. Además, cuenta con Webster, útil para obtener definiciones de términos, un servidor de OKBC, y Chiamera, para combinar distintas ontologías y traducir éstas entre diversos lenguajes (Loom, Prolog, CLIPS, etc.) [74].

El servidor de Ontolingua está provisto de un repositorio de ontologías, y permite que éstas sean modificadas, que se creen otras nuevas e incluso que se integren varias de ellas. La herramienta, asimismo, admite que varios usuarios trabajen de manera colaborativa en el desarrollo de la ontología. Otra característica importante es la posibilidad que ofrece de acceder a las ontologías a través de otros programas, lo que permite, por ejemplo, a agentes inteligentes buscar información en ellas. Por último, es posible exportar la ontología realizada y así poder integrarla en cualquier aplicación que soporte el uso de estas estructuras de conocimiento [75].

Una vez la ontología está en el servidor de Ontolingua, se puede acceder a ella de dos maneras [76]:

- Conectándose al servidor OKBC desde un cliente remoto. Si se hace así, la aplicación remota accede al servidor pudiendo usar cualquiera de los lenguajes que éste reconozca, como Java, C, Lisp, etc. Así, las ontologías pueden ser leídas o actualizadas mediante primitivas del protocolo OKBC.
- Usando el editor de ontologías para traducir la ontología en un lenguaje de implementación. Ontolingua Server puede exportar la ontología a los lenguajes KIF, LOOM, CLIPS, CORBA y Prolog. Una vez se ha generado el nuevo fichero y el usuario lo tiene almacenado localmente, puede ser usado por las aplicaciones deseadas.

3.4.3. OntoEdit

OntoEdit es un editor gráfico de ontologías en un entorno Web o en una Intranet. Fue diseñado en la Universidad de Karlsruhe, aunque ahora lo comercializa Ontoprise GmbH.

Existen dos versiones del programa, una gratuita, OntoEdit Free, y otra versión en la que es necesario pagar una licencia, OntoEdit Professional. La primera puede descargarse e instalarse localmente. La segunda incluye un mecanismo para realizar inferencias automáticas sobre los datos, herramientas gráficas de consulta, módulos para importación y exportación, editores de reglas gráficos, etc. [77].

Permite la representación semántica de lenguajes conceptuales y estructuras mediante conceptos, sus jerarquías, relaciones y axiomas. La característica principal que ofrece este editor es que, además de poder construir la ontología, ésta se almacena en la base de datos relacional proporcionada por la herramienta. Asimismo, soporta numerosos lenguajes de especificación como DAML+OIL, RDFs, OXML y F-Logic. Se ajusta a las necesidades del usuario, tanto en la interfaz como en las funcionalidades, permitiendo que se instalen plug-ins para ofrecer más opciones.

Existen varias versiones del programa. En la última versión es posible:

- Importar las estructuras del directorio.
- Importar tablas de Excel.
- Construir reglas gráficas.
- Dispone de reglas que eliminan fallos en la visualización del gráfico.
- Visualizar y editar ontologías en un gráfico.

3.4.4. OilEd

OilEd es un editor gráfico de ontologías que permite desarrollarlas en diversos lenguajes de especificación. Fue diseñado por la Universidad de Manchester, la Universidad de Ámsterdam e Interprice GmbH [78].

En un principio se orientó al lenguaje DAML+OIL, aunque actualmente se ha expandido a RDF y OWL. Su característica más destacable es su razonador “FaCT”, utilizado para comprobar si una ontología es consistente.

OilEd permite a los usuarios:

- Construir ontologías.
- Usar el razonador FaCT para comprobar la consistencia de las ontologías y añadir relaciones de jerarquía implícitas.
- Exportar ontologías en un amplio número de formatos, incluidos OIL-RDF y DAML-RDF.

La intención de esta herramienta es proporcionar un editor simple y gratuito que demuestre la usabilidad de OIL a la vez que se estimula su uso. Si bien ofrece las funcionalidades básicas para la construcción de ontologías, no es un entorno de desarrollo completo de las mismas: no soporta ontologías a gran escala, la migración e integración de ontologías, control de versiones, etc.

3.4.5. Comparativa

La información para realizar esta comparativa se ha extraído de las páginas especificadas en los sendos apartados de las herramientas y en los artículos de las referencias [75], [76], [79] y [80].

	Protégé	Ontolingua	OntoEdit	OilEd
Precio	Acceso Web gratuito	Open Source	Freeware / Licencia	Freeware
Arquitectura software	Independiente	Cliente-Servidor	Independiente y cliente-servidor	Independiente
Extensibilidad	Mediante plug-ins	Mediante plug-ins	Mediante plug-ins	No
Lenguajes compatibles	XML, RDFs, DAML+OIL, OWL, F-Logic, HTML	KIF, Prolog, LOOM, OKBC, IDL, CML, Clips	XML, RDFs, F-Logic, DAML+OIL	OIL, RDFs, DAML+OIL, Owl, SHIQ, HTML
Almacenamiento de la ontología	BBDD compatibles con JDBC, ficheros de texto, etc.	En el servidor	Ficheros de la base de datos	En ficheros *.daml*
Acceso	A través de Servlets o Applets	Vía Web a través de navegador	No proporciona acceso Web	XML Schema muy limitado
Consistencia	Mediante plug-ins	En tiempo real durante su construcción	A través del plug-in OntoBroker	A través de su razonador FaCT
Lenguaje de axiomas	PAL	KIF	F-Logic	DAML+OIL
Motor de inferencias	Sí, PAL	No	sí, OntoBroker	Sí, FaCT

Tabla 5: Comparativa entre herramientas de ontologías

4. ANÁLISIS

Una vez desarrolladas las bases tecnológicas sobre las que debe asentarse el sistema, resulta conveniente analizar las necesidades de los usuarios para poder crearlo acorde a las mismas de modo que cumpla con su cometido.

En este sentido, incentivar al futuro alumnado mediante diversas técnicas que den dinamismo al sistema virtual mejoraría la motivación y favorecería el desarrollo de las clases, llegando éstas a convertirse en un medio más gratificante para ellos que la asistencia física.

Para ello, a continuación se especifican los requisitos de usuario del sistema, gracias a los cuales se determinarán las características principales del mismo. Seguidamente se procederá a realizar un análisis exhaustivo de las herramientas encontradas para cada una de las tecnologías implicadas en el proyecto, a fin de establecer cuáles son las más convenientes en este caso concreto. Por último, mediante los requisitos de software, se detallará cómo deben llevarse a cabo los requerimientos de usuario de una manera más técnica.

4.1. Requisitos de usuario

Los requisitos de usuario mencionados anteriormente son los siguientes:

ID	RU-001		
DESCRIPCIÓN	La aplicación debe proporcionar acceso a los usuarios desde cualquier ordenador a través de Internet.		
FUENTE	Cliente		
PRIORIDAD	☑ Alta	☐ Media	☐ Baja
NECESIDAD	☑ Esencial	☐ Deseable	☐ Opcional

Tabla 6: Requisito de Usuario RU-001

ID	RU-002		
DESCRIPCIÓN	La materia debe impartirse en un entorno virtual.		
FUENTE	Cliente		
PRIORIDAD	☑ Alta	☐ Media	☐ Baja
NECESIDAD	☑ Esencial	☐ Deseable	☐ Opcional

Tabla 7: Requisito de Usuario RU-002

ID	RU-003		
DESCRIPCIÓN	El temario cargado en el entorno virtual ha de ser el referente a Arquitectura de Software.		
FUENTE	Cliente		
PRIORIDAD	☑ Alta	☐ Media	☐ Baja
NECESIDAD	☑ Esencial	☐ Deseable	☐ Opcional

Tabla 8: Requisito de Usuario RU-003

ID	RU-004		
DESCRIPCIÓN	La gestión del entorno debe realizarse a través de los dispositivos teclado, ratón y micrófono.		
FUENTE	Cliente		
PRIORIDAD	☑ Alta	☐ Media	☐ Baja
NECESIDAD	☑ Esencial	☐ Deseable	☐ Opcional

Tabla 9: Requisito de Usuario RU-004

ID	RU-005		
DESCRIPCIÓN	Los usuarios podrán disponer de dispositivos de realidad virtual (gafas, guantes) para interactuar con el entorno.		
FUENTE	Cliente		
PRIORIDAD	☐ Alta	☐ Media	☑ Baja
NECESIDAD	☐ Esencial	☐ Deseable	☑ Opcional

Tabla 10: Requisito de Usuario RU-005

ID	RU-006		
DESCRIPCIÓN	Para impartir la materia, se seguirá una metodología de enseñanza diseñada específicamente para la Arquitectura de Software.		
FUENTE	Cliente		
PRIORIDAD	☑ Alta	☐ Media	☐ Baja
NECESIDAD	☑ Esencial	☐ Deseable	☐ Opcional

Tabla 11: Requisito de Usuario RU-006

4.2. Análisis de las herramientas

En el punto “3 Herramientas” del presente documento, se ha realizado un análisis acerca de las herramientas existentes en el mercado para cada una de las tecnologías que están previstas formen parte de este proyecto. En este punto se procederá a analizar cada una de ellas, viendo su utilidad global dentro del sistema a construir, y se tomará una decisión teniendo en cuenta su conveniencia para el cliente.

En primer lugar, es necesario mencionar el caso de mashups. Tras realizar un concienzudo examen de dicha tecnología, se ha llegado a la conclusión de que no es necesaria su inclusión dentro del proyecto, pues no aporta soluciones interesantes al cliente.

4.2.1. Mundos virtuales

El pilar fundamental sobre el que se sustenta este proyecto es el mundo virtual a desarrollar. Éste actuará como punto de contacto entre el cliente y el sistema. Por ello es de vital importancia la elección de una herramienta que permita construir un entorno en el que el usuario se sienta cómodo y perciba como real, haciendo que la inmersión de los alumnos en el entorno docente sea completa.

Puesto que el software tiene como objetivo plantear un mundo en el que se puedan ofrecer clases, la herramienta HeroEngine queda automáticamente descartada, pues está especializada en el desarrollo de juegos online multijugador.

El resto de herramientas son muy parecidas: todas proporcionan un entorno en el que desarrollar el mundo virtual, siendo común en todas ellas que el producto resultante es multiusuario, algo esencial para este proyecto. El espacio necesario en cliente para instalar la aplicación varía según la herramienta: Multiverse es la que menos espacio necesita, con 10 MB, seguida de Vizard con 85 MB. Project Wonderland ofrece la posibilidad de ejecutarlo online o de instalarlo en local, variando así el espacio necesario, siendo el máximo 147 MB. En cualquier caso, gracias a las capacidades computaciones actuales, este espacio puede ser considerado insignificante. Por otra parte, todas ellas facilitan los servidores necesarios para lanzar la aplicación, excepto Multiverse en el caso de realizar el pago por adelantado, y Vizard, que no lo hace en ninguna circunstancia.

En cuanto a las herramientas de construcción y creación de entornos 3D, la que mayor número de éstas proporciona es Project Wonderland, estando entre ellas las más conocidas como Maya, Blender o Wings 3D. Asimismo, Multiverse suministra un buen número de ellas, aunque en menor medida, mientras que VRSpace proporciona solo dos y Vizard no especifica ninguna, dando a entender que utiliza las suyas propias.

Las plataformas tanto del cliente como del servidor no son concluyentes, puesto que todas ellas admiten gran variedad, desde Windows a entornos Linux y Mac, a excepción de Vizard, aplicable sólo en los primeros.

Los lenguajes de programación son Java y Python, ambos extendidos entre los programadores, por lo que son fácilmente accesibles. Destacar que Project Wonderland únicamente acepta Java y Vizard solo Python. Respecto al sistema de almacenamiento utilizado por las diferentes herramientas, tan sólo Multiverse y Vizard especifican cuáles utilizan, siendo en ambos casos bases de datos basadas en SQL.

Es útil conocer el servicio técnico que ofrecen, pues en caso de contar con dificultades a la hora de realizar el mundo virtual es necesario saber cuáles son las opciones disponibles para resolverlas. Todos ellos proporcionan cierto soporte técnico, siendo Vizard y Multiverse los que ofrecen un trato más personalizado, aunque previo pago. En general, para el servicio gratuito, todas facilitan wikis, foros y comunicación vía correo electrónico.

Evidentemente, lo más importante para este proyecto es tener en cuenta cuáles son las posibilidades de comunicación que se ofrecen dentro del mundo virtual una vez concluido, pues va a ser la principal vía de docencia. El método que se utilizará más a menudo es la mensajería instantánea, pudiendo hacerse tanto entre profesor y alumno como entre alumnos, ya sea para realizar trabajos colaborativos o para resolverse dudas mutuamente. A tal fin, todas las herramientas propuestas ofrecen la tecnología necesaria para llevarla a cabo. Otra buena forma de comunicación es a través de voz, pudiendo hacer el trato más personalizado, por ejemplo, durante tutorías. En este caso únicamente Project Wonderland y Vizard ofrecen esta posibilidad.

Otra de las opciones es proporcionar hardware de realidad virtual, como gafas, guantes, etc. que hagan que el estudiante se sumerja en una experiencia más real. Si bien esto podría ser interesante en muchas ocasiones, para el caso que aquí atañe no sería tan necesario como, por ejemplo, si la materia impartida fueran clases de vuelo o cirugía. La Arquitectura de Software no precisa de un entorno virtual a ese nivel. En el caso de utilizarlo, además, se incurriría en el hecho de tener que pagar por una característica que no aportaría beneficios aparentes a la hora de realizar la docencia.

Por último, es imprescindible hablar del coste económico de cada una de las herramientas. Vizard oscila entre 49$ y 15.170$, siendo esta suma tan elevada debido a las técnicas de realidad virtual que soporta. Multiverse ofrece la posibilidad, si el software se realiza con ánimo de lucro, de dividir las ganancias o llegar a un acuerdo de pagar por adelantado cierta suma. Por otra parte, Project Wonderland y VRSpace son gratuitas.

Tomando en consideración lo dicho anteriormente, se pueden descartar HeroEngine por no tener un fin acorde a las pretensiones en este proyecto, y Vizard, puesto que no interesa el hardware de realidad virtual. Por ende, la herramienta que más se adecúa a las necesidades del sistema es Project Wonderland, ofreciendo mensajería instantánea y por voz como medio de comunicación, así como multitud de herramientas de diseño.

4.2.2. Cloud Computing

A pesar de que la herramienta de creación de mundos virtuales escogida, Project Wonderland, pone a disposición del cliente sus servidores, se ha tomado la decisión de no utilizar esta característica y hacer recaer dicha responsabilidad en una de las herramientas de Cloud Computing estudiadas. El fin es poder realizar una administración de los servidores más adecuada a las necesidades del sistema, favoreciéndose de las ventajas que ofrecen dichas

herramientas, como la escalabilidad, la utilización según las necesidades en cada instante y el consecuente pago en función dicha utilización, o el balanceo de carga, comunes a todas ellas.

Microsoft Azure y Amazon son empresas con sus propias arquitecturas de desarrollo, por lo que para la programación de los sistemas que vaya a alojar es necesario que se adecúen a sus restricciones, como la utilización de .NET en Azure. Google, en cambio, permite aplicaciones Web en Java o Python. Aunque el desarrollo de las aplicaciones se va a hacer en un entorno propio, tan sólo la herramienta de Google sería apropiada para este caso concreto por compatibilidad de lenguajes.

Siguiendo esta misma línea, la integración con otros servicios es más amplia con Google, puesto que Azure es exclusivo para servicios .NET, mientras que Amazon ni siquiera admite esta posibilidad. En cuanto a las bases de datos, Google ofrece variedad y APIS, mientras que Azure SQL y Amazon brindan sus propios productos, S3 y SimpleDB, debiendo pagar éstos de forma independiente.

Los precios son muy similares unos con otros, siendo diferencias tan minúsculas que no llegan a ser concluyentes en ninguno de los casos.

La herramienta más adecuada para este caso es Google App Engine, pues proporciona un entorno flexible sin ataduras empresariales como ocurre con Azure y Amazon.

4.2.3. Ontologías

El uso de las ontologías en este proyecto se basa en que son la base del conocimiento de la materia a impartir: en ella se pretende almacenar la información académica acerca de la Arquitectura del Software. En el estudio de las herramientas se propusieron cuatro que podían satisfacer las necesidades del proyecto.

Todas ellas tienen al menos una versión gratuita, por lo que el precio no es una variable a considerar en esta decisión.

Es importante la extensibilidad de la aplicación, aunque, de nuevo, en la mayoría de los casos, es fácilmente extensible gracias a plug-ins proporcionados por las propias compañías desarrolladoras de las herramientas, a excepción de OilEd, que no admite dicha funcionalidad. De esta misma manera, mediante plug-ins, es posible garantizar la consistencia del conocimiento en las herramientas Protégé y OntoEdit; OilEd tiene su propio razonador, mientras Ontolingua lo comprueba en tiempo real durante la construcción de la ontología. Igualmente esta última no tiene motor de inferencias, cosa que sí facilitan las otras tres.

Hasta aquí se pueden descartar las herramientas Ontolingua, por no disponer de motor de inferencias, y OilEd, al no ser extensible.

Si se tienen en cuenta los lenguajes compatibles, tanto Protégé como OntoEdit hacen uso de los más utilizados: XML, RDFs, F-Logic y DAML-OIL. En cuanto al acceso a la ontología, únicamente Protégé facilita acceso Web a través de Servlets o Applets, lo que sería extremadamente útil para el proyecto, teniendo además en cuenta que se realizará en Java. Considerando estas características, la herramienta que más se ajusta a las necesidades del sistema es Protégé.

4.2.4. Resumen

A modo de resumen de las soluciones elegidas en los apartados anteriores, se muestra la tabla siguiente:

TECNOLOGÍA	HERRAMIENTA
Mashups	No procede
Mundos virtuales	Project Wonderland
Cloud Computing	Google App Engine
Ontologías	Protégé

Tabla 12: Selección de herramientas definitivas

4.3. Requisitos Software

Los requisitos de software resultantes son:

ID	RS-001		
DESCRIPCIÓN	La aplicación deberá estar alojada en un servidor de tal forma que los usuarios puedan acceder a la misma haciendo uso de su conexión de red.		
TRAZABILIDAD	RU-001		
ESTABILIDAD	☑ Alta	☐ Media	☐ Baja
NECESIDAD	☑ Esencial	☐ Deseable	☐ Opcional

Tabla 13: Requisito Software RS-001

ID	RS-002		
DESCRIPCIÓN	Para el alojamiento de la aplicación se utilizarán los servicios ofrecidos por Google App Engine.		
TRAZABILIDAD	RU-001		
ESTABILIDAD	☑ Alta	☐ Media	☐ Baja
NECESIDAD	☑ Esencial	☐ Deseable	☐ Opcional

Tabla 14: Requisito Software RS-002

ID	RS-003		
DESCRIPCIÓN	La información de los usuarios y sus acciones dentro del entorno serán almacenadas utilizando el API JDO (Java Data Object), proporcionado por Google App Engine.		
TRAZABILIDAD	RU-001		
ESTABILIDAD	☑ Alta	☐ Media	☐ Baja
NECESIDAD	☑ Esencial	☐ Deseable	☐ Opcional

Tabla 15: Requisito Software RS-003

ID	RS-004		
DESCRIPCIÓN	El entorno gráfico de la aplicación será un mundo virtual en tres dimensiones.		
TRAZABILIDAD	RU-002		
ESTABILIDAD	☐ Alta	☑ Media	☐ Baja
NECESIDAD	☐ Esencial	☐ Deseable	☑ Opcional

Tabla 16: Requisito Software RS-004

ID	RS-005		
DESCRIPCIÓN	El mundo virtual deberá aceptar múltiples usuarios simultáneamente.		
TRAZABILIDAD	RU-001, RU-002		
ESTABILIDAD	☑ Alta	☐ Media	☐ Baja
NECESIDAD	☑ Esencial	☐ Deseable	☐ Opcional

Tabla 17: Requisito Software RS-005

ID	RS-006		
DESCRIPCIÓN	El desarrollo del entorno virtual será mediante la herramienta Project Wonderland.		
TRAZABILIDAD	RU-002		
ESTABILIDAD	☑ Alta	☐ Media	☐ Baja
NECESIDAD	☑ Esencial	☐ Deseable	☐ Opcional

Tabla 18: Requisito Software RS-006

ID	RS-007		
DESCRIPCIÓN	El entorno virtual en el que los usuarios interactúen contará con diversos entornos, cada uno destinado a una labor docente.		
TRAZABILIDAD	RU-002		
ESTABILIDAD	☐ Alta	☑ Media	☐ Baja
NECESIDAD	☐ Esencial	☐ Deseable	☑ Opcional

Tabla 19: Requisito Software RS-007

ID	RS-008		
DESCRIPCIÓN	Los usuarios podrán moverse por el entorno entrando a su voluntad en los diferentes entornos.		
TRAZABILIDAD	RU-002		
ESTABILIDAD	☐ Alta	☑ Media	☐ Baja
NECESIDAD	☐ Esencial	☐ Deseable	☑ Opcional

Tabla 20: Requisito Software RS-008

ID	RS-009		
DESCRIPCIÓN	El desarrollo del mundo virtual se realizará en el lenguaje de programación Java.		
TRAZABILIDAD	RU-002		
ESTABILIDAD	☑ Alta	☐ Media	☐ Baja
NECESIDAD	☑ Esencial	☐ Deseable	☐ Opcional

Tabla 21: Requisito Software RS-009

ID	RS-010		
DESCRIPCIÓN	El temario disponible en la aplicación será de la asignatura Arquitectura de Software.		
TRAZABILIDAD	RU-003		
ESTABILIDAD	☑ Alta	☐ Media	☐ Baja
NECESIDAD	☑ Esencial	☐ Deseable	☐ Opcional

Tabla 22: Requisito Software RS-010

ID	RS-011		
DESCRIPCIÓN	Los usuarios podrán moverse por el entorno mediante el ratón y las teclas de dirección del teclado.		
TRAZABILIDAD	RU-004		
ESTABILIDAD	☑ Alta	☐ Media	☐ Baja
NECESIDAD	☑ Esencial	☐ Deseable	☐ Opcional

Tabla 23: Requisito Software RS-011

ID	RS-012		
DESCRIPCIÓN	Los usuarios podrán seleccionar contenido temático mediante el puntero del ratón.		
TRAZABILIDAD	RU-004		
ESTABILIDAD	☑ Alta	☐ Media	☐ Baja
NECESIDAD	☑ Esencial	☐ Deseable	☐ Opcional

Tabla 24: Requisito Software RS-012

ID	RS-013		
DESCRIPCIÓN	Los usuarios podrán chatear mediante mensajes instantáneos introducidos por teclado.		
TRAZABILIDAD	RU-002, RU-004		
ESTABILIDAD	☑ Alta	☐ Media	☐ Baja
NECESIDAD	☑ Esencial	☐ Deseable	☐ Opcional

Tabla 25: Requisito Software RS-013

ID	RS-014		
DESCRIPCIÓN	Los usuarios podrán comunicarse mediante chat de voz.		
TRAZABILIDAD	RU-002, RU-004		
ESTABILIDAD	☑ Alta	☐ Media	☐ Baja
NECESIDAD	☑ Esencial	☐ Deseable	☐ Opcional

Tabla 26: Requisito Software RS-014

ID	RS-015		
DESCRIPCIÓN	Los profesores dispondrán de direcciones de correo, facilitadas a los alumnos, para poder consultar dudas de forma asíncrona.		
TRAZABILIDAD	RU-004		
ESTABILIDAD	☑ Alta	☐ Media	☐ Baja
NECESIDAD	☑ Esencial	☐ Deseable	☐ Opcional

Tabla 27: Requisito Software RS-015

ID	RS-016		
DESCRIPCIÓN	Los usuarios podrán introducirse en el entorno a través de gafas de realidad virtual.		
TRAZABILIDAD	RU-005		
ESTABILIDAD	☐ Alta	☐ Media	☑ Baja
NECESIDAD	☐ Esencial	☐ Deseable	☑ Opcional

Tabla 28: Requisito Software RS-016

ID	RS-017		
DESCRIPCIÓN	Los usuarios podrán interactuar con el entorno a través de guantes de realidad virtual.		
TRAZABILIDAD	RU-005		
ESTABILIDAD	☐ Alta	☐ Media	☑ Baja
NECESIDAD	☐ Esencial	☐ Deseable	☑ Opcional

Tabla 29: Requisito Software RS-017

ID	RS-018		
DESCRIPCIÓN	La información mostrada será dividida en función de las áreas temáticas de la materia impartida.		
TRAZABILIDAD	RU-006		
ESTABILIDAD	☑ Alta	☐ Media	☐ Baja
NECESIDAD	☑ Esencial	☐ Deseable	☐ Opcional

Tabla 30: Requisito Software RS-018

ID	RS-019		
DESCRIPCIÓN	La información acerca de la materia será almacenada en forma de ontología.		
TRAZABILIDAD	RU-003, RU-006		
ESTABILIDAD	☑ Alta	☐ Media	☐ Baja
NECESIDAD	☑ Esencial	☐ Deseable	☐ Opcional

Tabla 31: Requisito Software RS-019

ID	RS-020		
DESCRIPCIÓN	Para crear la ontología se utilizará el programa Protégé.		
TRAZABILIDAD	RU-006		
ESTABILIDAD	☑ Alta	☐ Media	☐ Baja
NECESIDAD	☑ Esencial	☐ Deseable	☐ Opcional

Tabla 32: Requisito Software RS-020

ID	RS-021		
DESCRIPCIÓN	Para la creación de la ontología, se creará un nuevo proyecto que siga un formato de ficheros OWL - RDF.		
TRAZABILIDAD	RU-006		
ESTABILIDAD	☑ Alta	☐ Media	☐ Baja
NECESIDAD	☑ Esencial	☐ Deseable	☐ Opcional

Tabla 33: Requisito Software RS-021

ID	RS-022		
DESCRIPCIÓN	Una vez creada, la ontología será exportada a formato RDF.		
TRAZABILIDAD	RU-006		
ESTABILIDAD	☑ Alta	☐ Media	☐ Baja
NECESIDAD	☑ Esencial	☐ Deseable	☐ Opcional

Tabla 34: Requisito Software RS-022

ID	RS-023		
DESCRIPCIÓN	Se seguirán los pasos necesarios para crear la metodología específica para Arquitectura de Software.		
TRAZABILIDAD	RU-006		
ESTABILIDAD	☑ Alta	☐ Media	☐ Baja
NECESIDAD	☑ Esencial	☐ Deseable	☐ Opcional

Tabla 35: Requisito Software RS-023

5. GUÍA PARA LA CREACIÓN DE ESCENARIOS VIRTUALES DE APRENDIZAJE

En el presente apartado del proyecto, se va a indicar cuáles son los pasos a seguir para la creación de un mundo virtual cuyo objetivo sea la enseñanza de una determinada materia. En un principio, se expondrán los puntos que se deben tener en cuenta a la hora de diseñar el entorno, continuando con la adaptación de dicha guía a un escenario de trabajo en el que se imparta Arquitectura de Software.

5.1. Definición de la guía para la creación un escenario virtual de aprendizaje

Cuando se va a desarrollar cualquier tipo de software, es necesario conocer cuáles son los objetivos esenciales del mismo a fin de realizar un correcto diseño que satisfaga tanto a clientes como usuarios. Si se habla de la enseñanza, el primordial propósito es transmitir a los estudiantes el conocimiento que se desea impartir para que, en un futuro, puedan utilizarlo o incluso transmitirlo ellos mismos. Evidentemente, cuando la enseñanza se realiza en un entorno virtual se necesita seguir otros pasos distintos a los tradicionales, utilizados en las habituales aulas físicas.

En primer lugar es necesario seleccionar el temario de la materia que se va a impartir, pues dependiendo de éste, si es eminentemente teórico o práctico, se tomarán unas decisiones u otras. Entre estas decisiones se encuentra seleccionar el número de entornos o ambientes disponibles en el mundo virtual. Dicho mundo podría tratarse de una simple habitación donde se encontraran todos los alumnos haciendo las tareas pertinentes o bien tratarse de distintas salas, cada una con una temática o ambiente distintos. Aquella persona que diseñe el mundo virtual deberá, como se ha mencionado, tener en cuenta los temas a impartir y las diferentes actividades dentro de cada uno de ellos para determinar el número de ambientes y qué papel deben representar cada uno de ellos.

Otra de las tareas que han de realizarse atendiendo al temario es la selección de las técnicas de aprendizaje que se van a usar en cada uno de los ambientes, dependiendo de qué se va a encargar de impartir cada uno de ellos. Hay que estudiar dentro de cada tema de la materia qué es lo que se va a necesitar para impartirlo: documentos teóricos, enlaces externos, ejercicios, prácticas, trabajos en grupo, etc. Con la información recogida en el punto “2.6 Técnicas de enseñanza”, donde se hizo un examen de las técnicas de enseñanza disponibles de una manera teórica, se puede dar una relación entre las técnicas y su finalidad:

- Para que el alumno aprenda correctamente siguiendo el planteamiento del mundo virtual y aproveche todos sus beneficios, es necesario que se sienta motivado. Hay dos formas diferentes de conseguir esto:
    - Mediante métodos de motivación intrínseca, tales como proponerle desafíos o problemas y suscitar su curiosidad mediante bibliografía externa interesante.
    - Mediante métodos de motivación extrínseca, reforzando sus resultados positivos y no dando tanta importancia cuando éstos sean negativos, dándole a entender que forma parte del proceso educativo. Otra técnica es plantear problemas que aumenten en dificultad gradualmente, siendo los primeros muy sencillos y acabando por los más complicados, los que supongan un reto para el alumno.
- La actitud del alumno frente a la materia es vital, siempre ha de ser positiva y no verla como algo inaccesible. Se puede plantear desde dos puntos de vista:
    - Proponer problemas con la posibilidad de resolverlos con la ayuda del profesor y bibliografía proporcionada especialmente para dichos problemas.
    - Plantear un calendario a seguir en el que se establezcan una serie de hitos, momento en el cual el alumno debe haber alcanzado unas determinadas metas.
- Para mantener el interés del alumno sobre la materia impartida se emplean técnicas como la evaluación o autoevaluación, y realización de problemas.
- Las estrategias de selección ayudan a que el alumno interiorice el conocimiento que se ha impartido de manera teórica. Se utiliza para ello la realización de esquemas, la toma de apuntes y notas y la selección de libros. Estas técnicas también sirven para que el alumno memorice dicho conocimiento.
- Los mapas conceptuales, en los que los términos y conceptos básicos del dominio de la materia impartida se relacionan unos con otros, son buenos para que el alumno organice el conocimiento en su mente, permitiendo una mayor comprensión del mismo.
- La realización de ejercicios propuestos, con dificultad gradual, permiten que el alumno ponga en práctica lo que ha aprendido, mejore su comprensión y facilite que se mantenga en su memoria a largo plazo.
    - Primero han de plantearse ejercicios fáciles parecidos entre sí. A continuación, hacer que el alumno, con ayuda del profesor, identifique los puntos comunes y generalice los problemas. Por último, se deben plantear problemas en diversas situaciones.

- Los ejercicios individuales con bibliografía y ayuda de los profesores proponen a los alumnos un reto a superar que, de ser realizado con éxito, favorece su autoestima y le motiva a seguir con la tarea. En caso de no realizar el ejercicio correctamente, es necesario que el profesor y él mismo lo presenten como parte del proceso de aprendizaje y establecer como meta la mejora personal a través de más ejercicios, buscando la superación en éstos.

- Los ejercicios en grupo mejoran la capacidad de comunicación del alumno al relacionarse y al exponer sus conocimientos a otros individuos que se encuentran a su mismo nivel.

    - Cuando se hace un trabajo colaborativo democrático, todos ellos proponen su solución, ayudándoles a ser críticos y objetivos con sus soluciones y la de sus compañeros, aprendiendo así de otros semejantes a él.

    - Cuando los alumnos asumen cada uno un role, los estudiantes adoptan diferentes perspectivas, permitiendo ver el problema desde distintos ángulos y haciendo que tengan una mentalidad más abierta respecto a los ejercicios plateados.

- La evaluación permite a alumno y profesor conocer sus avances respecto a la materia impartida:

    - Test y problemas de autoevaluación a lo largo del curso permiten al alumno conocer cuáles son los aspectos de la materia en los que debe hacer especial hincapié, así como demostrarse a sí mismo si comprende realmente lo estudiado, reforzándolo positivamente en caso afirmativo.

    - Un examen a final del curso permite al profesor conocer hasta qué punto el alumno ha adquirido los conocimientos impartidos, evaluando si ha aprobado el curso o no.

La siguiente tabla muestra un resumen de las técnicas y su función:

TÉCNICA	FUNCIÓN
Proposición de desafíos y problemas.	Motivación intrínseca del alumno.
Proposición de bibliografía externa.	Motivación intrínseca del alumno.
Refuerzo de resultados positivos.	Motivación extrínseca del alumno.
Ofrecer un punto de vista positivo a los resultados negativos.	Motivación extrínseca del alumno.
Planteamiento de problemas con dificultad gradual.	Motivación extrínseca del alumno. Le ayuda a familiarizarse con los problemas, presentándoselos como un reto, pudiendo resolver en un futuro otros similares en situaciones diferentes. Facilita que el alumno memorice y comprenda la materia, conservando el interés de éste hacia el tema estudiado.
Acceso a los profesores a la hora de realizar un problema.	Permite que los alumnos pidan ayuda y no se desmotiven cuando no sepan cómo resolver un problema.
Calendario de eventos de la asignatura.	El alumno se motiva alcanzando las metas, manteniendo un punto de vista positivo.
Autoevaluación.	Permite al alumno conocer su grado de avance en la materia. Se autorefuerza cuando obtiene resultados positivos. Ayuda a mantener el interés sobre la materia.
Realización de esquemas, toma de apuntes, toma de notas.	Ayuda al alumno a que interiorice y memorice el conocimiento sobre la materia.
Realización de mapas conceptuales con las relaciones entre los términos.	Ayudan a que el alumno comprenda y organice los términos de la materia, facilitando su memorización.
Planteamiento de problemas a resolver en grupo democrático.	Ayuda al alumno a ser crítico y objetivo con sus ideas y las de sus compañeros. Mejora su socialización.
Planteamiento de problemas a resolver en grupo con asunción de roles.	Permite al alumno afrontar un problema desde distintos puntos de vista, teniendo en cuenta la situación de sus compañeros. Mejora su socialización.
Evaluación.	Permite tanto a alumno y profesor conocer cuál es el grado de asimilación de la materia del primero de éstos.

Tabla 36: Resumen de técnicas y su función

Siguiendo con las dependencias del temario, se ha de seleccionar la bibliografía que se tendrá en cuenta a la hora de impartir la materia, permitiendo que los alumnos accedan a la

misma o a información relativa a ella, como, por ejemplo, nombre, autor y editorial en caso de tratarse de un libro o su enlace si se trata de un recurso informático.

Aparte de los alumnos, las otras entidades importantes que se relacionarán con el mundo virtual son los profesores, los cuales serán, de igual modo, personas reales. Éstos se encontrarán en el mundo virtual a unas horas determinadas, debiendo conocer el alumno cuáles son. Teniendo en cuenta la función del profesor, se establecerá cuántos de éstos son necesarios atendiendo al número de alumnos matriculados en el curso. Este número podrá variar dependiendo de la materia a impartir, de la carga práctica y teórica de la misma, y de su dificultad. A mayor carga práctica, menor número de estudiantes por profesor, pues así se les puede dedicar una mayor atención, igual que ocurriría cuando la dificultad de la materia es alta.

Cada estudiante y cada profesor contarán con un avatar personalizado que les sustituirá dentro del mundo virtual. Además de éstos, existirán avatares controlados por el programa informático que transmitirán conocimientos sobre el dominio de la materia al alumno utilizando la información proporcionada por la ontología construida a tal fin. Igualmente, podrá proponer ejercicios que hagan al estudiante practicar dichos conocimientos. El número de estos avatares debe ser establecido teniendo en cuenta qué parte de la materia se desea impartir, el número de alumnos y el número de temas.

Por último, hay que decidir cuáles serán el o los puntos de acceso al mundo virtual, así como si estarán todas las aulas disponibles las veinticuatro horas del día o no.

A modo de resumen, se enumerarán los pasos descritos en los párrafos anteriores según su ordenación.

1. Selección del temario a impartir.
2. Selección de las metodologías y técnicas de enseñanza.
3. Selección del número de entornos dentro del número virtual, así como el ambiente dentro de cada uno de ellos.
4. Selección del número de profesores y el cometido de éstos últimos en función del número de alumnos.
5. Configuración del número de avatares no reales que estarán disponibles para los alumnos.
6. Selección de los puntos de acceso al mundo.
7. Establecimiento del horario de los entornos del mundo virtual.

El siguiente diagrama muestra más claramente esta serie de pasos:

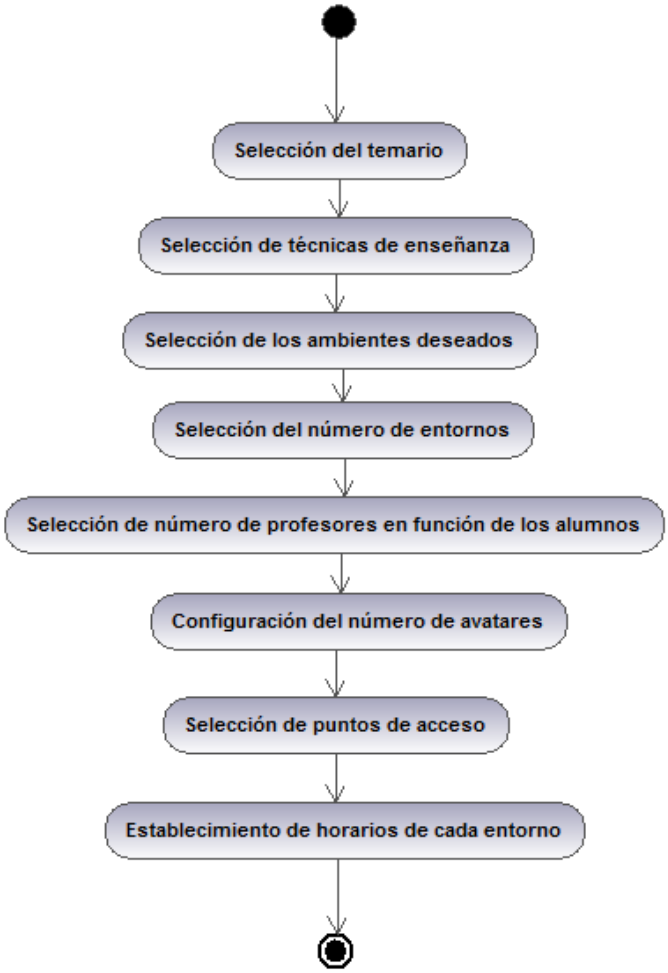

Imagen 54: Guía para la creación de entornos virtuales de enseñanza

5.2. Escenario de uso: Entorno virtual para Arquitectura de Software

Siguiendo los pasos que se han descrito en el apartado anterior, se va a proceder a realizar una especificación de los mismos para el caso que aquí compete: enseñar la asignatura Arquitectura de Software.

La materia constará de cinco temas:

1. Introducción a la arquitectura de software.
2. Introducción a los patrones.
3. Patrones de arquitectura.
4. Patrones de diseño.

5. Patrones de arquitectura Web.

Los dos primeros temas son eminentemente teóricos. Los tres últimos tienen una parte de teoría, aunque la carga principal es práctica, pues es así como los alumnos sabrán enfrentarse a futuras situaciones en las que tengan que hacer uso de lo aprendido con esta materia. Así pues, se necesitan ambientes para estudiar temas teóricos y donde poder realizar problemas. De la misma forma, estos últimos deberán poder realizarse tanto individualmente como en grupos, favoreciendo así el trabajo colaborativo. Es necesario asimismo poder proporcionar a los estudiantes bibliografía que poder consultar, así como la posibilidad de acudir a los profesores en caso necesario.

Teniendo en cuenta lo dicho anteriormente, los ambientes del mundo virtual serían los siguientes:

- Para impartir teoría, se tendrá una sala a través de la cual se podrá acceder a otras cinco diferentes, una por cada tema establecido. Cada una de estas salas, a su vez, estará dividida en diferentes áreas de proyección, con pantallas en las que se podrán ver vídeos cortos explicativos de algún punto de la teoría o imágenes relacionadas con la misma. En dichas salas, además, se podrá utilizar la ayuda de avatares controlados por el programa, que a través de la información que recogerán de la ontología, podrán establecer relaciones y dar anotaciones sobre los términos y conceptos de la materia.

- Para la realización de ejercicios individuales y de autoevaluación, se tendrá una sala con ordenadores. En ellas, los alumnos podrán acceder a test y problemas, con cuya resolución obtendrán una puntuación, aportándoles así el reto de superarse con cada nuevo ejercicio según avanzan en nivel.

- Para la realización de prácticas de proyectos, evaluaciones grupales y trabajos interactivos, se contará con una sala con ordenadores, donde los alumnos acudirán a unas horas determinadas. Una vez dentro, cada uno se adjudicará un role y resolverán las prácticas propuestas, pudiendo contar con la ayuda de avatares en esa misma sala. Los alumnos podrán navegar por todo el mundo virtual para consultar teoría o bibliografía, aunque siempre deberán acabar la práctica dentro de esta aula.

- Para la realización de trabajos en grupo se tendrá una sala con aspecto de cafetería. En la barra los alumnos podrán elegir los problemas, pudiendo éstos ser de un determinado tema o generales. Los estudiantes podrán resolver los ejercicios escogidos sentados en las mesas de las cafeterías con sus compañeros de grupo, manteniendo conversaciones textuales o por voz.

- Los alumnos podrán tener acceso a los profesores para tutorías en una sala similar a un jardín, con bancos, mesas para poder situarse un grupo de personas, etc. Los estudiantes podrán acercarse a los profesores, ya sea individualmente o en grupo, y les expondrán sus dudas por mensajería instantánea o por chat de voz. Esta sala

permanecerá abierta durante el horario destinado a estas tutorías, cuando los profesores accederán al mundo virtual para cumplir su labor.

- Los alumnos dispondrán de una herramienta que les permita utilizar un cuaderno de notas, en el que apuntar aquello que crean relevante sobre la materia impartida.

- Se tendrá una sala disponible para la consulta de material didáctico con forma de biblioteca. Aquí, los estudiantes podrán consultar documentos proporcionados por los profesores, acceder a listas de enlaces externos, etc.

- Una sala representando a una clase convencional será la que recoja a los alumnos en aquellos momentos en los que tengan que realizarse exámenes finales. Esta sala permanecerá abierta durante todo el día escogido para la evaluación, dando la posibilidad de presentarse a todos los alumnos independientemente del horario en el que éstos puedan acceder.

- Se tendrá una sala común con tres tablones de anuncios: el primero contendrá un calendario en el que se podrán consultar las fechas importantes, como entrega de ejercicios, etc.; el segundo actuará como foro de preguntas entre los alumnos, un profesor lo supervisará para responder o corregir en caso de ser necesario. El último tablón mostrará la información relativa al entorno: un mapa del sitio con las distintas salas y su función, y el nombre de los profesores, su dirección de correo electrónico y su horario de tutorías. En esta sala, además, se encontrará un buzón de correos, a través del cual los alumnos podrán "depositar" documentos o mensajes asociándolos a un profesor como destinatario.

Como se puede comprobar, la función del profesor es sobre todo de apoyo, ya sea individualmente o por grupos. Por tanto, el número de alumnos por profesor no debe ser excesivamente pequeño, pues no es necesario que sea un trato tan individual, ni muy grande, para que puedan atender a todos los que tengan dudas sin que tengan que padecer grandes esperas. Se ha establecido, por tanto, que un buen número de alumnos por profesor sea diez.

Los avatares, controlados por el propio programa, son esenciales en la parte de teoría, pues son los que la explican y permiten navegar al alumno a través del conocimiento recogido en la ontología. Se necesitaría tener uno por cada pantalla de proyección, pues cada una estará centrada en un área de los cinco temas propuestos para la materia.

Además de en estos casos, se deberán tener otros tres en la cafetería para poder proporcionar a los alumnos los problemas que ellos deseen. Con este número se asegura a los usuarios que no tendrán que esperar demasiado tiempo en caso de encontrarse con más alumnos en la misma área. Igualmente, en las salas de ejercicios, tanto individuales como grupales, se contará con la presencia de dos avatares en cada una para dar apoyo a los estudiantes, así como en la sala común, en la sala de consulta y en el aula de exámenes.

Los alumnos podrán entrar al mundo virtual en cualquier zona, a excepción de la zona de tutorías y la zona de exámenes, que tan sólo permanecerán abiertas durante el horario de los profesores y cuando se realice un examen, respectivamente. Todos los demás ambientes estarán disponibles las veinticuatro horas al día.

La siguiente tabla muestra un resumen de lo explicado:

SALA	AMBIENTE	FUNCIÓN	AVATARES
Sala común principal	Jardín.	Motivación mediante calendario y foros. Información. Comunicación con los profesores. Acceso a los demás recintos del mundo.	Dos avatares como apoyo a los estudiantes.
Sala de teoría	Área de vídeo e imágenes.	Explicación de la materia de manera visual y textual.	Un avatar por cada área de la materia para transmitir la teoría a los alumnos.
Sala de ejercicios individuales	Sala de ordenadores.	Autoevaluación, motivación para la superación personal en los ejercicios. Problemas propuestos con dificultad gradual.	Dos avatares como apoyo a los estudiantes.
Sala de ejercicios grupales	Sala de ordenadores.	Trabajo colaborativo en grupo con asunción de roles, realización de prácticas y proyectos.	Dos avatares como apoyo a los estudiantes.
Sala de trabajo grupal	Cafetería.	Realización de ejercicios en grupo democrático.	Tres avatares para proponer los ejercicios a los estudiantes.
Sala de tutorías	Jardín.	Realización de tutorías con los profesores, individuales y grupales.	Sin avatares.
Sala de consulta	Biblioteca.	Motivación mediante la consulta de material didáctico, documentos propuestos y enlaces externos.	Dos avatares como apoyo a los estudiantes.
Sala de exámenes	Clase convencional.	Evaluación de conocimientos.	Un avatar.

Tabla 37: Resumen de salas del mundo virtual

6. CASOS DE USO

Los casos de uso definen la interacción entre los agentes externos al sistema, que pueden ser usuarios o incluso otros sistemas, y que se denominan actores, y el propio sistema. Cada caso de uso se ajusta a una funcionalidad ofrecida por la aplicación, y en él pueden participar varios actores, tanto por separado como de manera colaborativa para que dicho caso de uso pueda llevarse a cabo. Asimismo, un actor puede participar en varios casos de uso.

Estos diagramas se utilizaron en un principio para realizar la descripción de los requisitos del problema planteado, aunque en los últimos años su uso ha derivado a su especificación para la propia captura de dichos requisitos.

En este documento, sin embargo, se van a utilizar como medio para aproximar cuál sería el coste de producción del presente software atendiendo a su funcionalidad. Para ello se tomará en cuenta una novedosa métrica que tiene en cuenta precisamente estos diagramas.

6.1. Descripción de los Casos de Uso

En el apartado anterior se han descrito los diferentes entornos con los que contará el mundo virtual para realizar su labor docente. Para detallar los Casos de Uso obtenidos según la funcionalidad que se va a ofrecer y que el lector pueda entenderlos mejor, se ha decidido separar dichos Casos de Uso atendiendo a la sala del mundo virtual en la que estarán disponibles.

- Comunes a todas las salas.

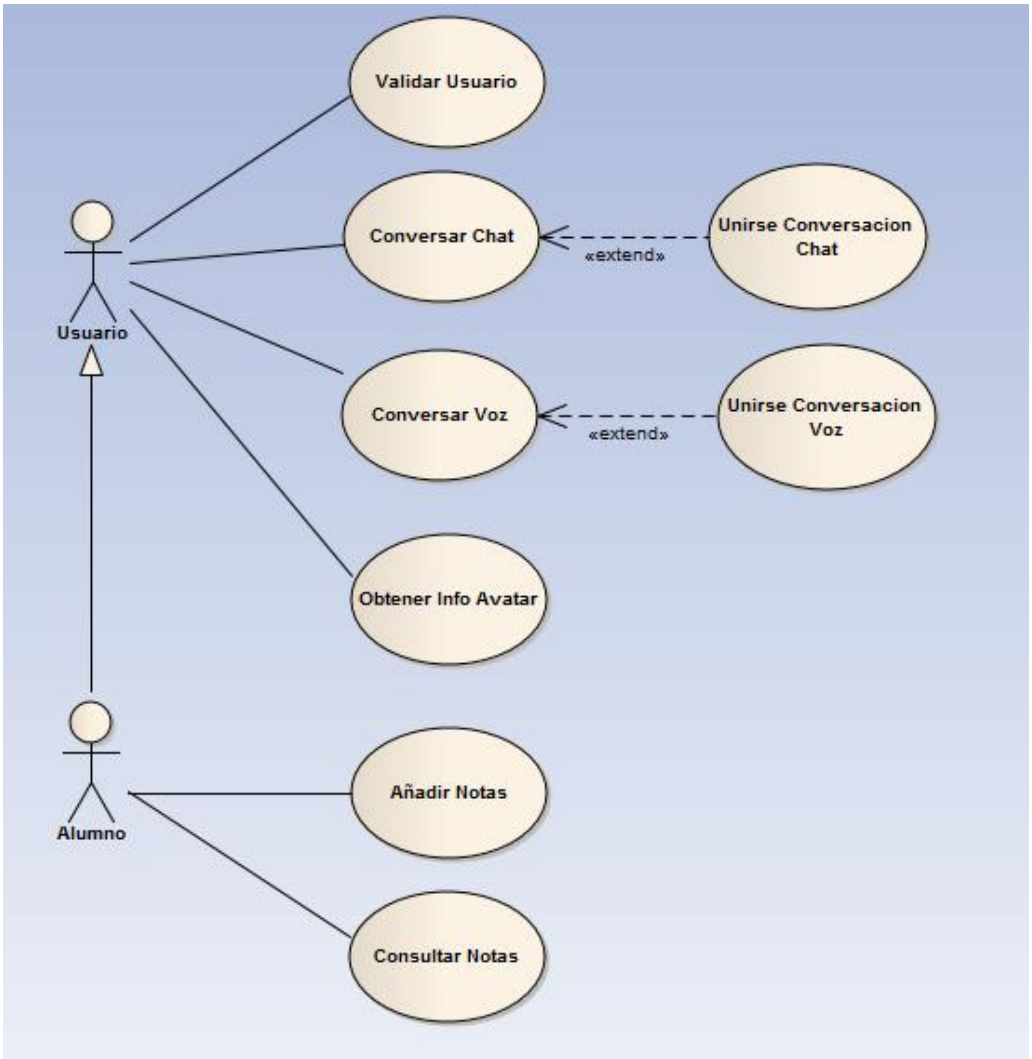

Imagen 55: CU Comunes

NOMBRE	Validar Usuario
ACTORES	Usuario.
DESCRIPCIÓN	Permite a los usuarios identificarse en la aplicación, ya sean alumnos o profesores.
PRECONDICIONES	
POSTCONDICIONES	➢ Usuario validado.
ESCENARIO BÁSICO	1. Introducir la URL de la aplicación. 2. Introducir nombre de usuario. 3. Introducir contraseña. 4. Pulsar el botón Aceptar.

Tabla 38: CU Validar Usuario

NOMBRE	Conversar Chat
ACTORES	Usuario.
DESCRIPCIÓN	Permite a dos usuarios empezar y mantener una conversación textual a través de chat.
PRECONDICIONES	➢ Usuario validado en el sistema.
POSTCONDICIONES	➢ Se abre una ventana en la que se puede mantener una conversación textual a través de chat.
ESCENARIO BÁSICO	1. Acercarse a la persona con la que se desea entablar la conversación. 2. Pulsar sobre la persona deseada. 3. Elegir entablar conversación textual.

Tabla 39: CU Conversar Chat

NOMBRE	Unirse Conversación Chat
ACTORES	Usuario.
DESCRIPCIÓN	Permite a los usuarios unirse a una conversación textual a través de chat abierta por otros usuarios.
PRECONDICIONES	➢ Usuario validado en el sistema. ➢ La persona o personas con las que se desea entablar conversación deben estar manteniendo previamente una conversación textual.
POSTCONDICIONES	➢ Se abre una ventana en la que se puede mantener una conversación textual a través de chat con la persona seleccionada y aquellas con las que mantenía dicha conversación previamente.
ESCENARIO BÁSICO	1. Acercarse a la persona con la que se desea entablar la conversación. 2. Pulsar sobre la persona deseada. 3. Elegir unirse a conversación textual.

Tabla 40: CU Unirse Conversación Chat

NOMBRE	Conversar Voz
ACTORES	Usuario.
DESCRIPCIÓN	Permite a dos usuarios empezar y mantener una conversación por voz utilizando auriculares y micrófono.
PRECONDICIONES	➢ Usuario validado en el sistema.
POSTCONDICIONES	➢ Se abre una ventana en la que se puede mantener una conversación por voz.
ESCENARIO BÁSICO	1. Acercarse a la persona con la que se desea entablar la conversación. 2. Pulsar sobre la persona deseada. 3. Elegir entablar conversación por voz.

Tabla 41: CU Conversar Voz

NOMBRE	Unirse Conversación Voz
ACTORES	Usuario.
DESCRIPCIÓN	Permite a los usuarios unirse a una conversación por voz abierta por otros usuarios.
PRECONDICIONES	➢ Usuario validado en el sistema. ➢ La persona o personas con las que se desea entablar conversación deben estar manteniendo previamente una conversación por voz.
POSTCONDICIONES	➢ Se abre una ventana en la que se puede mantener una conversación por voz con la persona seleccionada y aquellas con las que mantenía dicha conversación previamente.
ESCENARIO BÁSICO	1. Acercarse a la persona con la que se desea entablar la conversación. 2. Pulsar sobre la persona deseada. 3. Elegir unirse a conversación por voz.

Tabla 42: CU Unirse Conversación Voz

NOMBRE	Obtener Info Avatar
ACTORES	Usuario.
DESCRIPCIÓN	El avatar controlado por el sistema le ofrece al usuario información sobre la sala en la que se encuentre y las actividades que en ella pueden realizarse. En la sala de teoría no está disponible este caso de uso.
PRECONDICIONES	➢ Usuario validado en el sistema.
POSTCONDICIONES	➢ Se muestra una ventana con la información sobre la sala y las actividades que en ella pueden realizarse.
ESCENARIO BÁSICO	1. Acercarse a uno de los avatares controlados por la aplicación que se encuentran en la misma sala en la que está el usuario. 2. Pulsar sobre el avatar.

Tabla 43: CU Obtener Info Avatar

NOMBRE	Añadir Notas
ACTORES	Alumno.
DESCRIPCIÓN	Abre una nueva ventana en la que el alumno añade un comentario o nota a su cuaderno de notas personal.
PRECONDICIONES	➢ Usuario validado en el sistema.
POSTCONDICIONES	➢ La nueva nota queda almacenada en el sistema.
ESCENARIO BÁSICO	1. Pulsar sobre el icono de notas en la aplicación. 2. Pulsar sobre el botón Nueva Nota. 3. Introducir textualmente una nueva nota. 4. Pulsar sobre el botón Guardar.

Tabla 44: CU Añadir Notas

NOMBRE	Consultar Notas
ACTORES	Alumno.
DESCRIPCIÓN	Abre una nueva ventana en la que el alumno puede leer los comentarios o notas escritos en su cuaderno de notas personal.
PRECONDICIONES	➢ Usuario validado en el sistema.
POSTCONDICIONES	➢ Se muestra una ventana con las notas del usuario.
ESCENARIO BÁSICO	1. Pulsar sobre el icono de notas en la aplicación. 2. Seleccionar la nota que se desea ver con más detalle.

Tabla 45: CU Consultar Notas

➢ Sala común.

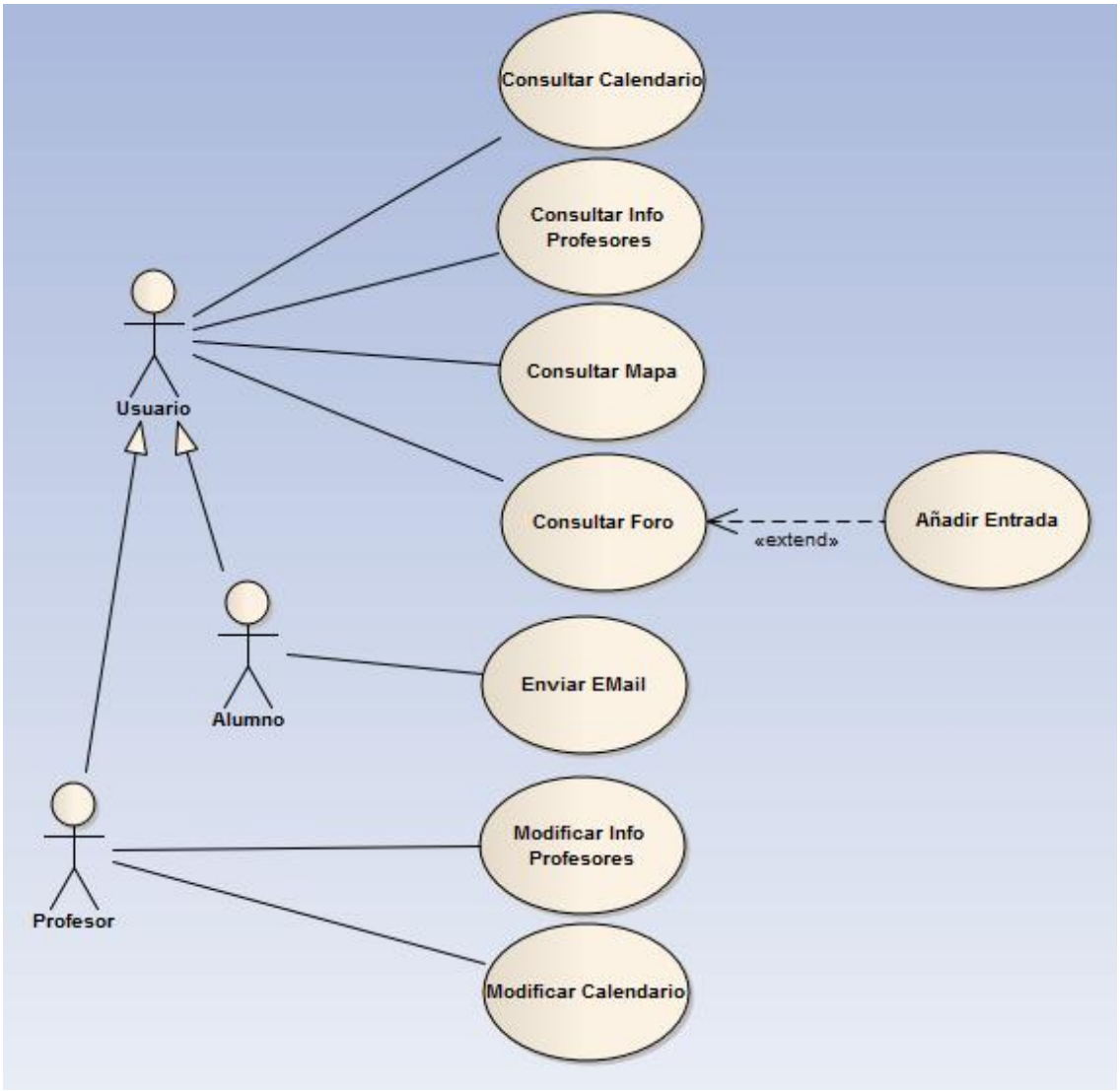

Imagen 56: CU Sala Común

NOMBRE	Consultar Calendario
ACTORES	Usuario.
DESCRIPCIÓN	Abre una nueva ventana en la que se muestra en un calendario los hitos importantes dentro del curso, señalando en el mismo las fechas de éstos.
PRECONDICIONES	➢ Usuario validado en el sistema.
POSTCONDICIONES	➢ Se muestra el calendario con la información sobre los eventos.
ESCENARIO BÁSICO	1. Ir a la sala común. 2. Pulsar sobre el tablón de anuncios que se corresponde con el calendario.

Tabla 46: CU Consultar Calendario

NOMBRE	Consultar Info Profesores
ACTORES	Usuario.
DESCRIPCIÓN	Abre una nueva ventana en la que se muestra la información de los profesores, sus nombres, dirección de correo electrónico, horarios de tutorías, etc.
PRECONDICIONES	➢ Usuario validado en el sistema.
POSTCONDICIONES	➢ Se muestra una ventana con la información almacenada en el sistema sobre los profesores del curso.
ESCENARIO BÁSICO	1. Ir a la sala común. 2. Pulsar sobre el tablón de anuncios que se corresponde con la información sobre el mundo virtual. 3. Elegir la opción de mostrar información sobre profesores.

Tabla 47: CU Consultar Info Profesores

NOMBRE	Consultar Mapa
ACTORES	Usuario.
DESCRIPCIÓN	Abre una nueva ventana en la que se muestra un mapa con la situación de los edificios dentro del mundo virtual y su función.
PRECONDICIONES	➢ Usuario validado en el sistema.
POSTCONDICIONES	➢ Se abre una ventana con el mapa del sitio.
ESCENARIO BÁSICO	1. Ir a la sala común. 2. Pulsar sobre el tablón de anuncios que se corresponde con la información sobre el mundo virtual. 3. Elegir la opción de mostrar mapa del mundo virtual.

Tabla 48: CU Consultar Mapa

NOMBRE	Consultar Foro
ACTORES	Usuario.
DESCRIPCIÓN	Abre una nueva ventana en la que se muestra el foro del mundo virtual, permitiendo navegar entre las diferentes discusiones entabladas en el mismo.
PRECONDICIONES	➢ Usuario validado en el sistema.
POSTCONDICIONES	➢ Se abre una ventana en la que aparecen las discusiones del foro.
ESCENARIO BÁSICO	1. Ir a la sala común. 2. Pulsar sobre el tablón de anuncios que se corresponde con el foro. 3. Navegar entre las discusiones existentes.

Tabla 49: CU Consultar Foro

NOMBRE	Añadir Entrada
ACTORES	Usuario.
DESCRIPCIÓN	Añade una entrada escrita por el usuario dentro de la discusión del foro en la que se encuentra.
PRECONDICIONES	➢ Usuario validado en el sistema. ➢ El usuario debe tener abierto el foro en una discusión determinada.
POSTCONDICIONES	➢ Entrada añadida al sistema.
ESCENARIO BÁSICO	1. Ir a la sala común. 2. Pulsar sobre el tablón de anuncios que se corresponde con el foro. 3. Navegar entre las discusiones existentes y seleccionar la deseada. 4. Escribir la nueva entrada en el formulario adecuado a tal fin. 5. Pulsar sobre el botón Aceptar.

Tabla 50: CU Añadir Entrada

NOMBRE	Enviar EMail
ACTORES	Alumno.
DESCRIPCIÓN	Abre una ventana para que el alumno pueda enviar un e-mail a un profesor, adjuntando los archivos necesarios.
PRECONDICIONES	➢ Usuario validado en el sistema.
POSTCONDICIONES	➢ El correo electrónico es enviado.
ESCENARIO BÁSICO	1. Ir a la sala común. 2. Pulsar sobre el buzón de correos de dicha sala. 3. Elegir entre la lista de profesores el destinatario del correo electrónico. 4. Especificar el asunto del mensaje de correo electrónico. 5. Escribir el mensaje deseado. 6. Pulsar sobre el botón Enviar.
ESCENARIO ALTERNATIVO	5a. Adjuntar los documentos deseados.

Tabla 51: CU Enviar EMail

NOMBRE	Modificar Info Profesores
ACTORES	Profesor.
DESCRIPCIÓN	Un profesor modifica sus propios datos para mostrárselos al resto de usuarios.
PRECONDICIONES	➢ Usuario validado en el sistema.
POSTCONDICIONES	➢ La información sobre el profesor queda modificada.
ESCENARIO BÁSICO	1. Ir a la sala común. 2. Pulsar sobre el tablón de anuncios que se corresponde con la información sobre el mundo virtual. 3. Elegir la opción de mostrar información sobre profesores. 4. Pulsar sobre el botón Modificar. 5. Cambiar los datos deseados. 6. Pulsar sobre el botón Guardar.

Tabla 52: CU Modificar Info Profesores

NOMBRE	Modificar Calendario
ACTORES	Profesor.
DESCRIPCIÓN	Añade, modifica o elimina un evento del calendario.
PRECONDICIONES	➢ Usuario validado en el sistema.
POSTCONDICIONES	➢ El calendario queda modificado.
ESCENARIO BÁSICO	1. Ir a la sala común. 2. Pulsar sobre el tablón de anuncios que se corresponde con el calendario. 3. Pulsar sobre el botón Modificar. 4. En la nueva ventana elegir la acción deseada: añadir, modificar o eliminar. 5. Pulsar sobre el botón Guardar.
ESCENARIO ALTERNATIVO	4a. Tras pulsar el botón Añadir, rellenar los nuevos campos en relación a la fecha y evento. 4b. Tras pulsar el botón Modificar, cambiar aquellos campos que se desea corregir. 4c. Tras pulsar el botón Eliminar, seleccionar el evento que se desea borrar.

Tabla 53: CU Modificar Calendario

- Sala de teoría.

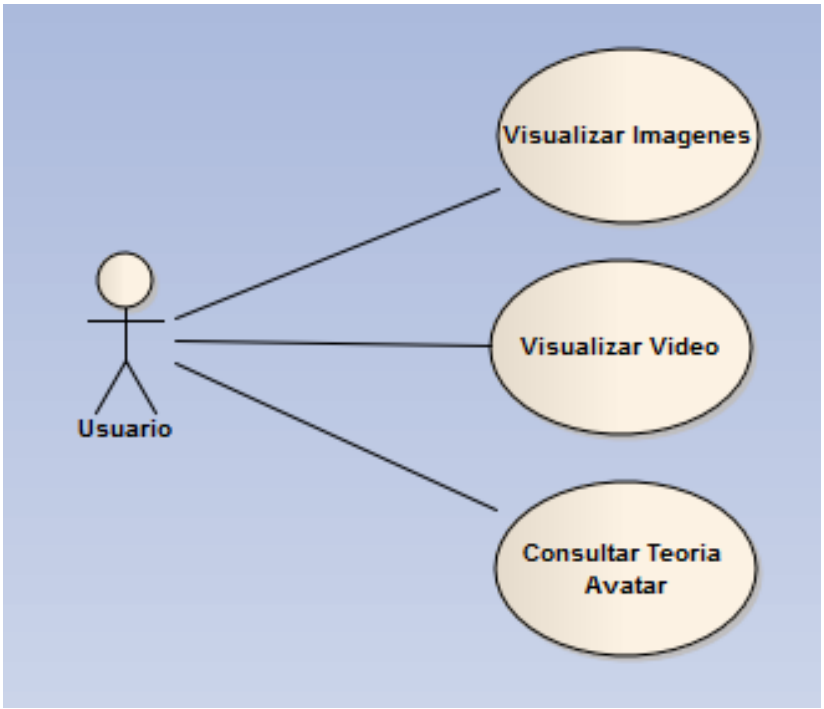

Imagen 57: CU Sala Teoría

NOMBRE	Visualizar Imágenes
ACTORES	Usuario.
DESCRIPCIÓN	Muestra en la pantalla seleccionada la imagen o sucesión de imágenes que ésta almacena.
PRECONDICIONES	➢ Usuario validado en el sistema.
POSTCONDICIONES	➢ Se muestran las imágenes asociadas a la pantalla seleccionada.
ESCENARIO BÁSICO	1. Ir a la sala de teoría. 2. Entrar en la sala correspondiente con el tema que se desee estudiar. 3. Entrar en el área de la sala que contenga la teoría sobre la que se desea estudiar. 4. Situarse en frente de la pantalla deseada. 5. Pulsar sobre la pantalla.

Tabla 54: CU Visualizar Imágenes

NOMBRE	Visualizar Vídeo
ACTORES	Usuario.
DESCRIPCIÓN	Muestra en la pantalla seleccionada el vídeo que ésta almacena.
PRECONDICIONES	➢ Usuario validado en el sistema.
POSTCONDICIONES	➢ Muestra el vídeo asociado a la pantalla seleccionada.
ESCENARIO BÁSICO	1. Ir a la sala de teoría. 2. Entrar en la sala correspondiente con el tema que se desee estudiar. 3. Entrar en el área de la sala que contenga la teoría sobre la que se desea estudiar. 4. Situarse en frente de la pantalla deseada. 5. Pulsar sobre la pantalla.

Tabla 55: CU Visualizar Vídeo

NOMBRE	Consultar Teoría Avatar
ACTORES	Usuario.
DESCRIPCIÓN	El avatar seleccionado permite al usuario navegar a través de la teoría mediante sinónimos, definiciones, conceptos relacionados, etc.
PRECONDICIONES	➢ Usuario validado en el sistema.
POSTCONDICIONES	➢ El avatar muestra la teoría seleccionada.
ESCENARIO BÁSICO	1. Ir a la sala de teoría. 2. Entrar en la sala correspondiente con el tema que se desee estudiar. 3. Entrar en el área de la sala que contenga la teoría sobre la que se desea estudiar. 4. Situarse en frente del avatar deseado. 5. Pulsar sobre el avatar. 6. Navegar a través de la información que éste ofrece.

Tabla 56: CU Consultar Teoría Avatar

- Sala de ejercicios individuales.

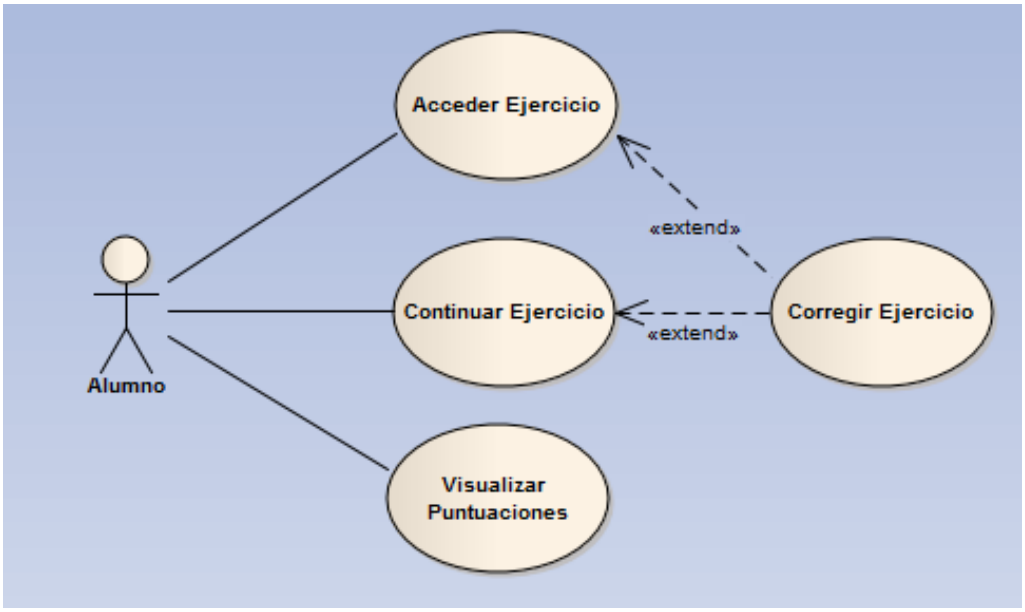

Imagen 58: CU Sala Ejercicios Individuales

NOMBRE	Acceder Ejercicio
ACTORES	Alumno.
DESCRIPCIÓN	Abre una nueva ventana en la que se muestra el ejercicio que el alumno desea y en la que puede realizarlo.
PRECONDICIONES	➢ Usuario validado en el sistema.
POSTCONDICIONES	➢ Se muestra el ejercicio y se guarda el avance del alumno en el sistema.
ESCENARIO BÁSICO	1. Ir a la sala de ejercicios individuales. 2. Acercarse a una mesa con un ordenador. 3. Pulsar sobre el ordenador. 4. Elegir en la ventana emergente Nuevo Ejercicio. 5. Elegir el tema sobre el que se desea practicar. 6. Realizar el ejercicio en la ventana abierta con el mismo. 7. Pulsar el botón Guardar.
ESCENARIO ALTERNATIVO	6a. Pulsar el botón Corregir.

Tabla 57: CU Acceder Ejercicio

NOMBRE	Continuar Ejercicio
ACTORES	Alumno.
DESCRIPCIÓN	Abre una nueva ventana en la que se muestra el ejercicio que el alumno desea y en la que puede realizarlo.
PRECONDICIONES	➢ Usuario validado en el sistema. ➢ Tener un ejercicio sin terminar en el sistema.
POSTCONDICIONES	➢ Se abre el ejercicio que el alumno dejó sin terminar y se guarda de nuevo su avance en el mismo.
ESCENARIO BÁSICO	1. Ir a la sala de ejercicios individuales. 2. Acercarse a una mesa con un ordenador. 3. Pulsar sobre el ordenador. 4. Elegir en la ventana emergente Ejercicios Comenzados. 5. Elegir el ejercicio que se desea continuar. 6. Continuar el ejercicio en la ventana abierta con el mismo. 7. Pulsar el botón Guardar.
ESCENARIO ALTERNATIVO	6a. Pulsar el botón Corregir.

Tabla 58: CU Continuar Ejercicio

NOMBRE	Corregir Ejercicio
ACTORES	Alumno.
DESCRIPCIÓN	Abre una nueva ventana en la que se muestra el ejercicio que el alumno desea y en la que puede realizarlo.
PRECONDICIONES	➢ Usuario validado en el sistema. ➢ Tener un ejercicio sin terminar en el sistema.
POSTCONDICIONES	➢ El ejercicio queda corregido y se muestra su puntuación.
ESCENARIO BÁSICO	1. Ir a la sala de ejercicios individuales. 2. Acercarse a una mesa con un ordenador. 3. Pulsar sobre el ordenador. 4. Elegir en la ventana emergente Ejercicios Comenzados. 5. Elegir el ejercicio que se desea continuar. 6. Continuar el ejercicio en la ventana abierta con el mismo. 7. Pulsar el botón Corregir.

Tabla 59: CU Corregir Ejercicio

NOMBRE	Visualizar puntuaciones
ACTORES	Alumno.
DESCRIPCIÓN	Abre una nueva ventana en la que se muestran los ejercicios realizados, su dificultad y la puntuación obtenida en cada uno de ellos.
PRECONDICIONES	➢ Usuario validado en el sistema.
POSTCONDICIONES	➢ Se muestran las puntuaciones que el alumno ha obtenido en los diferentes ejercicios realizados.
ESCENARIO BÁSICO	1. Ir a la sala de ejercicios individuales. 2. Acercarse a una mesa con un ordenador. 3. Pulsar sobre el ordenador. 4. Elegir en la ventana emergente Puntuaciones.

Tabla 60: CU Visualizar puntuaciones

➢ Sala de ejercicios grupales.

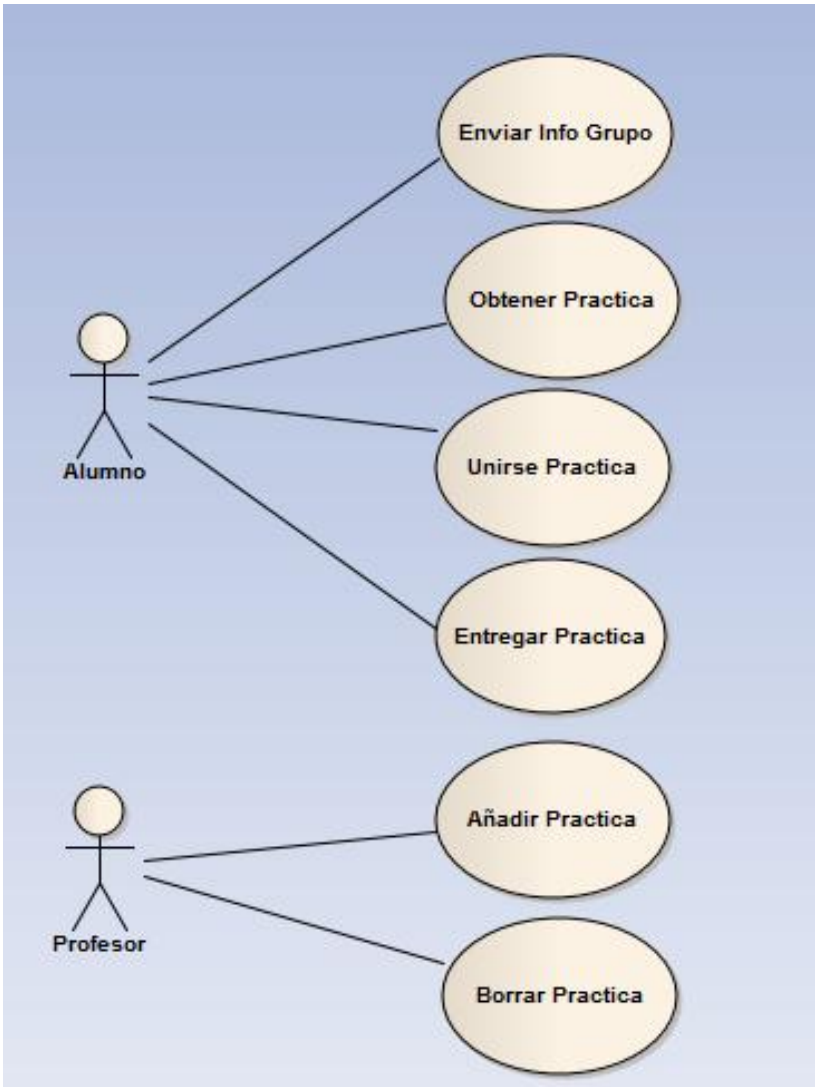

Imagen 59: CU Sala Ejercicios Grupales

NOMBRE	Enviar Info Grupo
ACTORES	Alumno.
DESCRIPCIÓN	Envía al sistema y a los profesores los nombres del grupo formado por los alumnos y su role dentro del mismo.
PRECONDICIONES	➢ Usuario validado en el sistema.
POSTCONDICIONES	➢ La información sobre el grupo es enviada al sistema.
ESCENARIO BÁSICO	1. Ir a la sala de ejercicios grupales. 2. Acercarse a una mesa. 3. Pulsar sobre el ordenador en la mesa. 4. Pulsar en la ventana emergente Nuevo Grupo. 5. Añadir los nombres de los componentes del grupo y su role, así como el nombre que desean dar al equipo. 6. Pulsar el botón Enviar.

Tabla 61: CU Enviar Info Grupo

NOMBRE	Obtener Práctica
ACTORES	Alumno.
DESCRIPCIÓN	Muestra a los usuarios la práctica que están realizando y el documento en el que están escribiendo la solución.
PRECONDICIONES	➢ Usuario validado en el sistema. ➢ Tener un grupo formado, habiendo enviado previamente la información del mismo al sistema.
POSTCONDICIONES	➢ La práctica perteneciente al grupo seleccionado se muestra en la pantalla.
ESCENARIO BÁSICO	1. Ir a la sala de ejercicios grupales. 2. Acercarse a una mesa. 3. Pulsar sobre el ordenador en la mesa. 4. Pulsar en la ventana emergente Seleccionar Grupo. 5. Seleccionar en la lista el nombre del grupo al que pertenece. 6. Seleccionar comunicación por chat. 7. Pulsar el botón Aceptar para que se abra las ventanas con el enunciado de la práctica y el documento en el que se está escribiendo la solución.
ESCENARIO ALTERNATIVO	6a. Seleccionar comunicación por voz.

Tabla 62: CU Obtener Práctica

NOMBRE	Unirse Práctica
ACTORES	Alumno.
DESCRIPCIÓN	Muestra al alumno la práctica que está realizando junto a su grupo y el documento en el que están escribiendo la solución.
PRECONDICIONES	➢ Usuario validado en el sistema. ➢ Tener un grupo formado, habiendo enviado previamente la información del mismo al sistema. ➢ Al menos un miembro del grupo del alumno debe tener abierta la práctica.
POSTCONDICIONES	➢ La práctica perteneciente al grupo seleccionado se muestra en la pantalla. ➢ Se establece la comunicación con los compañeros que ya estén conectados con la práctica.
ESCENARIO BÁSICO	1. Ir a la sala de ejercicios grupales. 2. Acercarse a una mesa. 3. Pulsar sobre el ordenador en la mesa. 4. Pulsar en la ventana emergente Seleccionar Grupo. 5. Seleccionar en la lista el nombre del grupo al que pertenece. 6. Seleccionar unirse a la conversación. 7. Pulsar el botón Aceptar para que se abra las ventanas con el enunciado de la práctica y el documento en el que se está escribiendo la solución

Tabla 63: CU Unirse Práctica

NOMBRE	Entregar Práctica
ACTORES	Alumno.
DESCRIPCIÓN	Entrega la práctica enviándosela por correo electrónico a los profesores.
PRECONDICIONES	➢ Usuario validado en el sistema. ➢ Tener un grupo formado, habiendo enviado previamente la información del mismo al sistema. ➢ Tener como role dentro del grupo “Jefe de Proyecto”.
POSTCONDICIONES	➢ La práctica perteneciente al grupo seleccionado se envía por correo electrónico a los profesores.
ESCENARIO BÁSICO	1. Ir a la sala de ejercicios grupales. 2. Acercarse a una mesa. 3. Pulsar sobre el ordenador en la mesa. 4. Pulsar en la ventana emergente Seleccionar Grupo. 5. Seleccionar en la lista el nombre del grupo al que pertenece. 6. Pulsar sobre el botón Enviar Práctica.

Tabla 64: CU Entregar Práctica

NOMBRE	Añadir Práctica
ACTORES	Profesor.
DESCRIPCIÓN	Añade el enunciado de una práctica obligatoria del curso al sistema.
PRECONDICIONES	➢ Usuario validado en el sistema.
POSTCONDICIONES	➢ La práctica queda añadida en el sistema.
ESCENARIO BÁSICO	1. Ir a la sala de ejercicios grupales. 2. Acercarse a una mesa. 3. Pulsar sobre el ordenador en la mesa. 4. Seleccionar el botón Nueva Práctica. 5. Subir el documento con el enunciado de la práctica. 6. Especificar fecha de entrega de la práctica. 7. Pulsar el botón Guardar.

Tabla 65: CU Añadir Práctica

NOMBRE	Borrar Práctica
ACTORES	Profesor.
DESCRIPCIÓN	Añade el enunciado de una práctica obligatoria del curso al sistema.
PRECONDICIONES	➢ Usuario validado en el sistema.
POSTCONDICIONES	➢ La práctica queda eliminada del sistema.
ESCENARIO BÁSICO	1. Ir a la sala de ejercicios grupales. 2. Acercarse a una mesa. 3. Pulsar sobre el ordenador en la mesa. 4. Seleccionar el botón Eliminar Práctica. 5. Seleccionar la práctica que se desea eliminar. 6. Pulsar el botón Aceptar.

Tabla 66: CU Borrar Práctica

- Sala de trabajo grupal.

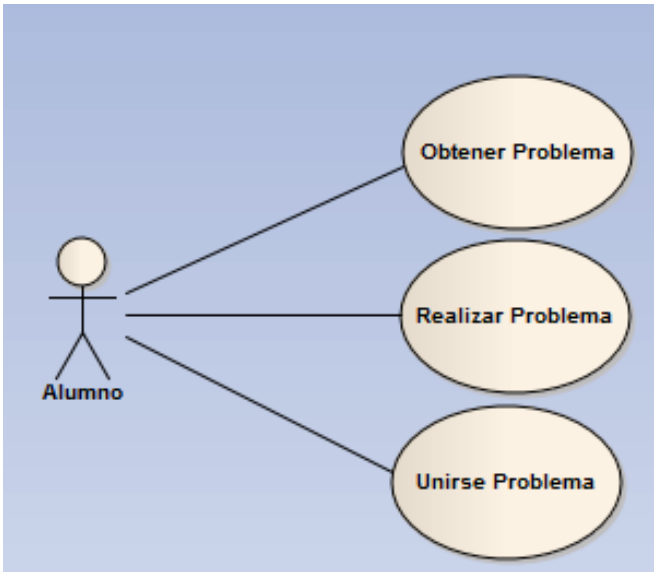

Imagen 60: CU Sala Trabajo Grupal

NOMBRE	Obtener Problema
ACTORES	Alumno.
DESCRIPCIÓN	El problema seleccionado queda asignado al alumno que lo ha escogido.
PRECONDICIONES	➢ Usuario validado en el sistema.
POSTCONDICIONES	➢ El usuario tiene asignado el problema escogido para posteriormente poder realizarlo.
ESCENARIO BÁSICO	1. Ir a la sala de trabajo grupal. 2. Acercarse a la barra. 3. Pulsar sobre uno de los avatares controlados por el sistema que se encuentran detrás de la barra. 4. Seleccionar el tema del cual se quiere obtener el problema. 5. Seleccionar el problema deseado. 6. Pulsar el botón Aceptar.

Tabla 67: CU Obtener Problema

NOMBRE	Realizar Problema
ACTORES	Alumno.
DESCRIPCIÓN	El problema seleccionado queda asignado al alumno que lo ha escogido, y queda abierto para aquellos que quieran unirse al mismo.
PRECONDICIONES	➢ Usuario validado en el sistema. ➢ El usuario debe tener asignado previamente un problema que haya escogido.
POSTCONDICIONES	➢ El problema queda abierto para ser resuelto. ➢ Queda seleccionado el método de comunicación entre el usuario dueño del problema y aquellos compañeros que deseen unirse al mismo.
ESCENARIO BÁSICO	1. Ir a la sala de trabajo grupal. 2. Acercarse a una mesa. 3. Pulsar sobre la mesa. 4. Seleccionar el tipo de conversación que se desea mantener: por chat o por voz.

Tabla 68: CU Realizar Problema

NOMBRE	Unirse Problema
ACTORES	Alumno.
DESCRIPCIÓN	El problema deseado se abre en la pantalla del alumno, así como se inicia la conversación con el resto de compañeros que estén trabajando sobre el mismo problema.
PRECONDICIONES	➢ Usuario validado en el sistema. ➢ El problema al que se desea unirse debe estar abierto por el alumno dueño del mismo.
POSTCONDICIONES	➢ El alumno obtiene una copia del problema que se está solucionando. ➢ El alumno se une a la conversación sobre el problema.
ESCENARIO BÁSICO	1. Ir a la sala de trabajo grupal. 2. Acercarse a la mesa en la que se encuentra el grupo de trabajo al que desea unirse. 3. Pulsar sobre la mesa escogida. 4. Pulsar sobre el botón Unirse.

Tabla 69: CU Unirse Problema

- Sala de tutorías.

En esta sala no se añaden casos de uso nuevos puesto que en ella se tendrán únicamente en cuenta aquellos que son comunes a todas las salas.

- Sala de consulta.

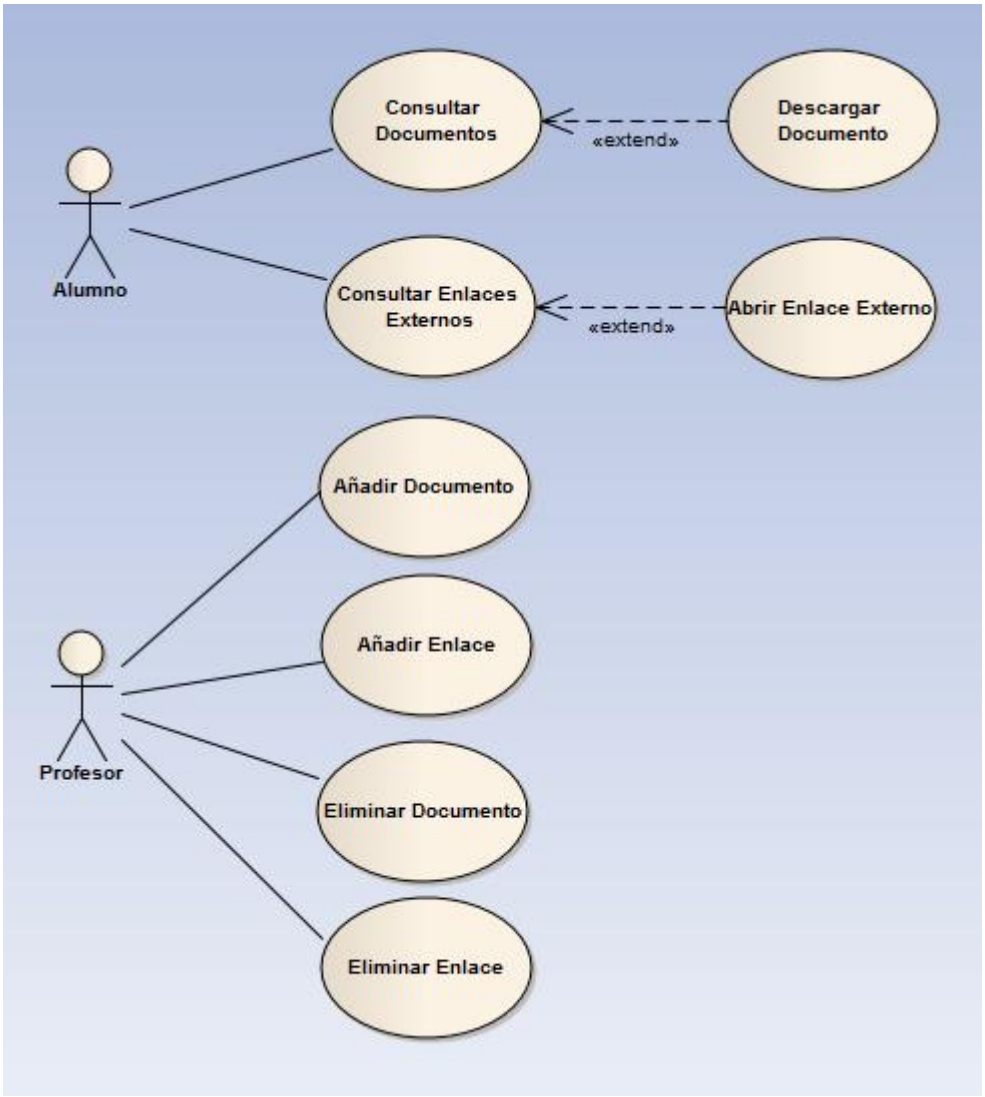

Imagen 61: CU Sala Consulta

NOMBRE	Consultar Documentos
ACTORES	Alumno.
DESCRIPCIÓN	El alumno obtiene un listado con los documentos almacenados en el sistema.
PRECONDICIONES	➢ Usuario validado en el sistema.
POSTCONDICIONES	➢ El alumno obtiene un listado con los documentos disponibles en el sistema.
ESCENARIO BÁSICO	1. Ir a la sala de consulta. 2. Acercarse a una de las estanterías. 3. Pulsar sobre la estantería. 4. Seleccionar el tema sobre el que se quieren ver los documentos.

Tabla 70: CU Consultar Documentos

NOMBRE	Descargar Documento
ACTORES	Alumno.
DESCRIPCIÓN	El documento seleccionado se descarga a la máquina del usuario.
PRECONDICIONES	➢ Usuario validado en el sistema.
POSTCONDICIONES	➢ El documento queda alojado en la máquina del alumno.
ESCENARIO BÁSICO	1. Ir a la sala de consulta. 2. Acercarse a una de las estanterías. 3. Pulsar sobre la estantería. 4. Seleccionar el tema sobre el que se quieren ver los documentos. 5. Seleccionar el documento deseado. 6. Pulsar sobre el botón Descargar. 7. Elegir la ruta en la que se desea que se guarde el documento. 8. Pulsar el botón Aceptar.

Tabla 71: CU Descargar Documento

NOMBRE	Consultar Enlaces Externos
ACTORES	Alumno.
DESCRIPCIÓN	Se muestra una nueva ventana con los enlaces externos concernientes al tema seleccionado.
PRECONDICIONES	➢ Usuario validado en el sistema.
POSTCONDICIONES	➢ El alumno obtiene en una nueva ventana un listado con los enlaces externos disponibles.
ESCENARIO BÁSICO	1. Ir a la sala de consulta. 2. Acercarse a una de las mesas con ordenadores. 3. Pulsar sobre el ordenador. 4. Seleccionar el tema sobre el que se quieren ver los enlaces.

Tabla 72: CU Consultar Enlaces Externos

NOMBRE	Abrir Enlace Externo
ACTORES	Alumno.
DESCRIPCIÓN	El enlace seleccionado se abre en una nueva ventana del navegador por defecto del usuario.
PRECONDICIONES	➢ Usuario validado en el sistema.
POSTCONDICIONES	➢ Queda abierta una nueva ventana del navegador con la página correspondiente al enlace seleccionado.
ESCENARIO BÁSICO	1. Ir a la sala de consulta. 2. Acercarse a una de las mesas con ordenadores. 3. Pulsar sobre el ordenador. 4. Seleccionar el tema sobre el que se quieren ver los enlaces. 5. Seleccionar el enlace deseado. 6. Pulsar el botón Aceptar.

Tabla 73: CU Abrir Enlace Externo

NOMBRE	Añadir Documento
ACTORES	Profesor.
DESCRIPCIÓN	Se añade un nuevo documento a la lista disponible.
PRECONDICIONES	➢ Usuario validado en el sistema.
POSTCONDICIONES	➢ El documento queda añadido en el sistema.
ESCENARIO BÁSICO	1. Ir a la sala de consulta. 2. Acercarse a una de las estanterías. 3. Pulsar sobre la estantería. 4. Pulsar sobre el botón Nuevo Documento. 5. Seleccionar la ruta donde se encuentra el documento que desea añadirse a la aplicación. 6. Seleccionar el tema al que pertenece el documento. 7. Pulsar el botón Aceptar.

Tabla 74: CU Añadir Documento

NOMBRE	Añadir Enlace
ACTORES	Profesor.
DESCRIPCIÓN	Se añade un nuevo enlace a la lista disponible.
PRECONDICIONES	➢ Usuario validado en el sistema.
POSTCONDICIONES	➢ El enlace queda añadido en el sistema.
ESCENARIO BÁSICO	1. Ir a la sala de consulta. 2. Acercarse a una de las mesas con ordenadores. 3. Pulsar sobre el ordenador. 4. Pulsar sobre el botón Nuevo Enlace. 5. Escribir la URL y descripción del enlace. 6. Seleccionar el tema al que pertenece el documento. 7. Pulsar el botón Aceptar.

Tabla 75: CU Añadir Enlace

NOMBRE	Eliminar Documento
ACTORES	Profesor
DESCRIPCIÓN	Se elimina un documento del sistema.
PRECONDICIONES	➢ Usuario validado en el sistema.
POSTCONDICIONES	➢ El documento seleccionado queda eliminado del sistema.
ESCENARIO BÁSICO	1. Ir a la sala de consulta. 2. Acercarse a una de las estanterías. 3. Pulsar sobre la estantería. 4. Pulsar sobre el botón Eliminar Documento. 5. Seleccionar el tema al que pertenece el documento. 6. Seleccionar de la lista mostrada el documento que se desea eliminar. 7. Pulsar el botón Aceptar.

Tabla 76: CU Eliminar Documento

NOMBRE	Eliminar Enlace
ACTORES	Profesor
DESCRIPCIÓN	Se elimina un enlace del sistema.
PRECONDICIONES	➢ Usuario validado en el sistema.
POSTCONDICIONES	➢ El enlace seleccionado queda eliminado del sistema.
ESCENARIO BÁSICO	1. Ir a la sala de consulta. 2. Acercarse a una de las mesas con ordenadores. 3. Pulsar sobre el ordenador. 4. Pulsar sobre el botón Eliminar Enlace. 5. Seleccionar el tema al que pertenece el enlace. 6. Seleccionar de la lista mostrada el enlace que se desea eliminar. 7. Pulsar el botón Aceptar.

Tabla 77: CU Eliminar Enlace

➢ Sala de exámenes.

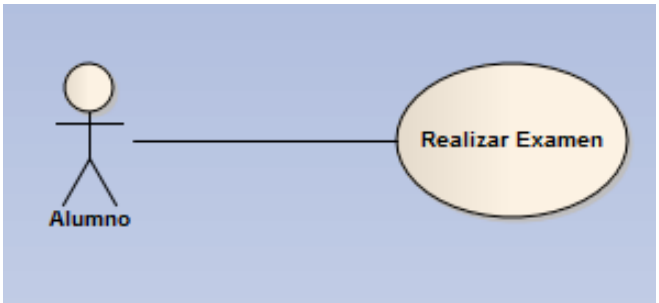

Imagen 62: CU Sala Exámenes

NOMBRE	Realizar Examen
ACTORES	Alumno.
DESCRIPCIÓN	El alumno obtiene una copia de su examen en una ventana en la que se le permite realizarlo.
PRECONDICIONES	➢ Usuario validado en el sistema.
POSTCONDICIONES	➢ El examen queda abierto en la máquina del usuario para que pueda realizarlo.
ESCENARIO BÁSICO	1. Ir a la sala de exámenes. 2. Acercarse a una de las mesas. 3. Pulsar sobre la mesa. 4. Pulsar sobre el botón Comenzar Examen. 5. Cuando se haya terminado de realizar el examen, pulsar sobre el botón Entregar Examen.

Tabla 78: CU Realizar Examen

7. DISEÑO

Una vez tratada la parte más teórica sobre el presente proyecto, este apartado tiene como finalidad adentrarse en los aspectos técnicos del mismo. Aunque el sistema propuesto no vaya a desarrollarse, se ha considerado interesante incluir la descripción del diseño del mismo con el fin de proporcionar una estructura arquitectónica que pueda modelar un problema como el propuesto: un mundo virtual para la enseñanza.

Se presentarán tanto el modelo arquitectónico del sistema como el diseño detallado a un alto nivel.

7.1. Arquitectura del sistema

A continuación se muestra un diagrama de despliegue en el que se detalla la distribución de los componentes del sistema en los diferentes dispositivos hardware de los que se compone, así como la relación existente entre estos dispositivos.

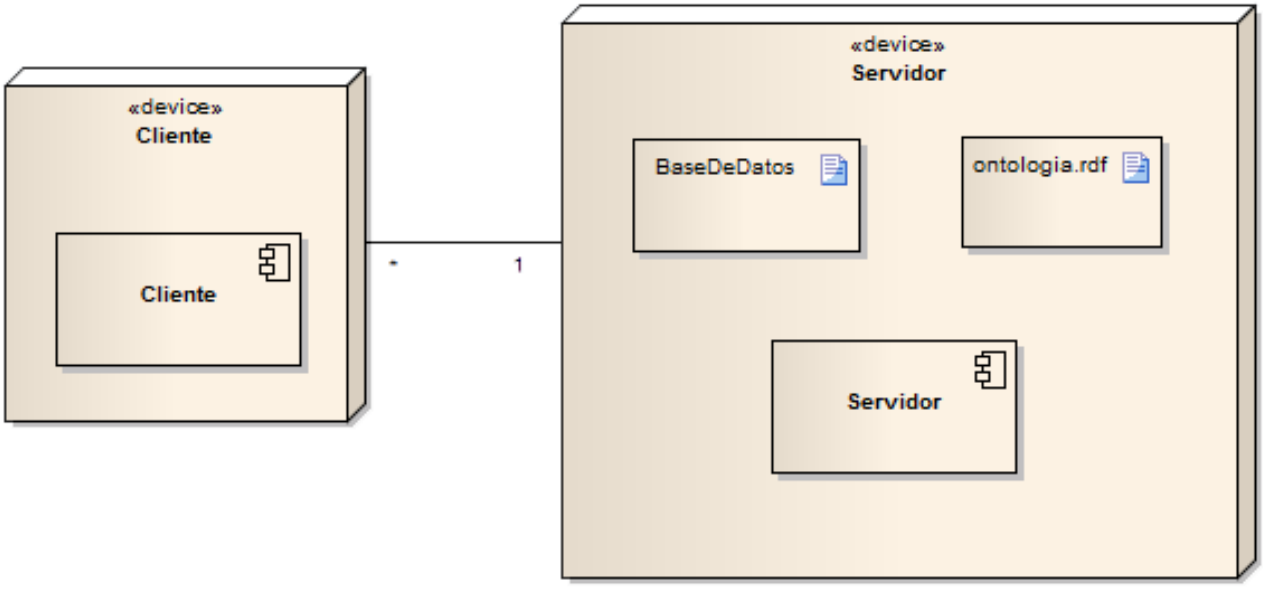

Imagen 63: Diagrama de despliegue

Al tratarse de una aplicación Web, se tienen dos partes bien diferenciadas en cuanto al hardware: el cliente, que será la máquina del usuario, ya sea profesor o alumno, y el servidor, donde estará alojada la aplicación.

En la imagen se pueden observar esas dos partes representadas por sendos nodos. La parte del cliente tendrá el peso del motor gráfico, simbolizado en el diagrama como el componente Cliente. Por otra parte, el servidor será el encargado de ejecutar la aplicación

para que el cliente pueda acceder a la misma, pudiéndose ver de nuevo como el componente Servidor. La relación que se observa entre ambos implica que, tal y como especifica la política de Google App Engine, se tendrá un servidor, aunque el número de máquinas físicas podrá variar teniendo en cuenta las necesidades de los clientes.

Para que la parte del servidor pueda realizar su labor, es necesario que acceda a información externa, donde se encuentran los datos relacionados con el temario del curso y los usuarios, así como la interacción de estos últimos con el sistema. En el diagrama pueden verse tanto la base de datos como la ontología, elementos indispensables para aquello concerniente a la gestión del conocimiento de la enseñanza de la materia a impartir.

Atendiendo al por qué de esta distribución de los componentes aquí expuesta, es necesario recordar que la aplicación ha de ejecutarse en un entorno Web, lo que significa que las comunicaciones que se realicen entre el cliente y el servidor son el punto crítico en el funcionamiento de la misma. Es por esto que se hace primordial disminuir en la medida de lo posible la complejidad de estas comunicaciones. Como medida para solucionar este problema, se ha llegado a la conclusión de que el motor gráfico del entorno virtual se ejecute en el cliente, intercambiando con el servidor exclusivamente la información relativa a la interacción de los usuarios con el entorno, realizando el cliente la tarea de interpretar esta información y mostrársela a los usuarios.

Para que el mundo virtual funcione, y que lo haga para un gran número de clientes simultáneamente, el servidor debe encargarse de gestionar a todos estos y transmitir esta información al motor gráfico mencionado anteriormente. Para ello, llevará la gestión de dichos usuarios, las comunicaciones entre ellos, las descargas y subidas de documentos y otras funcionalidades para que la interacción entre los usuarios y el mundo virtual sea posible. De igual forma, el servidor deberá gestionar la parte de enseñanza de la materia, para lo cual contará con los datos sobre la misma almacenados en la ontología.

7.2. Diseño detallado

A fin de poder mostrar cómo funciona el sistema de forma más detallada, se ha realizado el siguiente diagrama, en el que se presenta, desde un alto nivel, la estructura de componentes del mismo.

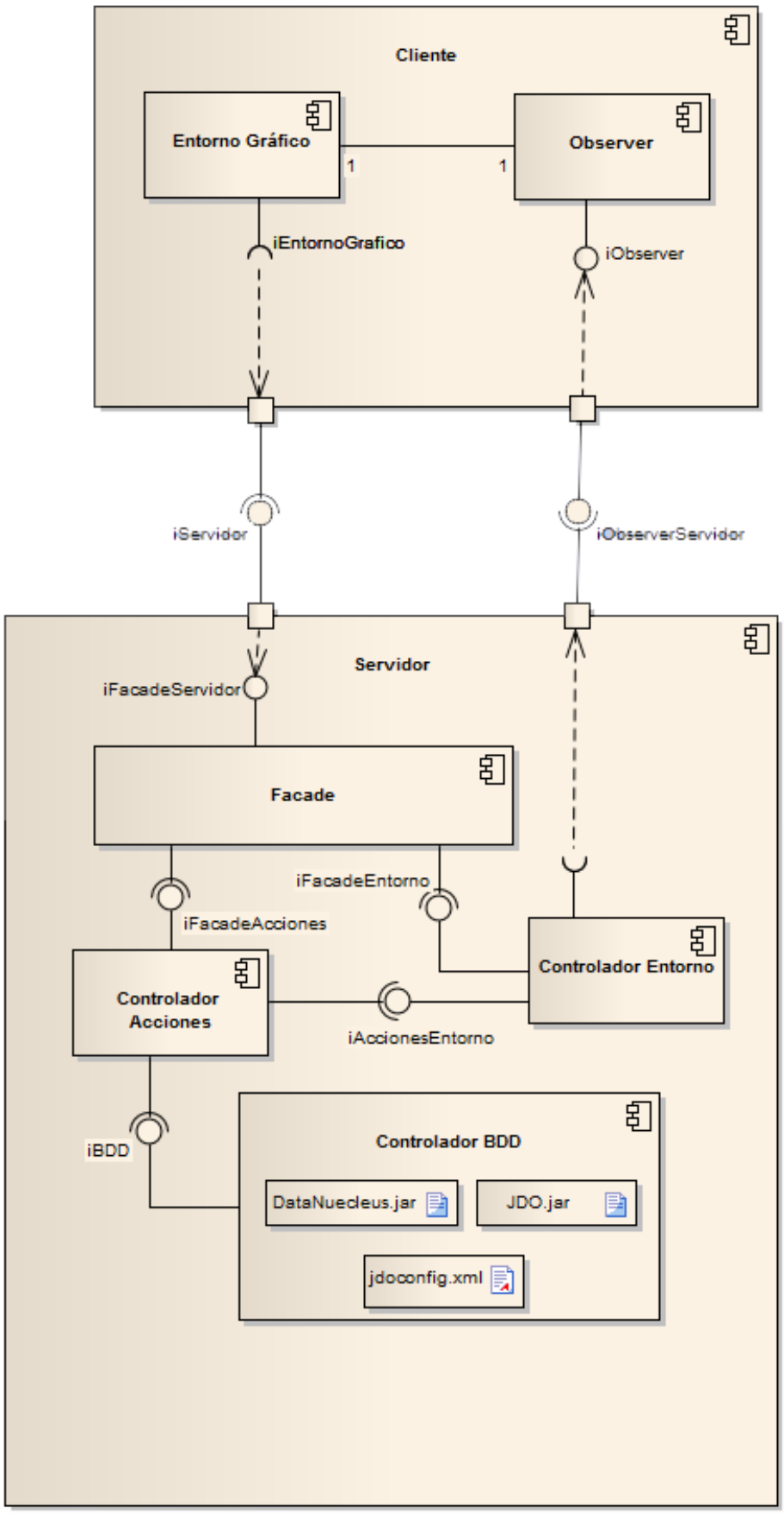

Imagen 64: Diagrama de componentes

Por tratarse, como bien se ha comentado anteriormente, de una aplicación Web, el patrón de diseño arquitectónico escogido ha sido Cliente – Servidor. En esta arquitectura, también denominada dos capas, el cliente se caracteriza por ser un conjunto de estaciones de trabajo con una interfaz unificada que acceden a la capa cliente, usualmente dedicada al almacenamiento y procesamiento de datos centralizados [108].

La mayoría de las páginas en Internet y aplicaciones Web implementan una arquitectura Cliente – Servidor, en las que este último ofrece una serie de interfaces al usuario a través de la red.

Como ventajas a la hora de utilizar esta arquitectura destacan las siguientes:

- El control de la aplicación queda centralizado en el servidor, el cual se concentra en que la integridad y seguridad de la información que almacena no sea violada por un cliente malintencionado o defectuoso.
- Es un sistema altamente escalable que admite tanto el aumento en el número de clientes como en el de servidores, siempre que se necesite aumentar la capacidad computacional de la aplicación.
- El mantenimiento se hace más sencillo, puesto que el servidor puede sufrir modificaciones sin afectar al cliente, siempre que mantenga la misma interfaz de comunicación.

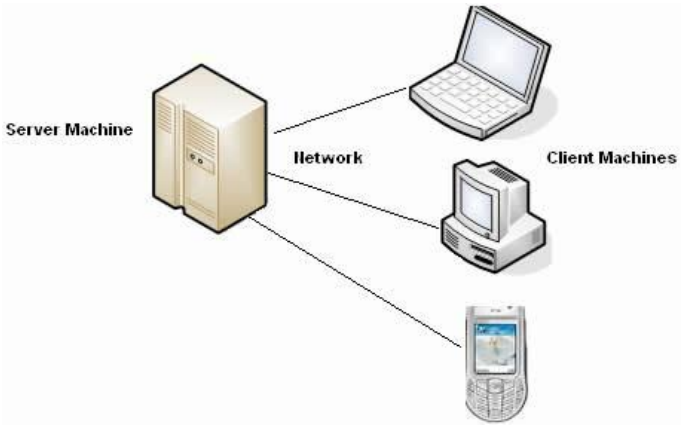

Imagen 65: Arquitectura Cliente – Servidor

Como bien detalla la arquitectura, el sistema está basado en dos componentes, el Cliente, que se corresponde con aquello que será ejecutado en las máquinas de los usuarios, y el Servidor, cuyo contenido estará alojado en los servidores proporcionados por Google App Engine.

En el diseño destacan dos patrones de diseño utilizados: el patrón Observer y el patrón Facade.

El patrón Observer [109] es un patrón de diseño orientado a controlar el comportamiento de una serie de objetos dentro de un sistema. Más concretamente, la función principal de este

patrón es la de notificar a una serie de objetos el cambio en el estado de otro objeto determinado. Para esto se establece una relación 1:N donde el 1 es el objeto cuyo estado resulta interesante para los otros N objetos.

La principal motivación para el uso de este patrón es la separación de la vista y los datos de un sistema, de modo que se puedan realizar actualizaciones de la GUI, de forma que todas ellas estén actualizadas con los cambios sobre los datos de forma síncrona.

En el siguiente diagrama puede apreciarse la estructura básica que debería cumplirse a la hora de realizar la implementación del patrón. Las dos clases principales son Observer y Subject: la primera es la clase observadora que estará atenta a los cambios acontecidos en los objetos provenientes de la clase Subject; la agregación significa que un Subject puede tener varios observadores. Así mismo un Observer podrá estar pendiente de varios Subject.

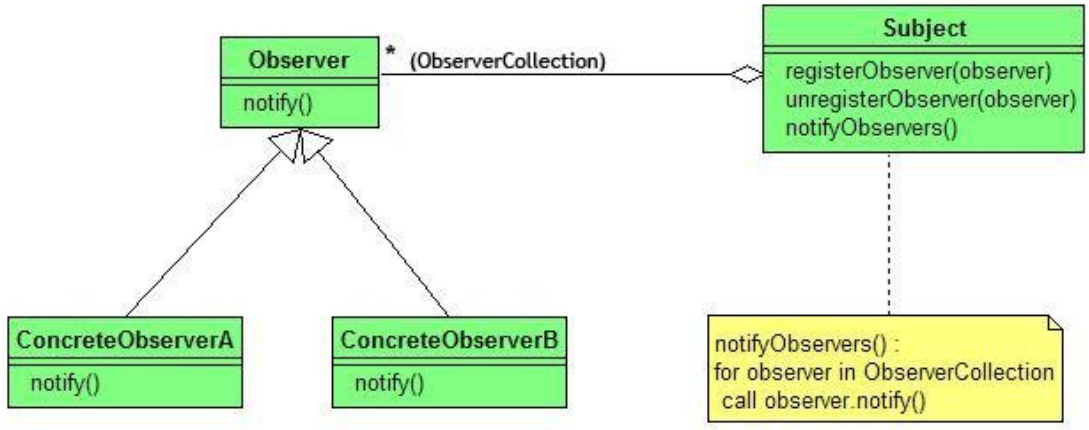

Imagen 66: Patrón Observer

El patrón Facade [109], o Fachada, es un patrón de diseño estructural cuya funcionalidad básica es la de ofrecer una interfaz unificada sencilla que sirva de intermediaria entre un cliente y una interfaz, o conjunto de éstas, con el único fin de hacer al subsistema más sencillo de utilizar.

De esta forma, Facade permite realizar accesos a partes definidas de funcionalidad respecto al todo de un sistema más complejo, permite que una biblioteca con código poco claro y difícil de entender sea más fácilmente accesible, realizar APIs intermedias con mejores diseños que las APIs originales que se desea utilizar o, incluso, aunar un conjunto de funcionalidades muy frecuentes en un código más sencillo y legible. Todo esto implica una disminución de la complejidad y una dependencia mínima entre componentes.

En resumen, las principales ventajas de la utilización del patrón son:

- El cliente que utiliza el Facade no necesita conocer lo que se esconde tras el mismo.

- Los cambios realizados sobre los componentes que engloba el Facade son transparentes para los usuarios.

- Se reduje la complejidad y se minimiza la dependencia.

La representación del patrón en un diagrama sería la siguiente, en la que se aprecia cómo entre la clase Facade y el resto de módulos existe una dependencia, pudiendo dichos módulos utilizar la interfaz ofrecida por esta clase, además de las que ya comparten entre sí.

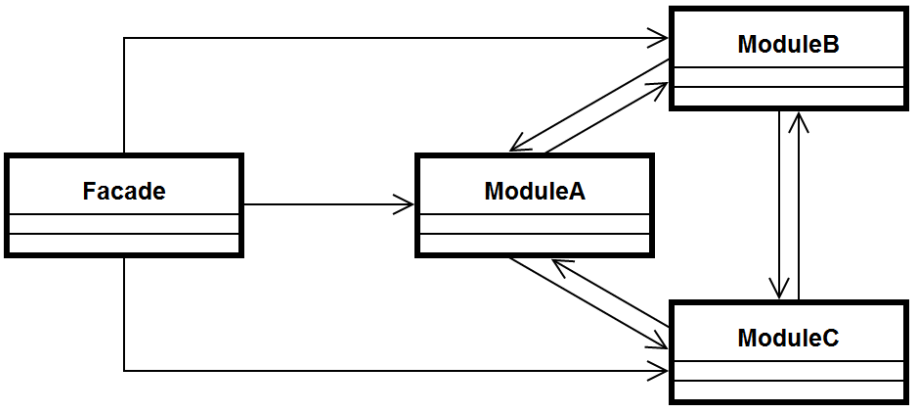

Imagen 67: Patrón Facade

Atendiendo a los subcomponentes que conforman el componente Cliente, se encuentra el motor gráfico, denominado en el diagrama como Entorno Gráfico, y una parte del patrón Observer. El primero, como se ha comentado anteriormente, hace referencia a aquellas clases encargadas de mantener la interfaz gráfica con la que el usuario estará en contacto directo, y a través de la cual navegará con su avatar. El segundo componente implementará el patrón Observer previamente explicado para la parte del cliente. Éste último componente se encarará de atender a los cambios que se realicen en el entorno para notificárselos al Entorno Gráfico y que éste se los muestre al usuario modificando como corresponda su interfaz.

El otro componente de primer nivel de este diagrama, llamado Servidor, alojado en el diagrama de despliegue en el nodo del mismo nombre, está compuesto a su vez de aquellos componentes necesarios para hacer realidad el funcionamiento global del entorno, tanto en la gestión de los usuarios, como en la comunicación, la gestión de los elementos que conforman la interfaz, los avatares controlados por el sistema, y la propia materia a impartir en el mismo.

El componente Facade se corresponde con el patrón de diseño del mismo nombre. Así, cumple con la función de servir de intermediario entre las peticiones del cliente y el componente que pueda responderlas, actuando como filtro y separador de dichas peticiones para que únicamente las reciba aquél que esté destinado a atenderlas.

El componente Controlador Entorno es el encargado de gestionar toda aquella funcionalidad relacionada con la interacción de los usuarios entre sí y con el medio. El componente Controlador Acciones se ocupa de la gestión de los usuarios, ya sean profesores o alumnos, del manejo de los avatares controlados por la aplicación, y de la recuperación de la

información referente a la materia impartida por parte del entorno y dichos avatares. Estos dos componentes serán especificados más en detalle en sucesivos diagramas.

Por último comentar el componente Controlador BDD, que se encargará de ofrecer la infraestructura necesaria para acceder y almacenar la información sobre la aplicación en la base de batos. Google App Engine proporciona una base de datos basada en JDO, por lo que este componente aportará la API requerida para hacer posible esta gestión sobre dicha base de batos. De esta manera, se tendrán las librerías "DataNucleus.jar" y "JDO.jar", ambos proporcionados por Google App Engine, así como el documento XML "jdoconfig.xml", destinado a la configuración del JDO.

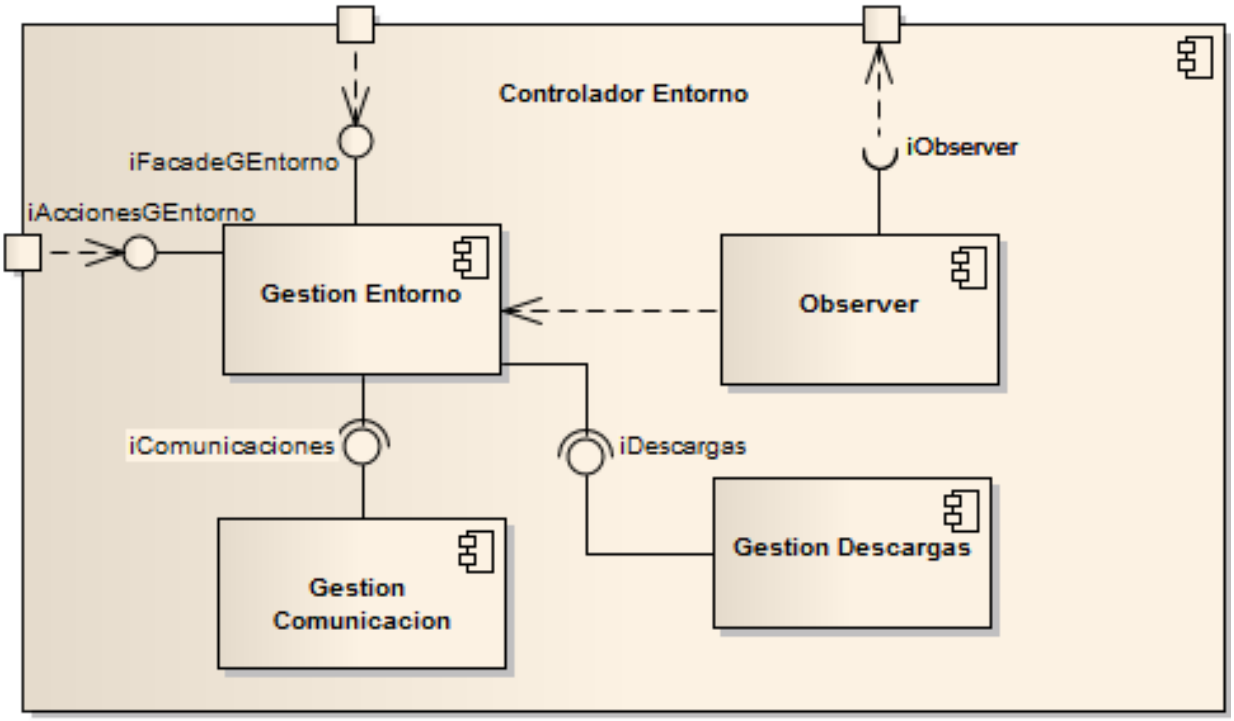

Imagen 68: Diagrama de componentes: Controlador Entorno

Como se ha comentado con anterioridad, este componente, Controlador Entorno, es el encargado de gestionar el entorno propiamente dicho: la interacción de los usuarios con el mundo virtual diseñado y las funcionalidades básicas que en él se recogen. Así, se tendrá un subcomponente para cada una de éstas áreas. El componente Gestión Entorno tendrá la información sobre el estado del entorno y los usuarios dentro de él. De igual forma, será el que reciba todas las peticiones, tanto las que provengan del componente Cliente como de las que le haga el componente Controlador Acciones. Éste dividirá dichas peticiones según su contexto, delegándolas a los componentes Gestión Descargas, el cual atenderá a las posibles descargas y subidas de documentos que hagan tanto profesores como alumnos, y Gestión Comunicación, que se encargará de hacer posible que los usuarios entablen relación entre sí, ya sea mediante conversaciones escritas a través de chats o mediante conversaciones de voz.

Tal y como se aprecia en la imagen, en este componente se encuentra la otra parte del patrón Observer. Éste será el que "observe" los eventos producidos en el entorno, para lo cual atenderá a los cambios que se produzcan dentro del componente Gestión Entorno. Cuando detecte alguna modificación, como puede ser la inclusión de nuevos usuarios en el mundo, se lo notificará a la parte del patrón dentro del Cliente.

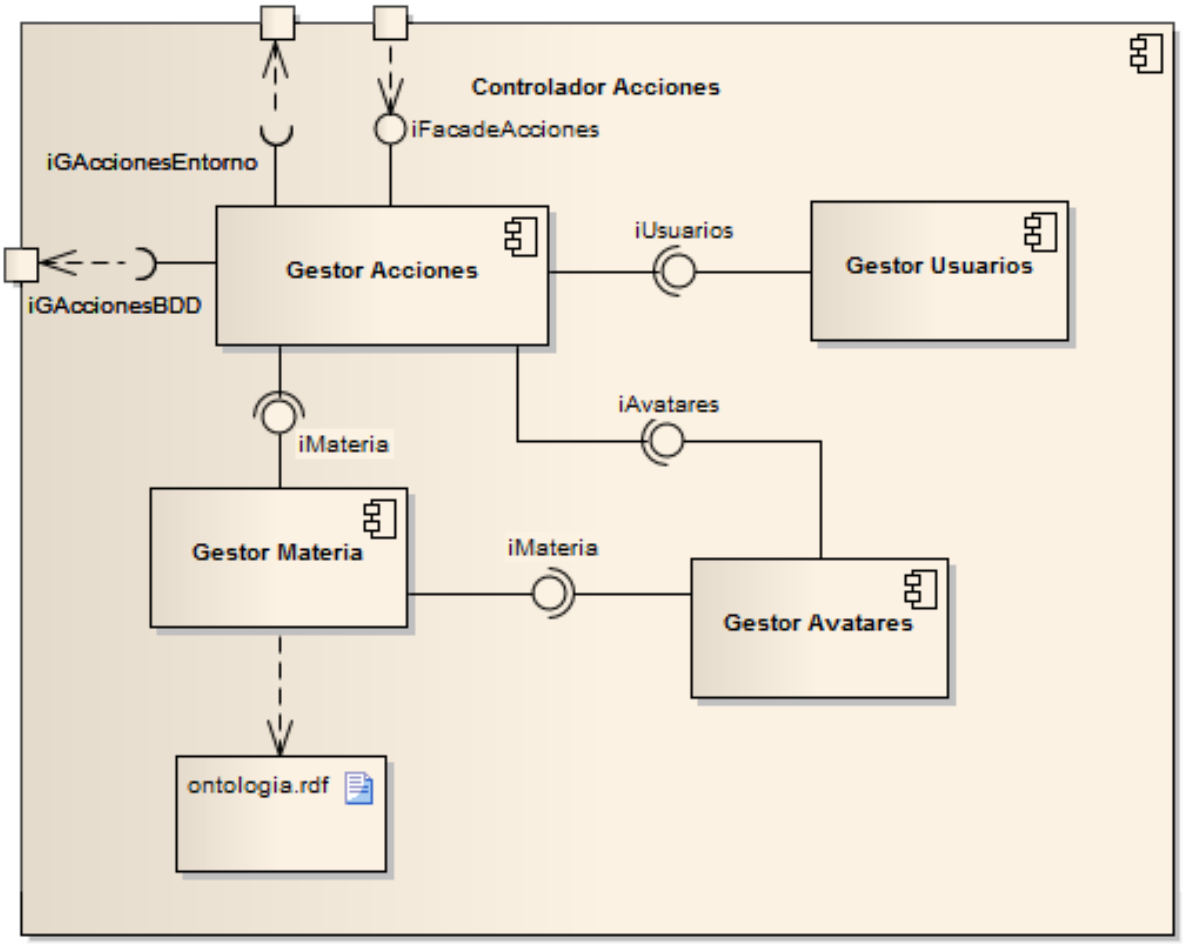

Imagen 69: Diagrama de componentes: Controlador Acciones

Para terminar el análisis sobre el diseño detallado del sistema, se procederá a comentar el último componente: Controlador Acciones. El primero de los subcomponentes, Gestor Acciones, será el encargado de recoger las peticiones traspasadas del patrón Facade relacionadas con los usuarios y las acciones que realizan en el mundo virtual, como por ejemplo pedir información a un avatar sobre un determinado tema. De nuevo, este primer componente dividirá las peticiones a los demás subcomponentes teniendo en cuenta la finalidad de la misma.

El Gestor Avatares se encarga de manejar a los individuos que no están controlados por usuarios externos, sino que son parte de la aplicación. Gestor Materia será el que proporcione, tanto a Gestor Acciones como a Gestor Avatares, la información sobre la materia impartida que está recogida en la ontología, para lo cual deberá contener las clases necesarias para

interpretar y traducir esta última. El Gestor Usuarios será el componente dedicado a manejar la información relevante a los usuarios, independientemente de si éstos son profesores o alumnos.

Este componente se comunica mediante sendas interfaces con el Controlador BDD y con Controlador Entorno, con el primero para recoger la información almacenada en la base de datos, y con el segundo para hacer las comunicaciones necesarias para que se produzca la interacción entre usuarios y entorno.

8. RESULTADOS

8.1. Prototipo

Para que el lector pueda hacerse una idea de cómo podría ser el mundo virtual, visualmente hablando, en este apartado se mostrarán una serie de prototipos de cada una de las salas. Es importante comprender que la imagen final no tiene por qué ser esta misma, simplemente se trata de una posible representación del entorno de enseñanza que se desea desarrollar. Para hacer estos bocetos se han utilizado las herramientas 3DXplorer [106] y PhotoShop [107].

El entorno principal será una sala al aire libre desde la cual se podrá acceder a los diferentes edificios donde se encontrarán los demás entornos.

Imagen 70: Sala común

En la imagen se puede observar lo que podrían ser dos edificios cualesquiera. En total habrá siete edificios, uno por cada ambiente especificado en el apartado “5.2 Escenario de uso: Entorno virtual para Arquitectura de Software”.

Esta sala tiene como funcionalidad básica servir de portal a los demás entornos, así como ser el punto de consulta de información general del mundo virtual. Para ello contará con la inclusión de tres paneles informativos: el mapa del mundo junto con la información sobre los profesores, horarios, forma de comunicarse con ellos, etc., el calendario con los eventos importantes en el ámbito académico del curso, y por último, el tablón dedicado al foro entre compañeros para resolver dudas. Para poder visualizar correctamente la información de los

mismos, tan sólo se tendrán que situar en frente suya y pulsar con el ratón encima de ellos, abriéndose en esos momentos una pequeña pantalla con la información correspondiente.

Otro de los elementos interesantes es el buzón a través del cual los alumnos podrán comunicarse con los profesores. Para su utilización, de nuevo los estudiantes únicamente tendrán que situarse a su lado y pulsar con el ratón encima del elemento con forma de buzón de correos, en la imagen representado como un buzón amarillo tradicional en la parte izquierda de la pantalla. En ese momento se abrirá una ventana para el envío de correo electrónico que el alumno rellenará según sus necesidades. Los profesores, en caso de ser necesario, contestarán a los alumnos en su dirección de correo electrónico particular.

Los profesores, por su parte, podrán modificar la información que se muestre en el calendario y sobre ellos mismos, dándoles el sistema esta posibilidad cuando sean ellos quienes pulsen sobre los correspondientes tablones.

Para proporcionar información sobre el mundo, dos avatares se situarán en esta sala. Los alumnos podrán acercarse a ellos y éstos les darán información sobre la sala en la que se encuentran y cómo acceder al resto de funcionalidades del entorno.

Desde la sala anterior, uno de los edificios a los que se puede entrar es la zona de teoría. Esta parte se compone de cinco salas independientes, una por cada tema de la materia. A su vez, dentro de cada una de estas salas existirán diferentes zonas para explicar cada una de las partes de los temas propuestos. Las dos imágenes siguientes podrían representar dos zonas diferentes dentro de un mismo tema.

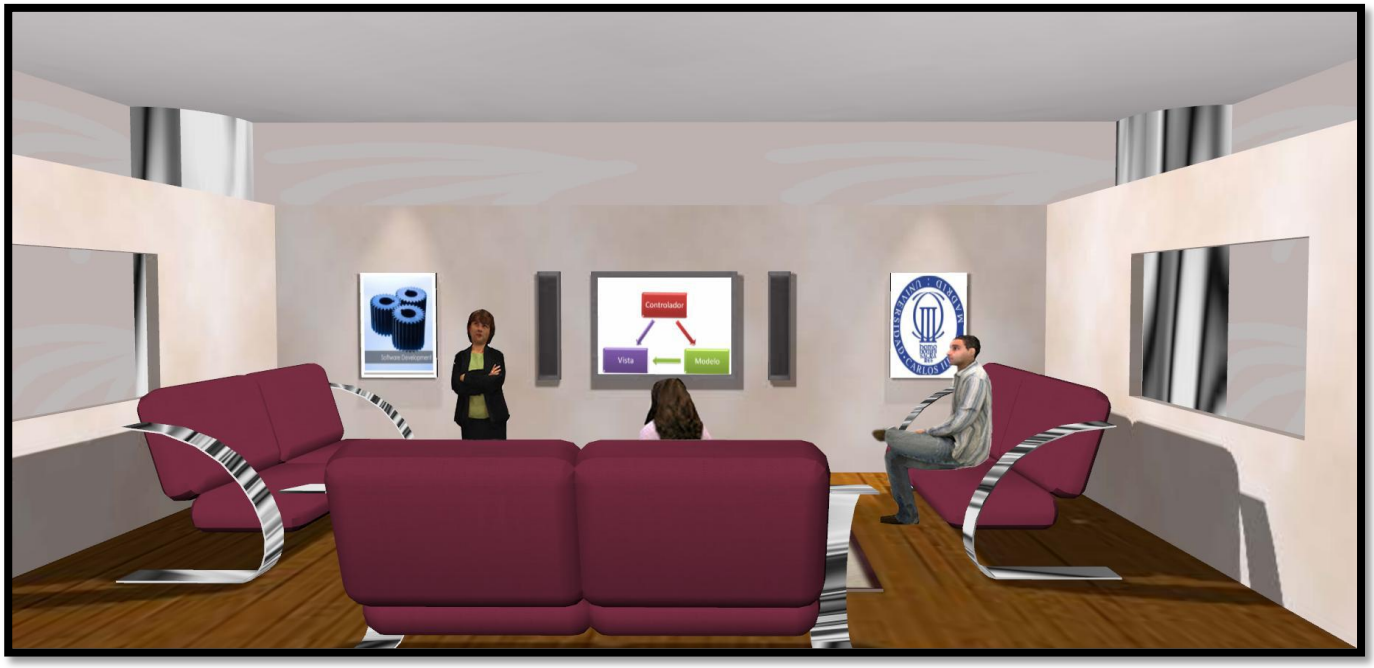

Imagen 71: Sala de teoría (I)

Imagen 72: Sala de teoría (II)

Las pantallas que se pueden observar representan el material audiovisual disponible. Cuando el alumno se sitúe en frente y pulse sobre la pantalla, si se trata de un vídeo éste comenzará a reproducirse. También es posible que se trate de una sucesión simple de imágenes. Los alumnos podrían ver los contenidos de estas pantallas tantas veces como quieran.

Como apoyo teórico a estos vídeos e imágenes, se introducirá la participación de avatares controlados por el propio sistema, en las imágenes representados por las mujeres vestidas con el traje negro. Como apunte comentar que para la simplificación de estos prototipos, la imagen de los avatares es siempre la misma, cosa que no tiene por qué ocurrir en la versión final desarrollada. Este avatar será accesible a los usuarios cuando pulsen sobre él. De esta manera, accederán al contenido teórico almacenado en la ontología, podrán ver las relaciones entre los términos específicos del tema que estén estudiando, sinónimos, definiciones, notas sobre dichos términos, etc. Así podrán construirse un mapa conceptual sobre la teoría, provocando que su comprensión sobre ésta sea lo más amplia posible.

Para que los alumnos den un buen uso a la teoría expuesta durante el curso es necesario que la apliquen a casos prácticos. Para ello se plantean ejercicios a realizar individualmente. Todos los alumnos deberán completar una serie de problemas, los cuales irán aumentando en dificultad según se vayan resolviendo correctamente, obteniendo puntuaciones cada vez más altas, promoviendo así su motivación frente a la materia.

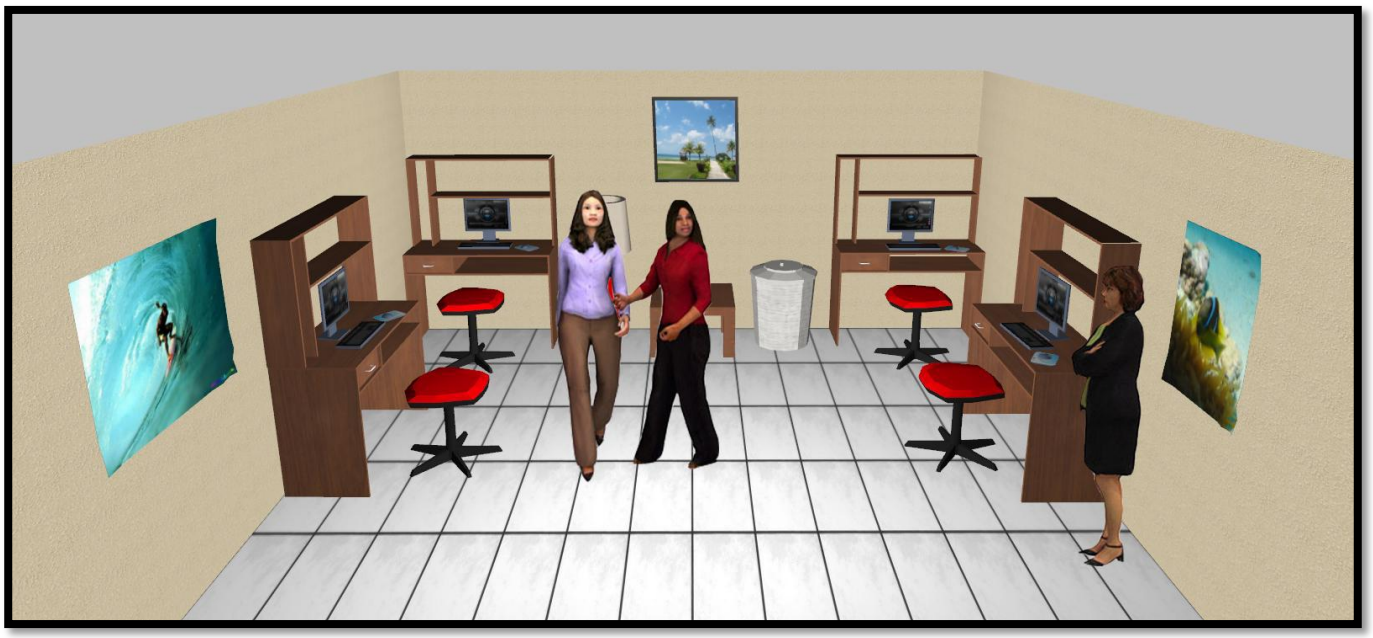

Imagen 73: Sala de ejercicios individuales

Cada una de las estaciones individuales de trabajo estará representada en el mundo virtual como un escritorio y un ordenador personal en él. Los alumnos escogerán una de estas mesas y pulsarán encima del ordenador. En ese momento, se abrirá una ventana desde la cual se le presentarán los ejercicios. Estos ejercicios se irán desbloqueando según el alumno vaya superando los anteriores, aumentando la dificultad durante el proceso, obteniendo una puntuación determinada teniendo en cuenta lo bien que lo haya hecho. Los ejercicios estarán compuestos de test y problemas cuya corrección sea posible a través del propio programa.

En esta aula estarán disponibles para los estudiantes dos avatares, los cuales tendrán información acerca de la sala y de cómo realizar los ejercicios que en ella se practican. Los usuarios deberán pulsar sobre el avatar deseado para que éste les proporcione la información necesitada.

El curso, además, implica la realización de prácticas en grupo que han de completarse para que el alumno se considere apto al final del curso. Dichas prácticas podrán llevarse a cabo en una sala específica para ello, en la cual se reunirán los grupos, adoptando cada miembro un role, y resolverán, teniendo en cuenta esa distribución, las prácticas propuestas.

Imagen 74: Sala de ejercicios grupales

Los alumnos se sentarán en grupo. Al pulsar sobre el ordenador se les abrirá una aplicación a través de la cual podrán mantener conversación por chat o por voz, según ellos elijan. Además, mediante otra ventana, tendrán acceso a la práctica propiamente dicha. En cada periodo del curso tendrán una única práctica disponible, con una fecha de entrega, pudiendo realizarse dicho envío únicamente desde esta sala y a través del alumno que tome el role de jefe de proyecto. Antes de empezar, los alumnos que conformen el grupo deberán identificarse y especificar cuál es el role que van a desempeñar dentro del grupo. A partir de ahí, irán realizando la práctica cuando ellos consideren oportuno, pudiendo en cualquier momento moverse por el mundo para consultar bibliografía, documentación externa, teoría, etc. Cuando hayan terminado el proyecto, lo notificarán en la aplicación y será enviado a uno de los profesores por correo electrónico para que proceda con su corrección.

Los profesores tendrán la funcionalidad añadida de incluir en el sistema nuevas prácticas, asignándolas una fecha máxima de entrega, y de eliminarlas del mismo cuando dicha fecha haya expirado.

Aquí también podrá contarse con la presencia de avatares controlados por la aplicación, dando nuevamente a los alumnos las instrucciones necesarias para que puedan utilizar de forma óptima la sala en la que se encuentran.

Otra forma de mejorar sus conocimientos es a través de la realización de ejercicios extras propuestos por los profesores. Para acceder a los mismos, los alumnos deberán dirigirse a la sala ambientada en una cafetería.

Imagen 75: Cafetería

Los alumnos, por grupos, podrán sentar a sus avatares en las mesas de la cafetería, sin existir un número determinado por grupo. Uno de ellos se dirigirá a la barra y solicitará al avatar, pulsando sobre él, un problema. El avatar le dará la opción de elegir el área al que pertenece el problema y la dificultad. Ese alumno volverá a la mesa y pulsará sobre ésta, abriendo así una conversación sobre el mismo. Los estudiantes que pulsen después sobre esta misma mesa se unirán al mismo problema y a la misma conversación, por lo que podrán comentar entre ellos la manera de resolverlo de forma grupal y colaborativa. Los alumnos, además, podrán elegir si desean que la conversación se desarrolle de manera textual o por voz.

Mientras los alumnos realizan estos ejercicios, prácticas, ejercicios individuales, o simplemente cuando están estudiando la teoría, pueden surgirles dudas. En ese caso, pueden acudir a los profesores y sus tutorías, establecidas en un determinado horario, el cual estará especificado en el tablón de anuncios de la sala común, junto al mapa del sitio. Este entorno será representado como un jardín, a fin de tratarse de un ambiente relajado y distendido en el que los alumnos y profesores puedan conversar acerca de las dudas de los primeros.

Durante el horario de tutorías los profesores accederán con sus respectivos avatares a esta sala, y estarán disponibles para contestar las preguntas que les puedan surgir a los estudiantes. Para iniciar una conversación, se deberá pulsar encima del profesor y así empezar un diálogo, ya sea por chat o voz, a la cual podrán sumarse más compañeros en caso de así desearlo.

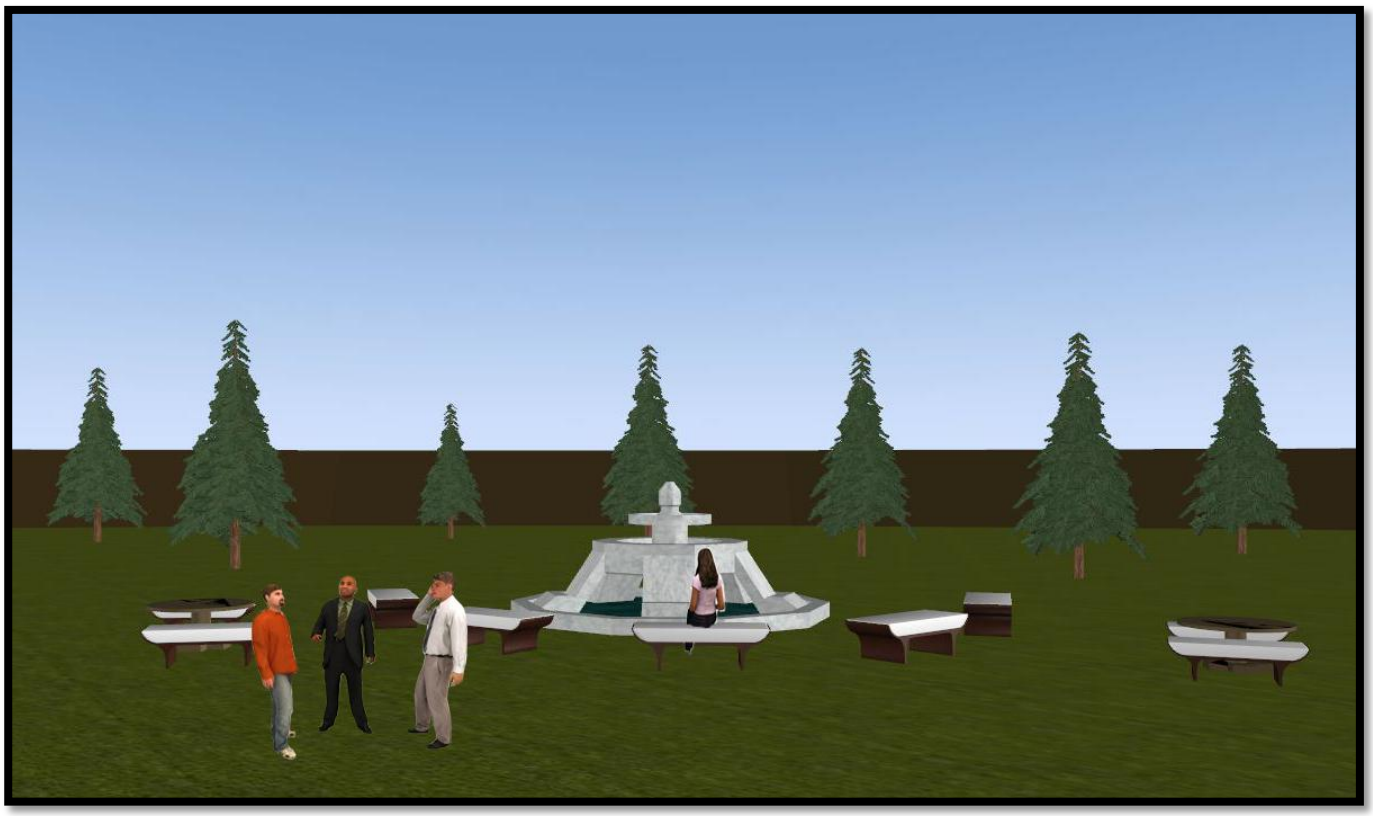

Imagen 76: Jardín

Para complementar los estudios teóricos, y como base de consulta para la realización de ejercicios, los profesores pondrán a disposición de los alumnos una bibliografía complementaria, así como diversos enlaces externos sobre los temas tratados durante el curso. Esta sala tendrá el aspecto de una biblioteca.

Imagen 77: Biblioteca

Los alumnos podrán acceder a esta sala para realizar consultas puntuales o para obtener sitios de referencia externa, como páginas web o información sobre libros de texto relacionados con el temario. Para aquellos documentos que hayan sido subidos por los profesores, como documentos, imágenes o archivos, los alumnos deberán dirigirse a las estanterías, representadas en la imagen anterior al final de la misma. Cuando pulsen sobre ellas, se les abrirá un formulario en el que podrán ver cuáles son los documentos disponibles, organizados por temas y por tipo de documento, y seleccionar aquellos que desea, para que la descarga pueda comenzar.

Si lo que desea es obtener información extra, podrá acercarse a la zona de ordenadores. Al pulsar sobre ellos se abrirá un formulario similar al anterior, en el que podrán navegar por los temas de la materia hasta que lleguen al deseado. Una vez en él, podrán ver todos los enlaces disponibles para esa área concreta y visitarlos a través de su propio navegador Web.

Los profesores, por su parte, podrán añadir y eliminar tanto nuevos enlaces como documentos, interaccionando con las estanterías y ordenadores de igual manera que lo hacen los alumnos, aunque escogiendo la opción de añadir o eliminar el recurso.

Por último, para que los responsables del curso puedan comprobar la efectividad de éste y su utilización y comprensión por parte de cada uno de los alumnos, se realizarán una serie de exámenes evaluables. La sala destinada a tal fin será una clase tradicional.

Imagen 78: Clase

Los exámenes estarán fijados en una fecha concreta, aunque podrán realizarse a lo largo de todo el día para tener en cuenta a aquellas personas que no dispongan de un horario flexible o que cuenten con ciertas restricciones de tiempo. Cuando deseen acudir al examen, deberán entrar en esta aula, habilitada únicamente las fechas señaladas, y sentarse en una mesa. Cuando pulse sobre ésta, se le abrirá una aplicación, momento en el cual empezará el examen. Una vez comiencen el examen no podrán abandonar el aula ni entablar conversación con otros compañeros.

Para resolver las dudas sobre cómo funciona el aula de exámenes y el propio proceso, en el aula se situará un avatar, que proporcionará esta información a los alumnos que se lo soliciten pulsando sobre él.

El mapa final podría resultar de la siguiente manera:

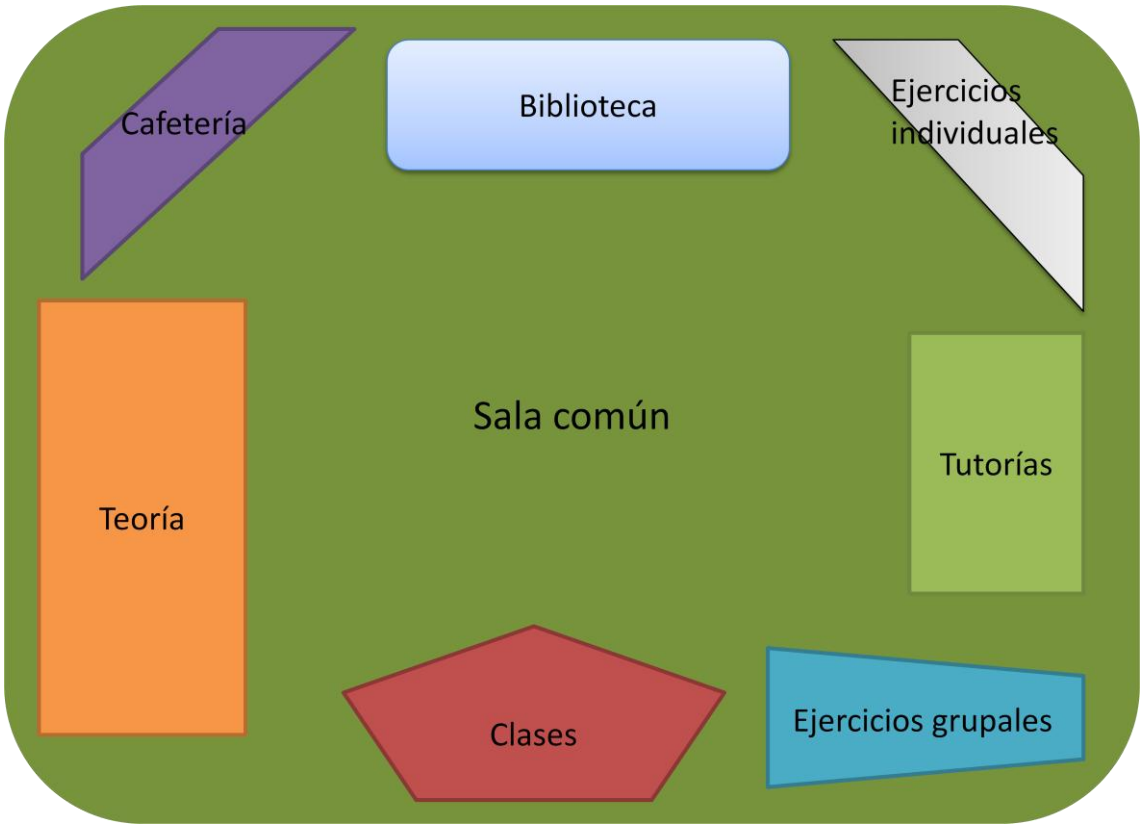

Imagen 79: Mapa del mundo virtual

8.2. Resultados de las encuestas

Para ver la aceptación que este tipo de formación tendría, se ha realizado una encuesta (ver Anexo 1: Encuesta) entre un grupo de personas de distintas edades y estudios. La muestra está formada por 54 individuos de ambos sexos, de entre 21 y 59 años. A continuación se adjunta la descripción de la muestra:

	N	Mínimo	Máximo	Media	Desviación típica
Edad	54	21	59	28.39	7.093

Tabla 79: Tabla de estadísticos descriptivos

VARIABLES	FRECUENCIA	PORCENTAJE
Mujer	19	35.2
Hombre	35	64.8
Total	54	100.0

Tabla 80: Tabla de frecuencia por sexos

VARIABLES	FRECUENCIA	PORCENTAJE
Titulado	22	40.7
Estudiante	19	35.2
Postgrado	10	18.5
Diplomado	2	3.7
Doctor	1	1.9
Total	54	100.0

Tabla 81: Tabla de frecuencia por estudios

VARIABLES	FRECUENCIA	PORCENTAJE
Ingeniería electrónica	2	4.2
Ingeniería Informática	36	75.0
Ingeniería Técnica en Informática	1	2.1
Química	1	2.1
Física	2	4.2
Ingeniería Industrial	1	2.1
Biología	2	4.2
Ingeniería en Telecomunicaciones	1	2.1
Ingeniería Civil	1	2.1
Ciencia y Tecnología de la Información	1	2.1
Total	48	100.0

Tabla 82: Tabla de frecuencias por carrera

Dentro de la muestra se observa una gran variabilidad de carreras (Tabla 83), lo que dificulta el análisis y la obtención de resultados, por lo que se ha creído conveniente dividir a los encuestados entre “informáticos y no informáticos”, quedando la muestra distribuida de la siguiente manera:

VARIABLES	FRECUENCIA	PORCENTAJE
Informáticos	37	77.1
No Informáticos	11	22.9
Total	48	100.0

Tabla 83: Tabla de frecuencias entre informáticos y no informáticos

Para realizar el análisis se ha utilizado el Programa Estadístico SPSS [105], haciéndose análisis tanto de Frecuencias como ANOVA de un factor y las pruebas T de Student. De estos últimos estudios estadísticos no se han obtenido diferencias significativas dentro de la muestra, sin embargo, se ha considerado interesante adjuntar en este proyecto algunos resultados de frecuencias, como muestra de las diferencias parciales encontradas dentro de los grupos (tanto sexo como nivel de estudios), dejando de lado la diferencias entre edades, puesto que no se consideran relevantes.

Las variables que se han considerado importantes a nivel de frecuencias son las siguientes:

- Se ha visto que una gran mayoría de la muestra (87%) consideraría interesante la impartición de cursos virtuales, independientemente de su nivel de estudios y sexo (Tabla 84 e Imagen 80). También se les pidió que opinaran sobre los cursos virtuales, encontrando, en su mayoría, opiniones favorables al respecto (Tabla 85).

VARIABLES	FRECUENCIA	PORCENTAJE
Si	47	87.0
No	7	13.0
Total	54	100.0

Tabla 84: Tabla de frecuencias atendiendo a si desea que le impartan cursos virtuales

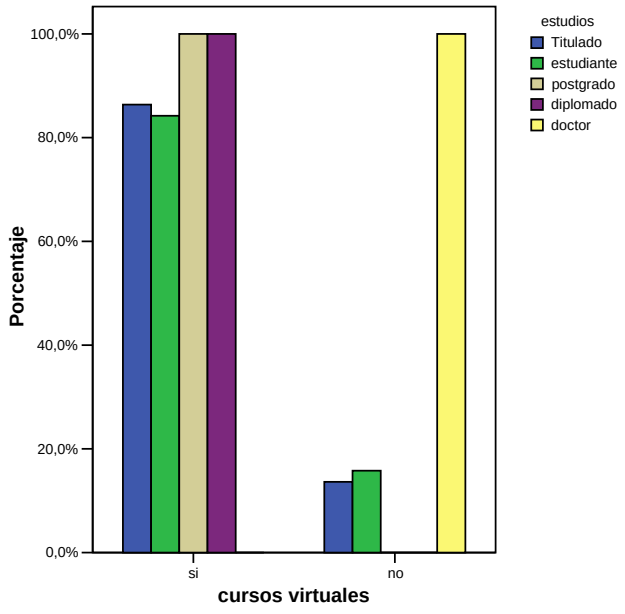

Imagen 80: Personas que realizarían cursos virtuales en función de su nivel de estudios

VARIABLES	FRECUENCIA	PORCENTAJE
Flexibilidad de horarios	18	40.9
Mayor motivación	13	29.5
Incorporación de tecnologías	6	13.6
No da resultados eficientes	5	11.4
Seguimiento complicado	2	4.5
Total	44	100.0
Perdidos	10	
Total	54	

Tabla 85: Tabla de frecuencias atendiendo a comentarios sobre los cursos virtuales

- Frente a la pregunta de valoración sobre si les resultaba novedoso este tipo de formación, se observa una gran variedad de opiniones, si bien es cierto que para la mayoría (77.8%) resulta novedoso o muy novedoso (Tabla 86 e Imagen 81).

VARIABLES	FRECUENCIA	PORCENTAJE
Muy poco novedoso	1	1.9
Poco novedoso	1	1.9
Normal	10	18.5
Novedoso	21	38.9
Muy novedoso	21	38.9
Total	54	100.0

Tabla 86: Tabla de frecuencias atendiendo a si le parece un método novedoso

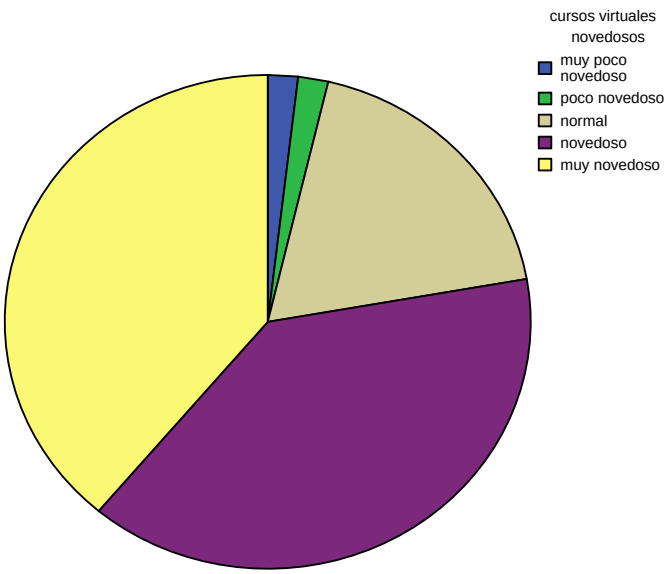

Imagen 81: Apreciación de los cursos virtuales en cuanto a su novedad

- También se solicitó a los encuestados que resumieran sus conocimientos sobre mundos virtuales de enseñanza. Se les preguntó sobre si conocían un método

similar (Tabla 87, Tabla 88 e Imagen 82) y si lo habían utilizado con anterioridad (Tabla 89). De este bloque de preguntas, cabe destacar que el 68,5% de los encuestados no han utilizado nunca este método de enseñanza y el hecho de que, pese a no haber diferencias significativas, se ve que hay un mayor conocimiento en este ámbito de los informáticos, frente a los encuestados que no pertenecen a este sector (Imagen 83 e Imagen 84).

VARIABLES	FRECUENCIA	PORCENTAJE
Si	24	44.4
No	30	55.6
Total	54	100.0

Tabla 87: Tabla de frecuencias atendiendo a si conoce un método similar

VARIABLES	FRECUENCIA	PORCENTAJE
Moddle	4	7.4
Wonderland	1	1.9
Second Life	7	13.0
WikiUniversity	2	3.7
Formación online	6	11.1
Formación en entornos 3D	2	3.7
No sabe / No contesta	32	59.3
Total	54	100.0

Tabla 88: Tabla de frecuencias sobre los mundos virtuales de enseñanza conocidos

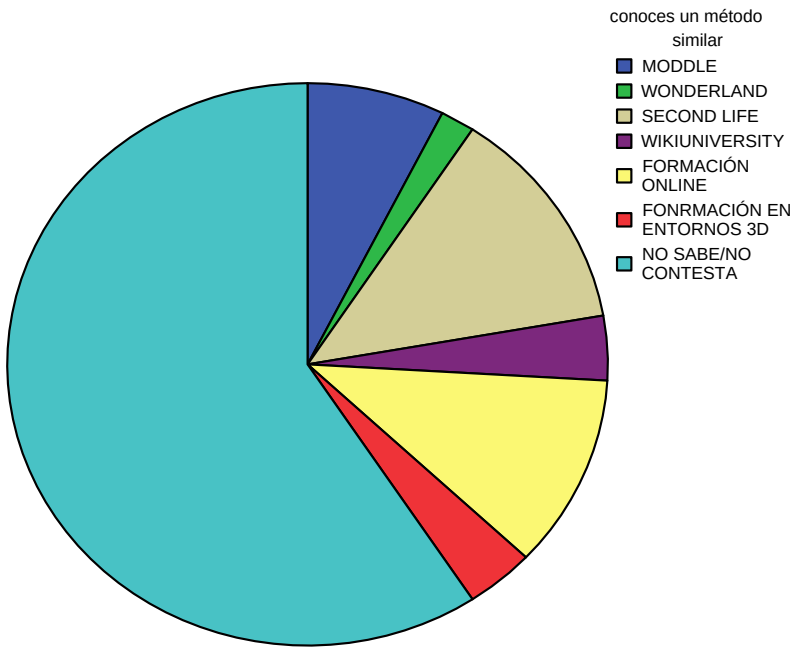

Imagen 82: Mundos virtuales de enseñanza conocidos

VARIABLES	FRECUENCIA	PORCENTAJE
Si	17	31.5
No	37	68.5
Total	54	100.0

Tabla 89: Tabla de frecuencias de utilización previa de mundos virtuales

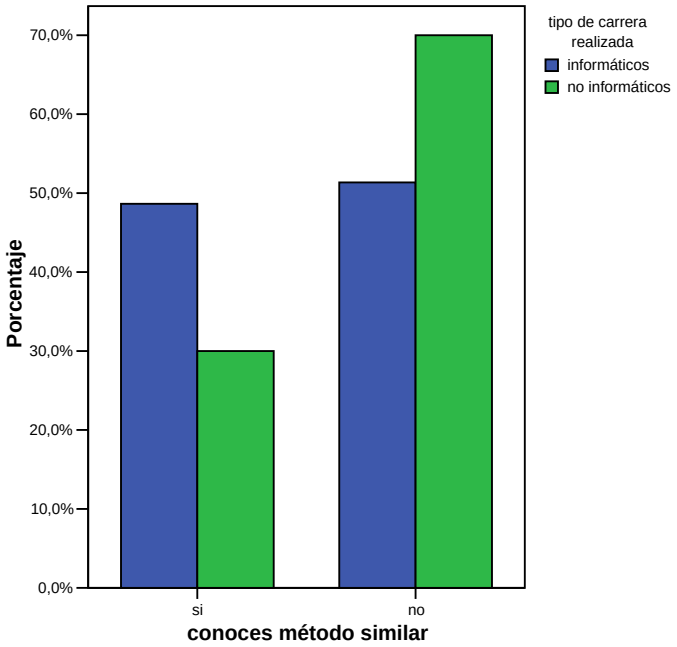

Imagen 83: Conocimiento de métodos similares en función del tipo de estudios

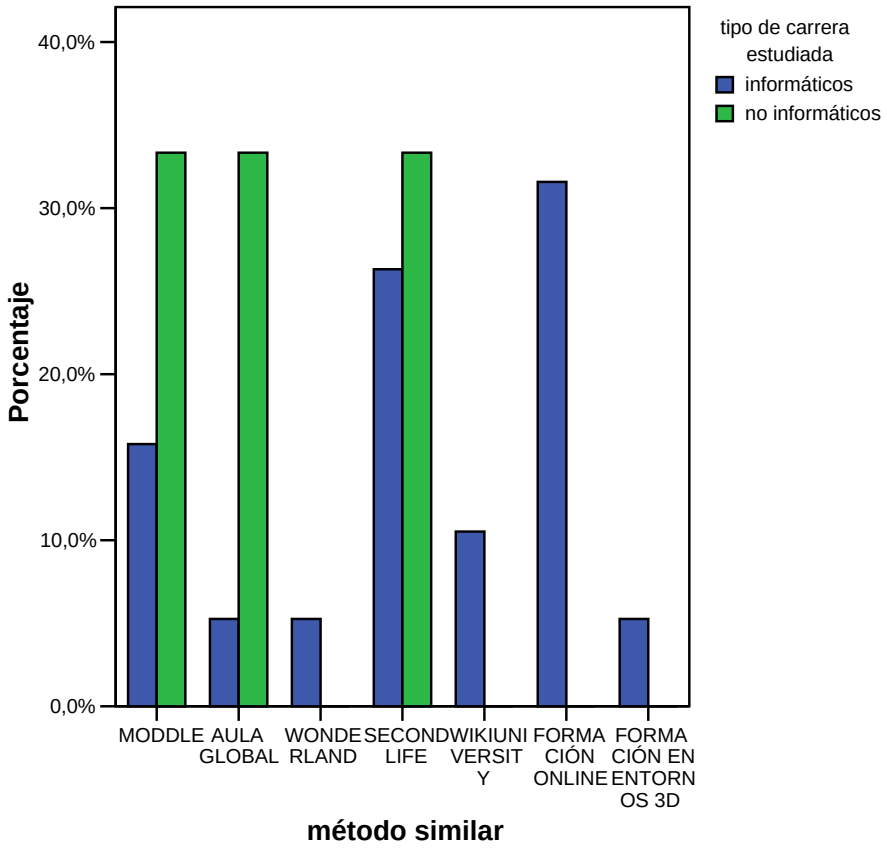

Imagen 84: Métodos similares conocidos en función del tipo de estudios

- Otra tanda de preguntas se refirió a los métodos de enseñanza online, tanto si habían realizado cursos de este tipo (Tabla 90) como su opinión al respecto (Tabla 91).

VARIABLES	FRECUENCIA	PORCENTAJE
Si	28	51.9
No	26	48.1
Total	54	100.0

Tabla 90: Tabla de frecuencias atendiendo a si ha realizado cursos online

VARIABLES	FRECUENCIA	PORCENTAJE
Bueno	17	60.7
Regular	7	25.0
Malo	4	14.3
Total	28	100.0
Perdidos	26	
Total	54	

Tabla 91: Tabla de frecuencias sobre los resultados obtenidos en los cursos online

Además de los datos mostrados anteriormente, se dejó un campo abierto en la encuesta para que los participantes dieran su opinión de forma más explícita. A continuación se muestran las conclusiones tanto del campo abierto como de los resultados analíticos:

- La realización de cursos virtuales, que permitan libertad de actuación por parte de los usuarios resulta una opción muy aceptada, previéndose una alta participación.

- Si bien es verdad que el desconocimiento de este ámbito es generalizado, se observa un mayor conocimiento del mismo dentro del grupo de informáticos, lo cual era algo de esperar, teniendo en cuenta su especialización.

- El mundo virtual más nombrado entre los individuos de la muestra es Second Life, conocido sobre todo por su carácter social, si bien entre los informáticos hay una mayor variación en las respuestas.

- Dada la aceptación de los cursos online actuales, que poseen un carácter frío e individualizado, podemos decir que la creación de cursos virtuales con una estructura más social, sin perder las ventajas de la enseñanza a distancia puede ser un gran avance no sólo para la tecnología actual, sino también para el mundo de la enseñanza y, sobre todo, para las universidades, que podrán aceptar a un mayor número de alumnos, al no necesitar un espacio físico de grandes dimensiones.

- Por último, cabe destacar que la mayoría de los encuestados estarían dispuestos a realizar estos cursos siempre y cuando supongan una ventaja frente a la enseñanza normal, sin perder las características de ésta. Es decir, es necesario crear mundos virtuales que permitan una comunicación profesor-alumno fluida y cuyo acceso esté al alcance de todos.

9. CONCLUSIONES

Tras haber finalizado el proyecto, es momento de hacer un repaso general de los datos recogidos y las impresiones finales sobre el sistema que se desea obtener.

Como cualquier producto que se desea implantar en el mercado es importante conocer cuáles son las posibilidades de aceptación entre la población. El sector donde se desea introducir el software es el mundo de la enseñanza. Cada vez hay más cursos por Internet y a distancia, la gente tiene menos tiempo para acudir a clases, sobre todo cuando se están realizando estudios superiores, ya sea porque se trabaje simultáneamente, por cuestiones geográficas, o por cualquier otra causa o responsabilidad. El hecho de que el sistema proporcione un curso desvinculado de un aula es un gran aliciente para aquellos que deseen adentrarse en el mundo de la Arquitectura de Software.

Otra de las ventajas que este software proporciona es que no se trata de un típico curso a través de la red en el que dan la teoría, y puede que ejercicios que hacer cada cual en su propia casa. Lo que se busca realmente es no perder el trato personalizado que dan las clases tradicionales, esa interacción entre alumno – profesor y entre los propios compañeros, no desestimando la importancia de las relaciones sociales para la solución de los problemas que se plantean, tanto en la materia impartida como en el mundo laboral real.

Además, el sistema toma ventaja de las facilidades que aporta ser un sistema software, ya que, para la enseñanza, apoya a los profesores con material audiovisual y avatares controlados por el sistema que ayudan a los alumnos a construirse mapas conceptuales y a entender la teoría, aunque puedan también consultar directamente a los profesores.

En cuanto a la parte técnica, es reseñable que utilice nuevas tecnologías que hagan que sea un software moderno con posibilidades reales de adentrarse en el mundo de la enseñanza. La inclusión de ontologías es un avance a la hora de utilizar bots para impartir la teoría a los estudiantes, permitiendo relacionar términos entre sí, y dotándolos de información semántica muy útil para su labor.

De igual forma, utilizar Cloud Computing, en este caso con Google App Engine, es una gran ventaja teniendo en cuenta que el software se accederá a través de Internet. En caso de querer utilizarse un servidor propio, se tendría que controlar las posibles caídas del sistema, ataques externos, balanceo de carga, etc., sin contar con el coste económico extra que esto conllevaría. Con los servidores de Google, esta tarea se facilita enormemente, siendo ellos quienes garantizan todas estas condiciones que deben cumplirse para que el sistema de enseñanza planeado tenga posibilidad de éxito.

Siendo analíticos, el sistema ofrece grandes ventajas para considerarse viable: permite una gran flexibilidad de horarios, permitiendo que los alumnos accedan al mismo cuando deseen, a excepción de horas de tutoría, aunque siempre podrán quedar de manera personal con el profesor deseado. Asimismo, no hay restricciones geográficas, no se necesitan

desplazamientos para introducirse en este mundo virtual, facilitando el encuentro de gente de cualquier parte. Evidentemente, no todo son ventajas, sí que se pierde trato personal, no es lo mismo encontrarse cara a cara con una persona que hablar con ella a través de un chat. Además, cualquier fallo en la red puede ser crucial para los alumnos si no se les permite acceder al sistema o si, en el peor de los casos, se pierde la información almacenada sobre ellos y su avance en el curso.

Resumiendo, si bien es cierto que estos últimos fallos comentados pueden ocurrir, el soporte elegido a través de Google minimiza estos riesgos. En general, se trata de un sistema novedoso en cuanto a la forma de impartir los cursos, al mezclar la enseñanza con profesores y una forma de E-Learning a través de los avatares propios del sistema, ofreciendo unas ventajas destacables como son la flexibilidad horaria y geográfica. Además, se ha realizado un estudio sobre técnicas de enseñanza a fin de encontrar la mejor manera de impartir la materia, haciendo que los alumnos se involucren en la tarea, motivándolos y ayudándoles a obtener los mejores resultados posibles, quedando así satisfechos con el resultado.

Si se piensa en el futuro, se pueden imaginar cuáles son los siguientes pasos para este tipo de sistemas. En este caso, el proyecto se ha centrado en la asignatura Arquitectura de Software, siendo el siguiente movimiento adaptarlo para impartir cualquier asignatura, tanto de Ingeniería del Software como cualquier otra área temática. Es evidente que, dadas las características de esta materia, es más fácil que se adapte a asignaturas informáticas, puesto que la mayoría de ellas siguen el mismo patrón dividiendo la carga didáctica entre teoría y práctica, dando gran peso a esta última, incluyendo prácticas obligatorias necesarias para comprobar que se ha entendido correctamente la teoría. Por tanto, se puede entender que no se necesitaría hacer cambios significativos a la hora de querer impartir dichas asignaturas, únicamente habría que adaptar la ontología.

Si se desea que el curso abarque otro tipo de materias, más teóricas, o totalmente prácticas, las bases pueden ser acogidas de igual manera, aunque sí se producirían cambios algo más significativos. De hecho, además de la ontología, cambiarían las técnicas de enseñanza y los ambientes y entornos escogidos, pues deberían adaptarse a las necesidades de los estudiantes.

De cualquier forma, se puede concluir que el presente proyecto marca las bases de lo que podría ser un apoyo en la enseñanza en un futuro, pudiendo incluir estos cursos en las propias universidades, dándolos como alternativas o complementos a las clases tradicionales a las que se está acostumbrado, teniendo, eso sí, que adaptar las técnicas a utilizar y las ontologías.

Respecto a la opinión personal que me merece este proyecto, comentar que me ha parecido una interesante manera de acabar la carrera universitaria en la que estoy inmersa. El proyecto me ha permitido realizar una labor investigativa en la parte relacionada con el Estado del Arte, estudiando técnicas que no conocía y asimilándolas, a la vez que aprendí cómo buscar documentación sobre áreas desconocidas y a entenderlas.

Evidentemente, no todo ha resultado sencillo. Dicha fase inicial fue complicada en sus comienzos debido a la falta de costumbre de buscar documentación técnica y formal sobre

esas áreas, puesto que durante la carrera gran parte de este tipo de documentación nos viene dada por parte de los profesores. De igual modo, la parte de diseño ha sido para mí un punto complejo y clave del proyecto. En las asignaturas previas la dimensión de los sistemas realizados es mucho menor, tratándose, en la mayoría de los casos, de productos sencillos fáciles de entender. La dificultad que éste entrañaba era que se trataba de un mundo virtual, teniendo que modelar su funcionamiento cuando, en un principio, no se conocía realmente.

A pesar de los problemas encontrados, se puede concluir que el proyecto ha resultado para mí satisfactorio, tanto personal como profesionalmente, dándome la oportunidad de sentirme realizada y con la sensación de haber aprendido algo, desde el estudio de nuevas tecnologías, a las propias tecnologías y a abrir la mente a nuevos y más grandes proyectos.

10. PLANIFICACIÓN Y COSTES

10.1. Planificación y costes de análisis

En este apartado se va a proceder a calcular el tiempo a invertir en el presente proyecto, de tal forma que se aporte una métrica del coste que conlleve realizarlo, tanto en esfuerzo en horas como en el aspecto económico.

La duración total del proyecto ha sido aproximadamente de seis meses, desde Julio de 2009 hasta Diciembre de ese mismo año. Puesto que no ha sido necesario compaginar la realización de este Proyecto Fin de Carrera con otras actividades, la jornada laboral de trabajo ha incluido, en la gran mayoría de ese periodo, el día completo y algunos fines de semana. Dicha jornada laboral varió en horas, siendo el primer mes y medio de seis horas diarias, sin contar fines de semana, y el resto de los meses de diez horas diarias, de 10:00 de la mañana a 14:00 de la tarde, y de 16:00 a 21:00. El último mes, además, se trabajó un día del fin de semana, haciendo que la semana de trabajo tuviera seis días en vez de cinco.

Para que el recuento de horas quede más claro, se desglosará en las distintas fases de las que se ha compuesto la elaboración del proyecto.

En primer lugar, se mostrará la planificación del proyecto, realizada con el programa Microsoft Project [111].

	Nombre de tarea	Duración	Comienzo	Fin
1	Reunión inicial	0 horas	mar 16/06/09	mar 16/06/09
2	**Estado del Arte**	**88 días**	**lun 06/07/09**	**lun 02/11/09**
3	Mashups	37 días	lun 06/07/09	mar 25/08/09
4	Entornos virtuales	7 días	jue 27/08/09	vie 04/09/09
5	Cloud Computing	7 días	lun 07/09/09	mar 15/09/09
6	Arquitectura de Software	12 días	mié 16/09/09	jue 01/10/09
7	Ontologías	7 días	vie 02/10/09	lun 12/10/09
8	Hito Reunión Tutor	0 días	mié 14/10/09	mié 14/10/09
9	Técnicas de enseñanza	3 días	vie 30/10/09	lun 02/11/09
10	Herramientas	5 días	vie 16/10/09	jue 22/10/09
11	**Análisis**	**30 días**	**mar 27/10/09**	**sáb 28/11/09**
12	Hito Reunión Tutor	0 días	mar 27/10/09	mar 27/10/09
13	Análisis de herramientas	1 día	jue 05/11/09	jue 05/11/09
14	Obtención de requisitos	4 días	vie 06/11/09	mar 10/11/09
15	Hito Reunión Tutor	0 días	mar 10/11/09	mar 10/11/09
16	Metodología de enseñanza	2 días	mar 10/11/09	mié 11/11/09
17	Casos de Uso	2 días	vie 27/11/09	sáb 28/11/09
18	**Diseño**	**21 días**	**lun 16/11/09**	**mar 08/12/09**
19	Hito Reunión Tutor	0 días	lun 16/11/09	lun 16/11/09
20	Diagramas arquitectónicos	8 días	dom 29/11/09	mar 08/12/09
21	**Resultados**	**10 días**	**jue 12/11/09**	**lun 23/11/09**
22	Prototipos	4 días	jue 19/11/09	dom 22/11/09
23	Encuestas	2 días	jue 12/11/09	vie 13/11/09
24	Hito Reunión Tutor	0 días	lun 23/11/09	lun 23/11/09
25	Conclusiones	1 día	dom 29/11/09	dom 29/11/09
26	Revisión	2 días	lun 30/11/09	mar 01/12/09
27	Hito Reunión Tutor	0 días	mié 02/12/09	mié 02/12/09
28	ENTREGA FINAL	0 días	vie 18/12/09	vie 18/12/09

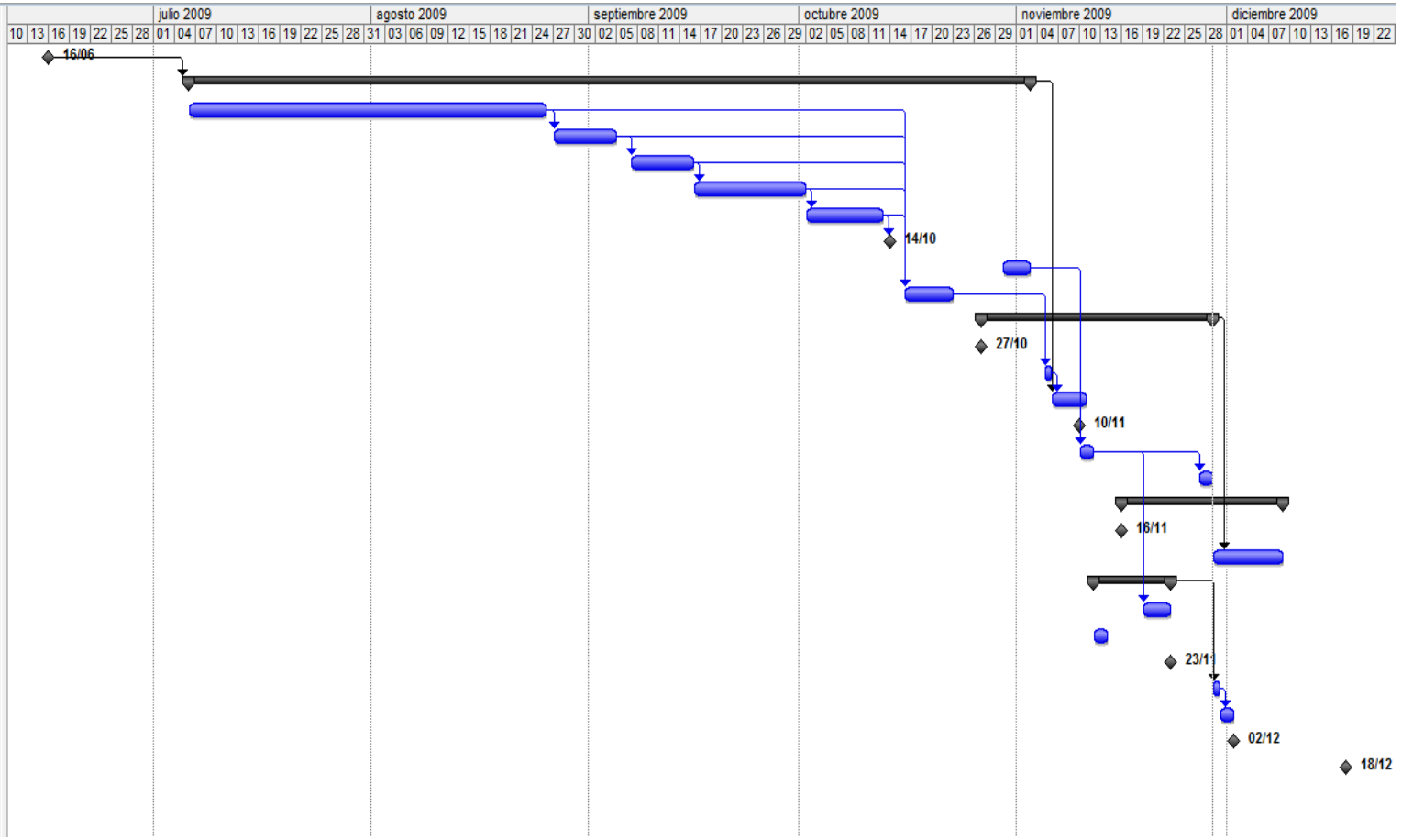

Imagen 85: Planificación

Tras la planificación, se muestra el recuento final de horas empleadas en el desarrollo del proyecto.

FASE	TAREAS	HORAS
Estado del Arte	Reuniones con el tutor	1 hora
	Recopilación y análisis de la bibliografía	514 horas
Análisis	Reuniones con el tutor	2 horas
	Recopilación de requisitos	40 horas
	Análisis de las herramientas	10 horas
	Análisis de las técnicas de enseñanza	20 horas
	Casos de Uso	20 horas
Diseño	Reuniones con el tutor	2 horas
	Diseño	90 horas
Resultados	Prototipos	40 días
	Encuestas	20 horas
TOTAL		759 horas

Tabla 92: Tiempo invertido por fase

El total de horas trabajadas para este proyecto ha sido de: 759 horas.

Una vez hecho el recuento de horas, se procederá a hacer una estimación del coste de la realización del proyecto. Se tendrá en cuenta que sólo ha habido una persona asociada al mismo, y que ha realizado las funciones de analista y diseñador.

- Analista / Diseñador: Verónica Casado Manzanero. Sueldo: 30000 €/año.
    - Coste por hora: 28,73 €/hora.

Siendo el coste total:

EMPLEADO	HORAS	SALARIO	TOTAL
Analista /Diseñador	759 horas	28,73 €/h	21.806,07 €

Tabla 93: Empleados / Sueldo

Dentro de los gastos asociados a la producción del presente documento, hay que tener en cuenta los costes por el hardware y software utilizados. Por convenio está estipulado que en el área de la consultoría se trabaja 1.850 horas/año. Teniendo en cuenta que el periodo de vida útil del hardware y software es de 3 años, el número de horas necesarias para amortizar ambos seria de 1.850 x 3 = 5.550 horas.

Los recursos que se han utilizado han sido los siguientes:

RECURSOS HARDWARE	
Ordenador de sobremesa	1.000 €
Ordenador portátil	900 €
RECURSOS SOFTWARE	
Windows Vista	194 €
Office 2007	460 €
Enterprise Architect	90,15 €
3DXplorer	0 €
Adobe PhotoShop CS4	984,84 €
TOTAL	3.628,99 €

Tabla 94: Recursos

El coste total sumando los puntos anteriores sería:

CONCEPTO	TOTAL
Empleado	21.806,07 €
Recursos	3.628,99 €
TOTAL	25.435,06 €

Tabla 95: Costes

Teniendo el total calculado, solo resta añadirle el beneficio y el riesgo que se estimen oportunos. En este caso se tomará un beneficio del 15% y un riesgo del 5%. Además, hay que contar con un I.V.A. del 16%.

PRESUPUESTO TOTAL	35.626,89 €

Tabla 96: Presupuesto total

El presupuesto total por la realización del proyecto sería **35.626,89 € (treinta y cinco mil seiscientos veintiséis euros con ochenta y nueve céntimos)**.

10.2. Costes de desarrollo del software

Una vez visto el coste de realización del proyecto, se va a proceder a realizar el cálculo de los gastos que conllevaría el desarrollo e implementación del mismo. Para ello, se ha contado con la métrica de Puntos Caso de Uso.

10.2.1. Los Casos de Uso como métrica

El método de Puntos Caso de Uso (UCP, Use Case Points) [110] se fundamenta en los tradicionales Puntos Función, y fueron diseñados por Gustav Karner en su tesis de máster en 1993 bajo la supervisión de Ivar Jacobson.

Los pasos a seguir para poder utilizar esta técnica en un proyecto son similares a los utilizados con los Puntos Función:

- De una revisión inicial de los requerimientos se obtiene un recuento de Puntos Casos de Uso sin ajustar (UUCP, Unadjusted Use Case Points).
- Realizar un análisis de los factores técnicos del proyecto.
- Ajustar los factores de ajuste para obtener los Puntos Caso de Uso Ajustados (UCP) necesarios para realizar la estimación del esfuerzo del proyecto en horas/hombre.

10.2.1.1. Cálculo de los Puntos Caso de Uso sin Ajustar (UUCP)

Para realizar este cálculo inicial de Puntos Caso de Uso es necesario realizar los siguientes tres pasos:

- Clasificar las interacciones entre actor y caso de uso en función de su complejidad y asignarle un peso.

Se definen tres categorías diferentes de complejidad en función del tipo de interacción entre actor y caso de uso: simple, medio y complejo. La siguiente tabla ilustra estos grados de complejidad y los pesos asignados a cada una:

TIPO DE INTERACCIÓN	PESO
Simple (a través de un API)	1
Medio (a través de un protocolo)	2
Complejo (a través de una interfaz)	3

Tabla 97: Complejidad Interacción Casos de Uso - Actor

La interacción simple representa una API utilizada por otro software. En el caso de un actor medio, se puede asumir que éste sea otro sistema que realice conexiones a través de un protocolo tipo TCP/IP o un usuario a través de una consola de comandos. Por último, un actor complejo se trataría de un usuario interactuando a través de una interfaz gráfica compleja con el sistema.

- Calcular la complejidad de los casos de uso en función del número de transacciones.

Atendiendo a la complejidad de cada caso de uso es necesario considerar el número de transacciones necesarias, tanto en el escenario básico del mismo como en los escenarios alternativos. En función de dicho número de transacciones, un caso de uso puede ser simple, si tiene 3 o menos transacciones, medio, si oscila entre 4 y 7 transacciones, y complejo, si supera las 7 transacciones.

TIPO DE CASO DE USO	NÚMERO DE TRANSACCIONES	PESO
Simple	3 o menos	1
Medio	de 4 a 7	2
Complejo	7 o más	3

Tabla 98: Complejidad Número Transacciones Casos de Uso

- Realizar el cálculo de los Puntos Caso de Uso no Ajustados.

Los UUCP resultantes se obtienen sumando el total de los pesos calculados, tanto para la complejidad de las interacciones, como la complejidad de los casos de uso en función de sus transacciones:

UUCP = pesos Actores + pesos Casos de Uso

10.2.1.2. Cálculo de los Factores de Ajuste

Para ajustar los UUCP obtenidos en los pasos anteriores, es necesario definir una serie de factores de ajuste, en este caso concreto dos: factores técnicos (TCF) y factores de entorno (EF).

- Factores Técnicos (TCF):

Para realizar el ajuste de los aspectos técnicos de un proyecto se han definido un total de 13 factores a los que se les asignará un valor en función del impacto que cada uno de estos factores tengan sobre el sistema a analizar. Los valores que cada factor puede tomar oscila entre 0 y 5, significando 0 que el factor tiene impacto nulo sobre el sistema y el 5 que resulta un factor crucial para el mismo.

Una vez asignados los valores de influencia de cada factor sobre el proyecto, se procederá a obtener el resultado final de cada factor, realizando el producto entre el peso del factor por el grado de influencia asignado.

Por último se aplicará la fórmula siguiente para obtener los TCF totales, donde el sumatorio será la suma de todos los productos de los factores:

$$TCF = 0{,}6 + (0{,}01 \times \sum_{n=1}^{13} (\text{peso}_n \times \text{influencia}_n)$$

La tabla que recoge todo lo mencionado es la siguiente:

FACTOR	DESCRIPCIÓN	PESO	INFLUENCIA	RESULTADO
R1	Sistema Distribuido	2	N1	R1=2xN1
R2	Objetivos de rendimiento	2	N2	R2=2 xN2
R3	Eficiencia respecto al usuario final	1	N3	R3=1 xN3
R4	Procesamiento complejo	1	N4	R4=1 xN4
R5	Código reutilizable	1	N5	R5=1 xN5
R6	Instalación sencilla	0,5	N6	R6=0,5 xN6
R7	Fácil utilización	0,5	N7	R7=0,5 xN7
R8	Portabilidad	2	N8	R8=2 xN8
R9	Fácil de cambiar	1	N9	R9=1 xN9
R10	Uso concurrente	1	N10	R10=1 xN10
R11	Características de seguridad	1	N11	R11=1 xN11
R12	Accesible por terceros	1	N12	R12=1 xN12
R13	Se requiere formación especial	1	N13	R13=1 xN13

Tabla 99: Factores técnicos de ajuste

- Factores de Entorno (EF):

Del mismo modo que se consideraron anteriormente los factores técnicos del sistema para realizar un ajuste de los UUCP, se han definido un total de 8 factores concernientes al entorno, que deberán ser ajustados igualmente en función de cómo afecten al proyecto.

Igual que en los factores técnicos, se deberá asignar un valor de influencia entre 0 y 5, siendo 0 un impacto nulo y 5 máximo sobre el sistema. Una vez asignados todos los valores para la influencia, se obtendrá el valor de cada EF realizando el producto entre el peso asignado al factor y la influencia.

Por último, se aplicará la siguiente fórmula para obtener los EF totales, siendo el sumatorio la suma de todos los productos anteriormente calculados:

$$EF = 1{,}4 + (-0{,}03 \times \sum_{n=1}^{8} (\text{peso}_n \times \text{influencia}_n)$$

La tabla que recoge lo comentado en este punto sobre los EF es la siguiente:

FACTOR	DESCRIPCIÓN	PESO	INFLUENCIA	RESULTADO
R1	Familiar con RUP	1,5	N1	R1=1,5xN1
R2	Experiencia en la aplicación	0,5	N2	R2=0,5xN2
R3	Experiencia en orientación a objetos	1	N3	R3=1xN3
R4	Capacidades de análisis	0,5	N4	R4=0,5xN4
R5	Motivación	1	N5	R5=1xN5
R6	Requisitos estables	2	N6	R6=2xN6
R7	Trabajadores a tiempo parcial	-1	N7	R7=-1xN7
R8	Lenguaje complejo	-1	N8	R8=-1xN8

Tabla 100: Factores de ajuste de entorno

10.2.1.3. Cálculo de los Puntos Caso de Uso Ajustados (UCP)

Una vez calculados los diferentes factores de ajuste en función de las especificaciones del proyecto y del equipo de trabajo que participará en el desarrollo del mismo, se obtendrán los Puntos Caso de Uso ajustados (UCP). Para realizar esto se utilizaran tanto los Puntos Caso de Uso sin Ajustar (UUCP), como los Factores Técnicos (TCF) y los Factores de Entorno (EF), aplicando la siguiente expresión:

$$UCP = UUCP \times TCF \times EF$$

Con este último cálculo, se habrán obtenido los Puntos de Casos de Uso ya ajustados.

10.2.1.4. Estimación del Esfuerzo

Una vez calculados los Puntos Caso de Uso ajustados se puede estimar el esfuerzo necesario para llevar a cabo el proyecto. Para obtener dicho esfuerzo es necesario aplicar la siguiente expresión:

$$Esfuerzo = UCP \times Factor\ de\ Productividad$$

Para la elección del factor de ajuste existen varias posibilidades:

- Utilizar el método originario de los Puntos Función.
- Karner sugiere utilizar un factor de 20 horas/hombre por cada Punto de Caso de Uso (UCP).
- Barnerjee propone utilizar un rango de entre 15 y 30 horas por UCP.

- Scheider y Winters sugiere un refinamiento en los factores de entorno (EF), realizando la suma de los factores, entre el R1 y el R6 cuyo valor sea inferior a 3 y los factores entre R7 y R8 superiores a 3. Una vez obtenido el resultado se aplicaría lo indicado en la siguiente tabla.

SUMA TOTAL	FACTOR DE AJUSTE
≤2	20 horas/hombre
≤4	28 horas/hombre
≥5	36 horas/hombre, considerar replantear el proyecto

Tabla 101: Factor ajuste esfuerzo

Por último indicar que este método de estimación del esfuerzo no atiende a todas las fases del ciclo de vida del proyecto, tan sólo se ajusta a la fase de codificación del mismo, lo que supone de media el 40% del esfuerzo total. El resto se distribuye entre el análisis (10%), el diseño (20%), las pruebas (15%) y la sobrecarga (15%).

10.2.2. Cálculo de los costes de desarrollo del software

A través del programa Enterprise Architect [112], en el cual se han realizado los diagramas de casos de uso, se ha establecido cuáles eran los valores en cuanto a la dificultad de cada caso de uso, atendiendo, tal y como marca la métrica, al número de transacciones de las que consta. En cuanto a la complejidad entre los usuarios y el sistema, se ha establecido como compleja, puesto que dicha interacción se hace a través de una interfaz. Para el resto de parámetros se han dejado los valores por defecto que el propio programa facilita.

El informe quedaría, finalmente, de la siguiente manera:

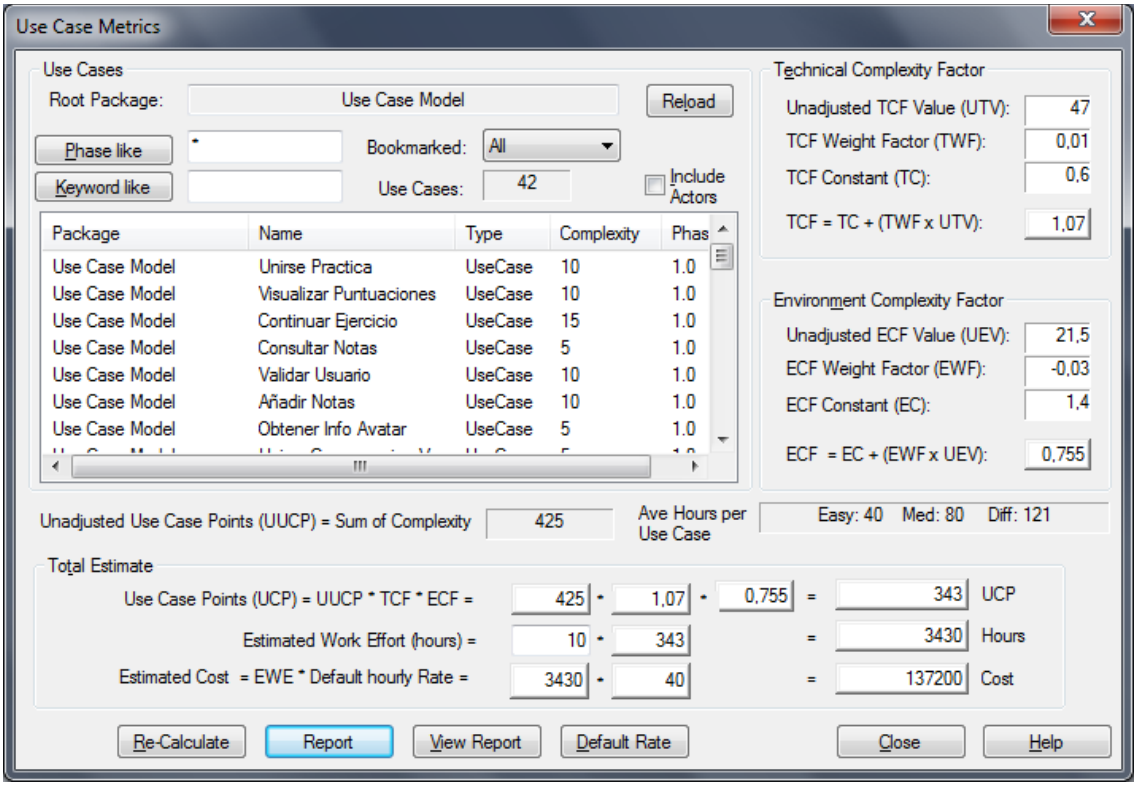

Imagen 86: Informe métricas de Casos de Uso

Como se comentó en el apartado dedicado a explicar esta métrica, el coste aquí especificado se ajusta únicamente a la etapa de codificación, que sería un 40% del total. Teniendo en cuenta los porcentajes asignados a cada una de las demás fases, el coste final sería el mostrado en las siguientes tablas:

FASE	PORCENTAJE	TOTAL
Análisis	10%	343 horas
Diseño	20%	686 horas
Codificación	40%	3.430 horas
Pruebas	15%	515 horas
Sobrecarga	15%	515 horas
TOTAL		5.488 horas

Tabla 102: Coste total en tiempo para el desarrollo del software

El coste total en horas para el desarrollo del software sería **5.488 (cinco mil cuatrocientas ochenta y ocho) horas**. Sería aproximadamente 34,3 meses.

En cuanto al aspecto monetario:

FASE	PORCENTAJE	TOTAL
Análisis	10%	34.300 €
Diseño	20%	68.600 €
Codificación	40%	137.200 €
Pruebas	15%	51.450 €
Sobrecarga	15%	51.450 €
TOTAL		343.000 €

Tabla 103: Coste monetario total para el desarrollo del software

El coste final sería, por tanto, **343.000 €** (**trescientos cuarenta y tres mil euros)**.

11. DEFINICIONES

- **API**: Una interfaz de programación de aplicaciones o API (del inglés Application Programming Interface) es el conjunto de funciones y procedimientos (o métodos, si se refiere a programación orientada a objetos) que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.

- **Aplicación nativa:** Aquella que se ejecuta directamente en la máquina del cliente, sin necesidad de un servidor.

- **Aplicación Web:** En la ingeniería software se denomina aplicación web a aquellas aplicaciones que los usuarios pueden utilizar accediendo a un servidor web a través de Internet o de una intranet mediante un navegador.

- **Arpanet**: La red de computadoras ARPANET (Advanced Research Projects Agency Network) fue creada por encargo del Departamento de Defensa de los Estados Unidos como medio de comunicación para los diferentes organismos del país. En ella se probaron las teorías y el software en los que está basado Internet.

- **Asíncrono:** Tipo de comunicación que envía datos usando control del flujo sin necesidad de sincronizar entre un terminal origen y un terminal destino.

- **Avatar**: Ser virtual que constituye la representación de un ser humano dentro de un mundo simulado, por ejemplo un juego. Habitualmente puede "personalizarse" dicho avatar para que se identifique el jugador con el personaje.

- **Back-end**: Es la parte del software que procesa la entrada desde el front-end, el cual es el que interactúa con el usuario. La idea general es que el front-end sea el responsable de recolectar los datos de entrada del usuario, que pueden ser de muchas y variadas formas, y procesarlas de una manera conforme a la especificación que el back-end pueda usar.

- **Background**: Se dice que una aplicación funciona "en background" cuando está trabajando sin afectar a la actividad del usuario.

- **Badget**: Pequeño banner que se suele incorporar en los con carácter de apoyo o adhesión.

- **Biblioteca**: Véase *Librería*.

- **Bot**: Un bot (diminutivo de *robot*) es un programa informático que realiza funciones muy diversas, imitando el comportamiento de un humano.

- **Brainstorming**: La lluvia de ideas o brainstorming, también denominada tormenta de ideas, es una herramienta de trabajo grupal que facilita el surgimiento de nuevas ideas sobre un tema o problema determinado. La lluvia de ideas es una técnica de grupo para generar ideas originales en un ambiente relajado.

- **Cliente**: El cliente es una aplicación informática que se utiliza para acceder a los servicios que ofrece un servidor, normalmente a través de una red de telecomunicaciones.

- **Commodore 64**: Ordenador doméstico de 8 bits lanzado por Commodore International en agosto de 1982, presentando 64 kilobytes (65,536 bytes) de RAM y gráficos y sonido muy por encima de otros equipos contemporáneos.

- **CRM**: Siglas de Customer Relationship Management. Son la metodología, la información y los procesos, que permiten a una empresa administrar sus contactos con los clientes de una forma organizada, necesarios para construir una relación entre una compañía y sus clientes.

- **CSS:** Siglas en inglés de Cascading Style Sheets. Conjunto de instrucciones HTML que definen la apariencia de uno o más elementos de un conjunto de páginas web con el objetivo de uniformizar su diseño.

- **Data Warehouse**: Un almacén de datos (del inglés data warehouse) es una colección de datos orientada a un determinado ámbito (empresa, organización, etc.), integrado, no volátil y variable en el tiempo, que ayuda a la toma de decisiones en la entidad en la que se utiliza.

- **DOM**: El Document Object Model, abreviado DOM, es esencialmente una interfaz de programación de aplicaciones que proporciona un conjunto estándar de objetos para representar documentos HTML y XML, un modelo estándar sobre cómo pueden combinarse dichos objetos, y una interfaz estándar para acceder a ellos y manipularlos.

- **E-Commerce**: El comercio electrónico, también conocido como e-commerce, consiste en la compra y venta de productos o de servicios a través de medios electrónicos, tales como el Internet y otras redes de ordenadores.

- **Erlang**: Erlang es un lenguaje de programación concurrente y un sistema de ejecución que incluye una máquina virtual y bibliotecas.

- **ESB**: consiste en un combinado de arquitectura de software que proporciona servicios fundamentales para arquitecturas complejas a través de un sistema de mensajes (el bus) basado en las normas y que responde a eventos. Los desarrolladores normalmente implementan un ESB utilizando tecnologías de productos de infraestructura de middleware que se basan en normas reconocidas.

- **Framework**: Un framework, en el desarrollo de software, es una estructura de soporte definida, mediante la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros software para ayudar a desarrollar y unir los diferentes componentes de un proyecto. Representa una arquitectura de software que modela las relaciones generales de las entidades del dominio. Provee una estructura y una metodología de trabajo la cual extiende o utiliza las aplicaciones del dominio.

- **Freeware**: Tipo de software que se distribuye sin costo, disponible para su uso y por tiempo ilimitado.

- **FTP**: siglas en inglés de File Transfer Protocol (Protocolo de Transferencia de Archivos). Es un protocolo de red para la transferencia de archivos entre sistemas conectados a una red TCP, basado en la arquitectura cliente-servidor. Desde un equipo cliente se puede conectar a un servidor para descargar archivos desde él o para enviarle archivos, independientemente del sistema operativo utilizado en cada equipo.

- **G.I.S.**: Un **Sistema** de Información Geográfica (SIG o GIS, en su acrónimo inglés) es una integración organizada de hardware, software y datos geográficos diseñado para capturar, almacenar, manipular, analizar y desplegar en todas sus formas la información geográficamente referenciada con el fin de resolver problemas complejos de planificación y gestión.

- **Google AdSense**: **AdSense** es un sistema de publicidad ideado por Google. Estos anuncios están administrados por Google y generan ingresos basándose en los clicks de los visitantes de la página y en las visualizaciones de la misma. Google utiliza su tecnología de búsqueda para incrustar anuncios según el contenido de la página web que se está visitando, la localización geográfica del usuario (mediante el ip), y otros datos como historia de búsqueda previa en Google o las páginas visitadas por el usuario, sus cookies, duración de la sesión, sistema operativo, browser utilizado, etc.

- **Google AdWords**: Google AdWords es el método que utiliza google para hacer publicidad patrocinada. Son anuncios que se muestran de forma relevante en los resultados de la búsqueda del usuario, en función de la consulta hecha por el usuario.

- **HTML**: Siglas en inglés de siglas de HyperText Markup Language.

- **HTTP**: Protocolo de comunicaciones usado en cada transacción de la Web (WWW), es decir este protocolo define la comunicación entre el software navegador de internet y el software servidor que publica las páginas consultadas.

- **JavaScript**: Lenguaje de programación interpretado, utilizado principalmente en páginas web para actualizar el contenido de la misma de manera dinámica.

- **Juego de role**: Juego en el que, tal como indica su nombre, uno o más jugadores desempeñan un determinado role, papel o personalidad.

- **Librería**: Conjunto de subprogramas utilizados para desarrollar software. Las bibliotecas contienen código y datos, que proporcionan servicios a programas independientes, es decir, pasan a formar parte de éstos.

- **Máquina virtual**: Una máquina virtual es un software que emula a un ordenador y puede ejecutar programas como si fuese un ordenador real.

- **Microformato**: Forma simple de agregar significado semántico a un contenido legible por el humano y que para la máquina es sólo texto plano. Están ideados para ser usadas en páginas web que usen HTML o XHTML, de manera tal de que la información pueda ser indexada, guardada, referenciada, reusada o combinada.

- **MMORPG**: Los juegos de role multijugador masivos en línea (o MMORPGs, del inglés massively multiplayer online role-playing games), son videojuegos de role que permiten a miles de jugadores introducirse en un mundo virtual de forma simultánea a través de Internet, e interactuar entre ellos.

- **Navegador**: Un navegador, o navegador web (del inglés, web browser) es un programa que permite visualizar la información que contiene una página web.

- **Objeto**: Unidad que en tiempo de ejecución realiza las tareas de un programa.

- **OKBC**: Open Knowledge Base Connectivity (OKBC) es un protocolo y una API para el acceso al conocimiento en sistemas de representación del conocimiento como ontologías o bases de datos relacionales basadas en objetos.

- **Open Source**: Traducido código abierto, es el término con el que se conoce al software distribuido y desarrollado libremente.

- **Proceso batch**: Un proceso batch (por lotes) es el que realiza automáticamente una secuencia de acciones según se listan en archivos con extensión .bat.

- **Recurso**: Objeto de datos de red o servicio que puede identificarse por un URL. Se llama así a la información que se encuentra en Internet, ofrecida por los servidores.

- **REST**: La Transferencia de Estado Representacional (Representational State Transfer) o REST es una técnica de arquitectura software para sistemas hipermedia distribuidos como la World Wide Web. Se basa en la existencia de *recursos* (elementos de información), que pueden ser accedidos utilizando un identificador global. Para manipular estos recursos, los componentes de la red (clientes y servidores) se comunican a través de un interfaz estándar (HTTP) e intercambian representaciones de dichos recursos.

- **RSS**: Es una familia de formatos para documentos utilizados para publicar contenidos que se actualizan frecuentemente en el Internet, por ejemplo entradas en un blog o encabezados de noticias.

- **Servidor**: Ordenador o programa que proporciona un determinado servicio a otro programa denominado cliente y que acostumbra a ejecutarse en otro ordenador.

- **Screen scraping:** Screen scraping es el nombre en inglés de una técnica de programación que consiste en tomar una presentación de una información (normalmente texto, aunque puede incluir información gráfica) para, mediante ingeniería inversa, extraer los datos que dieron lugar a esa presentación.

- **Sindicación (web)**: Redifusión web (o sindicación web) es el reenvío de contenidos desde una fuente original (sitio web de origen) hasta otro sitio web de destino (receptor) que a su vez se convierte en emisor puesto que pone a disposición de sus usuarios los contenidos a los que en un principio sólo podían tener acceso los usuarios del sitio web de origen.

- **SOAP**: Siglas de Simple Object Access Protocol. Es un protocolo para el intercambio de mensajes que utiliza XML y es independiente de la plataforma o sistema operativo, utilizado para codificar información en los diálogos de los Servicios Web.

- **Socket**: Objeto de software utilizado por un cliente para conectarse a un servidor. Los componentes básicos incluyen el número de puerto y la dirección de red del host local.

- **Start-up**: Una compañía startup o start-up es un negocio con una historia de funcionamiento limitada, pero con grandes posibilidades de crecimiento. Generalmente son llevadas por emprendedores que levantan compañías que aportan positivamente al desarrollo de sus países y de ellos mismos, al promover prácticas asociadas a la innovación, desarrollo de tecnologías, empleos de calidad, mejor distribución de la riqueza, etc.

- **SWRL**: Siglas de Semantic Web Rule Language. Es un lenguaje de reglas para la Web Semántica, combinando los lenguajes OWL con RML (Rule Markup Language).

- **Telnet**: Telnet (TELecommunication NETwork) es el nombre de un protocolo de red (y del programa informático que implementa el cliente), que sirve para acceder mediante una red a otra máquina, para manejarla remotamente como si se estuviera sentado delante de ella. Para que la conexión funcione, como en todos los servicios de Internet, la máquina a la que se acceda debe tener un programa especial que reciba y gestione las conexiones.

- **Thread**: Un hilo de ejecución, o thread, es una característica que permite a una aplicación realizar varias tareas a la vez (concurrentemente). Los distintos hilos de

ejecución comparten una serie de recursos tales como el espacio de memoria, los archivos abiertos, situación de autenticación, etc.

- **URI**: Uniform Resource Identifier, identificador uniforme de recurso, definido en RFC 2396. Un URI es una cadena corta de caracteres que identifica inequívocamente un recurso (servicio, página, documento, dirección de correo electrónico, enciclopedia, etc.). Normalmente estos recursos son accesibles en una red o sistema.

- **URL**: Esquema de direcciones de Internet que permite localizar recursos en la Red.

- **VRLM**: Siglas de Virtual Reality Modeling Language. Formato de descripción de entornos en tres dimensiones que permite dibujar dichos entornos y navegar por ellos con el ratón. Con este lenguaje se puede entrar en "mundos virtuales", recorrerlos e interactuar con los objetos contenidos en éstos.

- **W3C**: El World Wide Web Consortium, abreviado W3C, es un consorcio internacional que produce recomendaciones para la World Wide Web.

- **Web Service**: Un servicio web (en inglés, *Web service*) es un conjunto de protocolos y estándares que sirven para intercambiar datos entre aplicaciones. Distintas aplicaciones de software desarrolladas en lenguajes de programación diferentes, y ejecutadas sobre cualquier plataforma, pueden utilizar los servicios web para intercambiar datos en redes de ordenadores como Internet.

- **Widget**: Término genérico para referirse a los elementos de una interfaz gráfica con el usuario, como botones, barras de desplazamiento, campos de entra de texto, etc.

- **WSDL**: siglas de Web Services Description Language (Lenguaje de Descripción de Servicios), un formato XML que se utiliza para describir servicios Web. Está basado en XML y describe la forma de comunicación, es decir, los requisitos del protocolo y los formatos de los mensajes necesarios para interactuar con los servicios listados en su catálogo. Las operaciones y mensajes que soporta se describen en abstracto y se ligan después al protocolo concreto de red y al formato del mensaje.

- **XHTML**: Siglas en inglés de eXtensible Hypertext Markup Language.

- **XML**: Siglas de eXtensible Markup Language. Formato estándar para el intercambio de datos basado en archivos de texto plano con una estructura de tags.

- **XMLHttpRequest**: Siglas en inglés de Extensible Markup Language / Hypertext Transfer Protocol.

12. BIBLIOGRAFÍA

[1]. Raymond Yee. Pro Web 2.0 Mashups. Remixing Data and Web Services. Apress. 2008.

[2]. Duane Merrill. Mashups: The new breed of Web app. An introduction to mashups. 10/2006. Última visita 09/2009. http://www.ibm.com/developerworks/xml/library/x-mashups.html

[3]. Shaw. Business Mashups - mashup applications for and by the business. 10/2008. Última visita 10/2009. http://businessmashup.blogspot.com/2007/09/do-we-need-business-mashups.html

[4]. John Markoff. Technology: Marrying Maps to data for a new Web service. 07/2005. Última visita 09/2009. http://query.nytimes.com/gst/fullpage.html?res=9E05EFD81130F93BA25754C0A9639C8B63

[5]. Andreas Thor, David Aumueller, Erhard Rahm. Data Integration Support for Mashups. 2007. Última visita 09/2009. http://dbs.uni-leipzig.de/file/IIWeb2007_final.pdf

[6]. John Curpi, Chris Warner. Enterprise Mashups Part I: Bringing SOA to people. 05/2008. Última visita 09/2009. http://www.soamag.com/I18/0508-1.asp

[7]. John Curpi, Chris Warner. Enterprise Mashups Part II: Why SOA Architects Should Care. 08/2008. Última visita 09/2009. http://www.soamag.com/I21/0808-1.asp

[8]. Holt Adams. Mashup business scenarios and patterns: part 1. 02/2009. Última visita 09/2009. http://www.ibm.com/developerworks/lotus/library/mashups-patterns-pt1/

[9]. Aaron Bohannon. Building Secure Web Mashups. 07/2008. Última visita 09/2009. http://www.cis.upenn.edu/~bohannon/mashups.pdf

[10]. Javier Eguíluz Pérez. Introducción a Ajax. 06/2008. Última visita 09/2009. http://www.librosweb.es/ajax/

[11]. Jesse James. Ajax: A New Approach to Web Applications. 02/2005. Última visita 09/2009. http://adaptivepath.com/ideas/essays/archives/000385.php

[12]. Hao Hen. What is Service-Oriented Architecture. 09/2003. Última visita 09/2009. http://www.xml.com/pub/a/ws/2003/09/30/soa.html

[13]. Mike Gebhardt. Serviceorientierte vs. Event-basierte Architekturen. Johann Wolfgang Goethe – Universität. 2005.

[14]. Julián Astorga Campos, Juan Luis Quirós Venegas. Un enfoque teórico e intuitivo a la arquitectura orientada a servicios. 2007. Última visita 09/2009. http://www.di-mare.com/adolfo/cursos/2007-2/pp-SOA.pdf

[15]. Julio Alba. SOA. Arquitectura Orientada al Servicio. 03/2008. Última visita 09/2009.

http://www.coit.es/publicaciones/bit/bit167/quees.pdf

[16]. Jason Bloomberg. When Not to Use an SOA. 02/2004. Última visita 09/2009. http://www.zapthink.com/report.html?id=zapflash-02162004

[17]. Javier Cámara. Recomendaciones para la adopción de SOA. Última visita 09/2009. http://www.isa.us.es/downloads/proceedings/0212.pdf

[18]. Roger L. Costello. Building Web Services, the REST Way. Última visita 09/2009. http://www.xfront.com/REST-Web-Services.html

[19]. Paul Prescod. REST and the Real World. 02/2002. Última visita 09/2009. http://www.xml.com/pub/a/ws/2002/02/20/rest.html

[20]. Joshua Tauberer. What is RDF. 07/2006. Última vi/sita 09/2009. http://www.xml.com/pub/a/2001/01/24/rdf.html

[21]. Tim Bray. Qué es RDF. 02/2001. Última visita 09/2009. http://www.malditainternet.com/node/96

[22]. Félix Moral. Aspectos psicosociales de la comunicación y de las relaciones personales en Internet. 2001. Última visita 09/2009. http://www.raco.cat/index.php/AnuarioPsicologia/article/view/61665/96249

[23]. Cristina Botella, Rosa Baños, Azucena García-Palacios, Soledad Quero, Verónica Guillén. La utilización de nuevas tecnologías de la información y la comunicación en psicología clínica. 03/2007. Última visita 09/2009. http://www.uoc.edu/uocpapers/4/dt/esp/botella.pdf

[24]. Marco Antonio Pedraza**.** Los entornos virtuales de enseñanza-aprendizaje. Propuesta pedagógica. 09/2003. Última visita 09/2009. http://www.monografias.com/trabajos14/entornosvirt/entornosvirt.shtml

[25]. Javier Onrubia. Aprender y enseñar en entorno virtuales: actividad conjunta, ayuda pedagógica y construcción del conocimiento. 01/2005. Última visita 09/2009. http://www.um.es/ead/red/M2/conferencia_onrubia.pdf

[26]. José R. Hilera, Salvador Otón, Javier Martínez. Aplicación de la Realidad Virtual en la enseñanza a través de Internet. Última visita 09/2009. http://www.ucm.es/info/multidoc/multidoc/revista/num8/hilera-oton.html

[27]. Amy Susan Bruckman. Moose Crossing: Construction, Community and Learning in a Networked Virtual World for Kids. 06/1997. Última visita 09/2009. http://dspace.mit.edu/handle/1721.1/33821

[28]. Greg Boss, Padma Malladi, Dennis Quan, Linda Legregni, Harold Hall. Cloud Computing. 10/2007. Última visita 09/2009. http://www.scribd.com/doc/2911680/Cloud-computing-wp-final-8Oct

[29]. Peter Mell, Tim Grance. Draft NIST Working Definition of Cloud Computing. 07/2009. Última visita 09/2009. http://csrc.nist.gov/groups/SNS/cloud-computing/cloud-def-v15.doc

[30]. Cloud Computing: Las TI como servicio. 06/2008. Última visita 09/2009. http://www.networkworld.es/Articulo.aspx?ida=191003&seccion=recursos&AspxAutoDetectCookieSupport=1

[31]. Stephanie Falla Aroche. Cloud Computing: nueva era de desarrollo. 11/2008. Última visita 09/2009.

http://www.maestrosdelweb.com/editorial/cloud-computing-nueva-era-de-desarrollo/

[32]. Cloud Computing 3: Arquitectura. Ventajas y Desventajas. 12/2008. Última visita 09/2009.
http://trycatch.lacoctelera.net/post/2008/12/18/cloud-computing-3-arquitectura-ventajas-y-desventajas

[33]. Ana María Gil. Software as a Service. 01/2008. Última visita 09/2009.
http://www.noticias.com/opinion/software-as-service-4h3.html

[34]. Praising Gaw. What's the Difference Between Cloud Computing and SaaS? 07/2008. Última visita 09/2009.
http://cloudcomputing.sys-con.com/node/612033

[35]. Parallels. Especialistas en software de virtualización y automatización. SaaS – Software as a Service. Última visita 09/2009.
http://www.parallels.com/es/products/saas/

[36]. Alfonso Gutiérrez. ¿Qué es PaaS? 02/2009. Última visita 09/2009.
http://knol.google.com/k/alfonso-gutirrez/qu-es-paas/294ccfj1s1mhd/4#

[37]. Christopher Keene. What is Platform as a Service (PaaS)? 03/2009. Última visita 09/2009.
http://www.keeneview.com/2009/03/what-is-platform-as-service-paas.html

[38]. What is Infrastructure as a Service (IaaS)? 06/2009. Última visita 09/2009.
http://searchcloudcomputing.techtarget.com/sDefinition/0,,sid201_gci1358983,00.html

[39]. Modern software architecture definitions. Última visita 09/2009.
http://www.sei.cmu.edu/architecture/start/moderndefs.cfm

[40]. Bibliographic Software Architecture Definitions. Última visita 09/2009.
http://www.sei.cmu.edu/architecture/start/bibliographicdefs.cfm

[41]. Len Bass, Paul Clements, Rick Kazman. Software Architecture in Practice. Addison-Wesley Professional. 1997.

[42]. Josep Casanovas. Usabilidad y arquitectura del software. 09/2004. Última visita 09/2009.
http://www.desarrolloweb.com/articulos/1622.php

[43]. Adrián Lasso. Arquitectura de Software. Última visita 10/2009.
http://www.scribd.com/doc/210452/Arquitectura-de-Software-Adrian-Lasso

[44]. Carlos Billy Reynoso. Introducción a la arquitectura software. 03/2004. Última visita 10/2009.
http://www.willydev.net/descargas/prev/IntroArq.pdf

[45]. Paul Clements, Linda Northrop. Software architecture: An executive overview. 02/1996. Última visita 10/2009.
http://www.sei.cmu.edu/reports/96tr003.pdf

[46]. David Garlan. Software Architecture: A Roadmap. Última visita 10/2009.
http://www.cs.ucl.ac.uk/staff/A.Finkelstein/fose/finalgarlan.pdf

[47]. An Introduction to Software Architecture. David Garlan and Mary Shaw. 01/1994. Última visita 10/2009.
http://www.cs.cmu.edu/afs/cs/project/vit/ftp/pdf/intro_softarch.pdf

[48]. Enciclopedia Microsoft® Encarta® Online. Arquitectura cliente-servidor. 2009. Última visita 10/2009.

http://es.encarta.msn.com/encyclopedia_761556371/Arquitectura_cliente_servidor.html

[49]. Ernesto Bascón Pantoja. El patrón de diseño Modelo – Vista – Controlador y su implementación en Java Swing. 12/2004. Última visita 10/2009. http://www.ucbcba.edu.bo/Publicaciones/revistas/actanova/documentos/v2n4/v2.n4.bascon.pdf

[50]. Guillermo González, Mauricio Machuca. Tecnologías Web: Modelo Vista Controlador (MVC). 2008. Última visita 10/2009. http://www.opensolutions.com.py/~ggonzalez/fpuna/is2/files/06_TECNOLOGIAS_WEB_MVC.pdf

[51]. Sandra Victoria Hurtado Gil. Representación de la arquitectura de software usando UML. Última visita 10/2009. http://bibliotecadigital.icesi.edu.co/biblioteca_digital/bitstream/item/384/1/shurtado_repres-uml.pdf

[52]. Juan Manuel Cueva Lovelle. Introducción a UML. 10/1999. Última visita 10/2009. http://gidis.ing.unlpam.edu.ar/downloads/pdfs/IntroduccionUML.PDF

[53]. Francisco Mora. UML: Lenguaje Unificado de Modelado. 2002. Última visita 10/2009. http://www.itescam.edu.mx/principal/sylabus/fpdb/recursos/r43098.PDF

[54]. Francisco Javier Honrubia López. Introducción a las ontologías. Última visita 10/2009. http://www.info-ab.uclm.es/asignaturas/42551/trabajosAnteriores/Trabajo-Ontologias.pdf

[55]. Van Heijst, Schereiber, Wielinga. Using Explicit Ontologies in KBS Development. 1996. Última visita 10/2009. http://hcs.science.uva.nl/usr/gertjan/postscript/ijhcs-HSW.ps.gz

[56]. Nicola Guarino. Understanding, Building, and Using Ontologies. 10/1996. Última visita 10/2009. http://ksi.cpsc.ucalgary.ca/KAW/KAW96/guarino/guarino.html

[57]. R. Neches. Enabling technology for knowledge sharing. 1991. Última visita 10/2009. http://tomgruber.org/writing/AIMag12-03-004.pdf

[58]. Vicente Guerrero Bore, Adolfo Lozano Tello. Vínculos entre las Ontologías y la Biblioteconomía y Documentación. 1999. Última visita 10/2009. http://dialnet.unirioja.es/servlet/fichero_articulo?codigo=1300079&orden=0

[59]. Jian Quin, Stephen Paling. Converting a controlled vocabulary into an ontology: the case of GEM. 01/2001. Última visita 10/2009. http://informationr.net/ir/6-2/paper94.html

[60]. Antonio García Jiménez. Instrumentos de representación del conocimiento. 2004. Última visita 10/2009. http://revistas.um.es/analesdoc/article/viewFile/1691/1741

[61]. Natalya F. Noy, Deborah L. McGuinness. Ontology development 101: A guide to creating your first ontology. 09/2005. Última visita 10/2009. http://protege.stanford.edu/publications/ontology_development/ontology101-es.pdf

[62]. Asunción Gómez-Pérez. A survey on ontology tools. 05/2002. Última visita 10/2009.
http://userpages.uni-koblenz.de/~sure/publications/OntoWeb_Del_1-3.pdf

[63]. Miguel Rodríguez Artacho. Una ontología básica para una asignatura teórico-práctica. 07/2000. Última visita 10/2009.
http://sensei.lsi.uned.es/~miguel/tesis/node25.html

[64]. W3C. Lenguaje de Ontologías Web (OWL). Vista General. 02/2004. Última visita 10/2009.
http://www.w3.org/2007/09/OWL-Overview-es.html

[65]. Biopax. Página principal del grupo Biopax. Última visita 10/2009.
http://www.biopax.org/index.html

[66]. Dublin Core. Página principal de la iniciativa de metadatos Dublin Core. Última visita 10/2009.
http://dublincore.org/

[67]. Gellish. Página principal del proyecto Gellish. Última visita 10/2009.
http://sourceforge.net/apps/trac/gellish/wiki

[68]. WordNet. Página principal de la base de datos léxica WordNet. Última visita 10/2009.
http://wordnet.princeton.edu/

[69]. DAML.org. Ejemplo de DAML+OIL. Última visita 10/2009.
http://www.daml.org/2001/03/daml+oil-ex.daml

[70]. Yahoo Pipes. Página principal de la herramienta Yahoo Pipes. Última visita 10/2009.
http://pipes.yahoo.com/pipes/

[71]. IBM Mashup Center. Página principal de la herramienta IBM Mashup Center. Última visita 10/2009.
http://www-01.ibm.com/software/info/mashup-center/

[72]. JackBe Presto. Página principal de la herramienta JackBe Presto. Última visita 10/2009.
http://www.jackbe.com/products/

[73]. Protégé. Página principal de la herramienta Protégé. Última visita 10/2009.
http://protege.stanford.edu/

[74]. Ontolingua. Página principal de la herramienta Ontolingua. Última visita 10/2009.
http://www.ksl.stanford.edu/software/ontolingua/

[75]. A. J. Duineveld, R. Stoter, M. R. Weiden, B. Kenepa y V. R. Benjamins. WonderTools? A comparative study of ontological engineering tool. 06/2000. Última visita 10/2009.
http://hcs.science.uva.nl/wondertools/html/paper.htm

[76]. Asunción Gómez-Pérez, Mariano Fernández López, Oscar Corcho. Ontological Engineering. Springer. 2004.

[77]. OntoEdit. Página principal de la herramienta OntoEdit. Última visita 10/2009.
http://www.ontoknowledge.org/tools/ontoedit.shtml

[78]. OilEd. Página principal de la herramienta OilEd. Última visita 10/2009.
http://xml.coverpages.org/oilEdANn20001204.html

[79]. Institut de Robótica de la Universidad Valenciana. Estudio comparativo de soluciones para el desarrollo de ontologías. 09/2005. Última visita 10/2009. http://robotica.uv.es/~cicyt/doc_publicos/D2estadoartev1.pdf

[80]. Juan Pablo Palacios. Ontology Development Tools. 07/2004. Última visita 10/2009. http://sinbad.dit.upm.es/docencia/doctorado/curso0304/OntologyDevelopmentTools.pdf

[81]. Multiverse. Página principal de la herramienta Multiverse. Última visita 10/2009. http://www.multiverse.net/developer/index.jsp?cid=4&scid=0

[82]. Project Wonderland. Página principal del proyecto Wonderland. Última visita 10/2009. https://lg3d-wonderland.dev.java.net/

[83]. VR Space. Página principal de la herramienta VR Space. Última visita 10/2009. http://www.vrspace.org/

[84]. Vizard. Página principal de la herramienta Vizard. Última visita 10/2009. http://www.worldviz.com/products/vizard/index.html

[85]. HeroEngine. Página principal de la herramienta HeroEngine. Última visita 10/2009. http://www.heroengine.com/home

[86]. Google App Engine. Página principal del servicio Google App Engine. Última visita 10/2009. http://code.google.com/intl/es/App Engine/

[87]. Windows Azure. Página principal del servicio Windows Azure. Última visita 10/2009. http://www.microsoft.com/windowsazure/

[88]. Amazon EC2. Página principal del servicio Amazon EC2. Última visita 10/2009. http://aws.amazon.com/ec2/

[89]. Dmitry Sotnikov. Microsoft Azure vs. Amazon, Google, and VMware. 10/2008. Última visita 10/2009. http://cloudenterprise.info/2008/10/29/microsoft-azure-vs-amazon-google-and-vmware/

[90]. María Cristina Cicarelli. Procesos, estrategias y técnicas de aprendizaje. 10/2006. Última visita 10/2009. http://www.psicopedagogia.com/tecnicas-aprendizaje

[91]. Francisco de Asís Martín del Buey. Procesos, estrategias y técnicas de aprendizaje. Última visita 10/2009. http://www.profes.net/rep_documentos/Monograf/Aprendizaje.PDF

[92]. Adriana M. Meza Meza, Yara Elizabeth Pérez Guerrero, Berenice de la Barreda Bautista. Comunidades Virtuales de Aprendizaje como herramienta didáctica para el apoyo de la labor docente. 11/2002. Última visita 10/2009. http://www.funredes.org/mistica/castellano/ciberoteca/participantes/docuparti/esp_doc_72.html

[93]. Francisco Ignacio Revuelta Domínguez, Juan Carlos Pereña Moro, Giovanna Azuni, Miriam Herrerías Aonso. Las estrategias de enseñanza en entornos virtuales. Última visita 11/2009.
http://web.usal.es/~fird/docs/estrategias_de_ense%F1anza_espacios_virtuales.PDF

[94]. Germán Amaya Franky, José Antonio Martín Sánchez. Esquema de modelos de enseñanza. Última visita 11/2009.
http://noesis.usal.es/Documentos/ARTICULOS%20EDUCARE%202003/ESQUEMA%20DE%20MODELOS%20DE%20ENSE%D1ANZA.pdf

[95]. Modelos de enseñanza-aprendizaje. Repaso de los principales modelos de enseñanza y aprendizaje. Última visita 11/2009.
http://gcarvajalmodelos.wordpress.com/

[96]. Hugo Rosales, José Aldaña, Rafael Asenjo, Oswaldo Trelles. Adecuación de un modelo de enseñanza superior en un modelo de enseñanza virtual. Última visita 11/2009.
http://www.ateneonline.net/datos/67_03_rosales_hugo.pdf

[97]. Técnicas de aprendizaje. Sitio Web sobre la herramienta Knowledge Master para la creación de mapas conceptuales. Última visita 11/2009.
http://mail.udgvirtual.udg.mx/biblioteca/bitstream/123456789/462/1/KM-Recommend-esp.htm

[98]. Carles Dorado Perea. Estrategias de aprendizaje. 1997. Última visita 11/2009.
http://www.xtec.es/~cdorado/cdora1/esp/estrat.htm

[99]. Nils Pacherres Ganoza. Enfoques cualitativos y cuantitativos en las ciencias sociales. Última visita 11/2009.
http://www.monografias.com/trabajos32/enfoques-ciencias-sociales/enfoques-ciencias-sociales.shtml

[100]. Emelideth Valenzuela. Paradigma de evaluación comprensivista o alternativo. 2008. Última visita 11/2009.
http://coaprendices.blogspot.com/2008/03/paradigma-de-evaluacin-comprensivista-o.html

[101]. Apex Learning. Herramienta de aprendizaje virtual. Última visita 11/2009.
http://www.apexlearning.com/

[102]. ATutor. Herramienta de aprendizaje virtual. Última visita 11/2009.
http://www.atutor.ca/

[103]. OLAT. Herramienta de aprendizaje virtual. Última visita 11/2009.
http://www.olat.org/website/en/html/index.html

[104]. WebCT. Herramienta de aprendizaje virtual. Última visita 11/2009.
http://www.webct.com/

[105]. SPSS for Windows. Herramienta para el análisis estadístico. Última visita 11/2009.
http://www.spss.com/es/

[106]. 3DXplorer. Programa para la creación de mundos virtuales. Última visita 11/2009.
http://www.3dxplorer.com/static-v3/index.html

[107]. Adobe PhotoShop CS4. Programa para edición de imágenes. Última visita 11/2009.
http://www.adobe.com/es/products/photoshop/photoshop/

[108]. George Reese. Database Programming with JDBC and Java, Second Edition. Chapter 7: Distributed Application Architecture. Última visita 11/2009.
http://java.sun.com/developer/Books/jdbc/

[109]. Alfredo Goñi. Reutilización del Software, Patrones de Diseño. Última visita 11/2009.
http://siul02.si.ehu.es/~alfredo/iso/06Patrones.pdf

[110]. Mª Carmen García, Javier Garzás. El método de los puntos caso de uso (UCP).

[111]. Microsoft Project 2007. Herramienta de administración de proyectos.
http://office.microsoft.com/es-es/project/FX100487773082.aspx

[112]. Enterprise Architect. Herramienta de modelado de diagramas UML.
http://www.sparxsystems.com.au/

[113]. JDO. Java Data Objetcs. API para el almacenamiento de los datos.
http://db.apache.org/jdo/jdo_v_jpa.html

[114]. Uso de JDO con App Engine. Especificación del uso de JDO dada por Google.
http://code.google.com/intl/es/appengine/docs/java/datastore/usingjdo.html

ANEXO 1: ENCUESTA

ENCUESTA MUNDOS VIRTUALES EN LA ENSEÑANZA

Datos personales:

Sexo (H/M)		**Edad**	
Nivel de Estudio (Titulado, Estudiante, Postgrado)		**Trabaja mientras estudia (Si/No)**	
Carrera			

Encuesta:

1. ¿Le gustaría que le impartieran cursos utilizando mundos virtuales? (Si/No)	

2. Comentarios sobre gustos de la pregunta anterior si ha sido positiva o negativa

3. ¿Cómo de novedoso encuentra estudiar a través de entornos virtuales? (Muy novedoso 5 - 4 - 3 - 2 - 1 Poco novedoso)	

4. ¿Conoce algún método similar a éste? (Por favor, indicar cuál(es))

5. ¿Ha utilizado estas estrategias anteriormente? (Si/No)	

6. ¿Ha seguido algún curso de forma on-line? (Si/No)	

7. En caso afirmativo, ¿qué resultados obtuvo?

8. Comentarios