INCLUSIÓN DE SERVICIOS EN APLICACIONES BASADOS EN PATRONES DE USABILIDAD CASO UNDO/REDO

Tesista

M.Ing. Hernán D. MERLINO

Directores

Prof. Patricia Pesado (UNLP), Prof. Oscar DIESTE (UPM)

Co-Director

Prof. Ramón GARCÍA MARTÍNEZ (UNLP-UNLa)

TESIS PRESENTADA PARA OBTENER EL GRADO
DE
DOCTOR EN CIENCIAS INFORMÁTICAS

FACULTAD DE INFORMÁTICA
UNIVERSIDAD NACIONAL DE LA PLATA

Junio, 2014

RESUMEN

Los patrones de usabilidad son un aspecto central en el desarrollo de software, pues estos son los encargados de sentar las bases de un conjunto de principios validados y establecidos para la creación de una apropiada interfaz de usuario. En este sentido el esfuerzo por desarrollar un modelo de patrones de usabilidad esta justificado y permite sumar una instancia mas al proceso de automatización en el desarrollo de software. En esta tesis doctoral se propone un mecanismo que ha evolucionado desde los patrones hasta una arquitectura de usabilidad, detallando el proceso evolutivo que ha llevado el mismo, define en detalle el patrón de usabilidad UNDO/REDO y construye los cimientos para extender este modelo a otros patrones de usabilidad.

ABSTRACT

Usability patterns are a central aspect of software development, as these are responsible for laying the foundations of a set of validated and established principles for creating an appropriate user interface. In this sense the effort to develop a model of usability patterns is justified and can then add another instance to process automation in software. This PhD thesis proposes a mechanism that has evolved from an architecture patterns to usability, detailing the evolutionary process that has led it defines in detail the usability pattern UNDO/REDO and builds the foundation for extending this model to other usability patterns

DEDICATORIA

A Marisol por enseñarme el camino de la excelencia
y ser parte esencial de este logro académico.

A mi madre Graciela que siempre me ha apoyado para que mejore día a día.

A mi padre Luciano y a mi abuela María quienes ya no están.

A mi mentor y amigo Ramón.

A mis amigos Paola, Alejandro, Enrique, Pedro y Walter.

A mis alumnos, que todos los días me permiten que aprenda algo de ellos.

AGRADECIMIENTOS

A la Facultad de Informática de la Universidad Nacional de la Plata por acogerme con generosidad de *"alma mater"* para que pudiera llevar a cabo mis estudios de Doctorado en Ciencias Informáticas.

A la Licenciatura en Sistemas de la Universidad Nacional de Lanús por permitir desarrollarme como Profesor y darme en el Grupo de Investigación en Sistemas de Información un espacio donde desarrollar mi pasantía de investigación y desarrollo, proveyendo un estimulante ambiente de intercambio de ideas con otros tesistas de postgrado, dándome apoyo en todas las instancias del proceso para obtener el grado de Doctor.

A la Facultad de Ingeniería de la Universidad de Buenos Aires por permitir desarrollarme como auxiliar docente desde hace mas de una década y como profesor en estos últimos años.

Al Centro de Ingeniería del Software e Ingeniería del Conocimiento del Instituto Tecnológico de Buenos Aires por dar apoyo inicial en el desarrollo de mis estudios de postgrado.

Al Prof. Ramón García Martínez por dirigir mi trabajo de tesis con la inquebrantable dedicación del maestro y el afecto del amigo; sin cuyas cualidades, no hubiera sido posible culminar la presente obra.

Al Prof. Oscar Dieste por las sugerencias realizadas sobre el tema de mi tesis, siempre con la exactitud del científico y la calidez del docente de alma.

Al Prof. Patricia Pesado por su especial e incondicional apoyo y orientación en todo este proceso.

A mi compañera y amiga Paola Britos por su gran aporte en mi formación de postgrado.

A mi compañero de trabajo Enrique Fernández por su ayuda académica y sus valiosas sugerencias del desarrollo del presente trabajo.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

NOMENCLATURA

APIs	Application Programming Interface
BPEL	Business Process Execution Language
BPM	Business Process Management
BPPaaS	Programable PaaS
CCIF	Cloud Computing Foro de Interoperabilidad
CCUCDG	Cloud Computing Use Case Group
CPP	Clustered Platform Pattern
DMTF	Common Information Model
EC2	Elastic Compute Cloud
IaaS	Infraestructura como Servicio
IaaSA	Infrastructure-as-a-Service Aggregator
MCPP	Multiple CPP
MShPP	Multiple ShPP
OBDC	Object Data Base Connection
OCCIWG	Cloud Computing Open Interface Working Group
OFBiz	Apache Open For Business Project
PaaS	Plataforma como Servicio
PNR	Puntos de No Retorno
QoS	Calidad de Servicios
S3	Amazon Simple Storage
SaaS	Software como Servicio
ShPP	Shared Platform Pattern
SLA	Service Lavel Agreement
SOA	Service Oriented Architecture
SPP	Single Platform Pattern
TI	Tecnologías de la Información
ULC	Unidad Lógica de Cambio
XPDL	XML Process Definition Language
XSS	Cross-Site scripting

1. INTRODUCCIÓN

En este Capítulo se plantea el contexto de la tesis (sección 1.1), se establece su objetivo (sección 1.2), se presentan las publicaciones del tesista vinculadas a las investigaciones realizadas en el desarrollo de la tesis (sección 1.3) y se resume la estructura de la tesis (sección 1.4).

1.1. CONTEXTO DE LA TESIS

La usabilidad de software ha adquirido en los últimos años gran relevancia, ésto debido a la multiplicidad de dispositivos electrónicos que los usuarios de sistemas software tienen a su alcance. La imagen de una persona sentada frente a una pantalla y un teclado se ha convertido sólo en una de las tantas posibilidades que estas tienen de interactuar con un sistema software; la aparición de dispositivos móviles que son accesibles a un segmento creciente de usuarios que trabajan, se comunican y se distraen en su tiempo libre con éstos es cada vez mayor. Esto hace que la investigación y aportes en el campo de la usabilidad de software sean de importancia para el ámbito de la ingeniería de software.

Haciendo una breve reseña del marco histórico, los principios de usabilidad han derivado en el diseño y construcción de patrones de usabilidad debido a la necesidad de la ingeniería de software de formalizar los conocimientos adquiridos; estos patrones permiten que el desarrollo de software sea simple y predecible [Ferre et al, 2004] proporcionando a los artefactos de software un conjunto de características específicas de usabilidad [Ferre et al, 2003].

En la literatura referida a la usabilidad del software es posible reconocer un conjunto de patrones que son constantemente referenciados, a saber: Feedback, UNDO/REDO, Cancel, Form/Field Validation, Wizard, User Profile y Help [Juristo et al, 2005], esto se debe a que son utilizados por la gran mayoría de los sistemas software; en tal sentido, el desarrollo de patrones que soporten estas funcionalidades es un avance en el ámbito de la usabilidad.

En lo referente al estado de la cuestión sobre los patrones de usabilidad, éstos han sufrido un proceso de evolución hacia arquitecturas de usabilidad, debido a la experiencia que se ha adquirido en la definición y educción de patrones de usabilidad [Ferre et al, 2003; 2004; Juristo et al, 2004].

En este contexto se puede observar la dinámica del área en cuestión, donde se está buscando un punto de equilibrio entre funcionalidad, rapidez de diseño y robustez de los modelos propuestos para la construcción de artefactos software.

En función de los patrones más referenciados, se ha elegido el patrón UNDO/REDO; esta selección se ha debido a la complejidad del mismo, pues este debe poder deshacer o rehacer cualquier operatoria dada por un usuario. Esta complejidad sumada a la diversidad de artefactos de software que pueden utilizar esta funcionalidad, hacen de este patrón, según el entender del autor de esta tesis, el más complejo de implementar.

1.2. OBJETIVO DE LA TESIS

Se define como objetivo de esta tesis proponer una solución al problema de la funcionalidad UNDO/REDO. Dar solución a esto implica resolver los siguientes problemas, en primer lugar definir una abstracción para el modelo de datos a ser manejada por el UNDO/REDO; en segundo lugar, definir un proceso para la inclusión de la funcionalidad de UNDO/REDO en una aplicación nueva o existente.

1.3. PRODUCCIÓN CIENTÍFICA DERIVADA DE RESULTADOS PARCIALES DE LA TESIS

Durante el desarrollo de esta tesis se han comunicado resultados parciales a través de diversas publicaciones que a continuación se detallan:

Capítulos de Libros:

Merlino, H., Dieste, O., Pesado, H., García-Martínez, R. 2010. *Framework to Provide Highly Automated UNDO Capabilities on Software Systems*. En Ingeniería de Software e Ingeniería del Conocimiento: Tendencias de Investigación e Innovación Tecnológica en Iberoamérica (Editores: R. Aguilar, J. Díaz, G. Gómez, E- León). Pág. 194-204. Alfaomega Grupo Editor. ISBN 978-607-707-096-2..

Merlino, H., García-Martínez, R., Pesado, P., Dieste, O. 2012. *Inclusion Process of UNDO/REDO Service in Host Applications*. Chapter 4 in Software Engineering: Methods, Modeling, and Teaching, Volume 2. Pág. 29-36.

Sello Editorial de la Pontificia Universidad Católica del Perú. ISBN 978-612-4057-84-7.

Artículos en Revistas:

Merlino, H., Dieste, O., Pesado, P., Garcia-Martinez, R. 2014. *Move to Usability SOA Arquitecture: Undo Process Implementation*. Lecture Notes on Software Engineering, 2(2): 155-160. ISSN-2301-3559.

Congresos Internacionales:

Merlino, H., Dieste, O., Pesado, H., García-Martínez, R. 2012. *Software as a Service: Undo*. Proceedings 24th International Conference on Software Engineering and Knowledge Engineering. Pág. 328-332. ISBN 978-1-891706-31-8.

Merlino, H., Dieste, O., Pesado, P., Garcia-Martinez, R. 2012. *Service Oriented Architecture for Undo Functionality*. Website 6th International Conference on Research and Practical Issues of Enterprise Information Systems. September 19-21. Ghent, Belgica.

Merlino, H., Dieste, O., Pesado, P., Garcia-Martinez, R. 2013. *Formal Description for SaaS Undo*. Lecture Notes in Business Information Processing, 150: 217-222.

Congresos Nacionales:

Merlino, H., Dieste, O., Pesado, P., García-Martínez, R. (2009). *Design of an UNDO Framework*. Proceedings XV Congreso Argentino de Ciencias de la Computación. Workshop de Ingeniería de Software. Págs. 870-879. ISBN 978-897-24068-4-1.

1.4. VISIÓN GENERAL DE LA TESIS

En el capítulo Introducción se plantea el contexto de la tesis, se establece su objetivo, se presentan las publicaciones vinculadas a las investigaciones realizadas durante el desarrollo de la misma y se resume la estructura de la tesis.

En el capítulo Estado de la Cuestión se definen los patrones de usabilidad, se introducen las aproximaciones a arquitecturas de usabilidad y la utilización del software como servicio (SaaS) en

la ingeniería de software y especialmente en el área de usabilidad, se plantea la evolución hacia el software como servicio, y se da una discusión marco.

En el capítulo Descripción del Problema se presentan el contexto, la definición del problema abordado y se plantean las preguntas de investigación.

En el capítulo Solución Propuesta se presenta el abordaje propuesto en la tesis al problema introducido. Se dan las aproximaciones a las soluciones consideradas, entre las que se discuten soluciones Ad Hoc, Patrones, Software as a Service y Framework; se introducen las definiciones de Artefacto de Usabilidad UNDO/REDO dando una descripción formal, se introduce el concepto de Unidad Lógica de Cambio y Puntos de No Retorno. Se introduce el proceso de Inclusión de Servicios en Aplicaciones Anfitrionas Basados en Patrones de Usabilidad y para cada fase de este se definen las entradas, las salidas, los potenciales problemas que se pueden presentar y soluciones propuestas en sus fases.

En el capítulo Casos de Validación se presenta un conjunto de casos seleccionados para demostrar la viabilidad del modelo propuesto. Se da un primer caso donde se modela una aplicación Web y se inserta el servicio de UNDO/REDO, Caso 1: Aplicación Web; como segundo caso se implementa sobre el primer caso una extensión que posibilita acceder a la aplicación desde plataformas móviles, Caso 2: Aplicación Móvil; como tercer caso se presenta una reingeniería completa, para que el primer caso pueda ser ejecutado en un ambiente de Cloud Computing, Caso 3: Aplicación en Cloud Computing.

En el capítulo Conclusiones se presentan las aportaciones de esta tesis doctoral y se destacan las futuras líneas de investigación que se consideran de interés en base al problema abierto que se presenta en este trabajo de tesis.

En el capítulo Referencias se listan todas las publicaciones consultadas para el desarrollo de esta tesis.

2. ESTADO DE LA CUESTIÓN

En este capítulo se presenta el estado de la cuestión sobre la temática que se aborda en este trabajo de Tesis (Sección 2.1), se definen los patrones de usabilidad (sección 2.2), las aproximaciones a arquitecturas de usabilidad (Sección 2.3), utilización del software como servicio (SaaS) en la ingeniería de software y especialmente en el área de usabilidad (Sección 2.4), se plantea la evolución hacia el software como servicio (Sección 2.5) y se da una discusión marco (Sección 2.6).

2.1. ESTADO DE LA CUESTIÓN

La evolución que se ha dado en el área de usabilidad es similar a la encontrada en otras ramas de la ingeniería del software, la cual se inicia con esfuerzos individuales de los distintos equipos de investigación para la inclusión de funcionalidades de usabilidad dentro de una aplicación. Con la experiencia adquirida a través de un proceso ad hoc se evoluciona a un modelo de patrones donde se especifican un conjunto de buenas prácticas probadas que pueden ser extrapoladas a otros diseños; en base a estos conocimientos adquiridos, los diseñadores están en condiciones de elaborar mejores y más complejas soluciones, llegando así a la definición de arquitecturas de software. A partir de este punto se debería evolucionar a soluciones aún más complejas como ser el modelo software como servicio (SaaS de sus siglas en inglés) [Cysneiros & Kushniruk, 2003].

SaaS es el último eslabón de una cadena que comienza con el hardware que se encuentra en el centro de cómputos (DC de sus siglas en inglés), los servicios como infraestructura (IaaS de sus siglas en inglés) continua con la plataforma como servicio (PaaS de sus siglas en inglés) para finalizar con le SaaS, en la figura 2.1 se puede observar la pila de abstracción del concepto mencionado anteriormente [Meier, 2010].

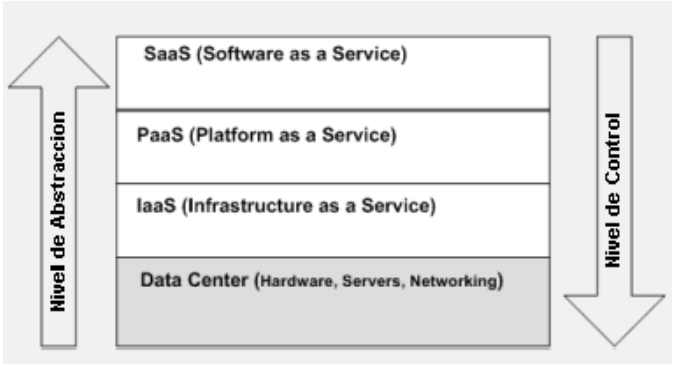


Figura 2.1. Pila de Abstracción.

Los requisitos de usabilidad pueden ser considerados como requisitos que capturan los objetivos y métricas de usabilidad asociadas a un sistema en desarrollo, con el fin de asegurar y garantizar la identificación de los mismos en una forma ordenada y adecuada a los sistemas en cuestión [Cysneiros & Kushniruk, 2003].

La usabilidad es un factor central para el éxito general de un sistema software; incluir características simples de usabilidad puede aumentar considerablemente la funcionalidad percibida del sistema por parte del usuario. La amplia aceptación de funcionalidades, como ser UNDO/REDO, muestra que tales características se han convertido en parte esencial de los sistemas interactivos [Roder, 2012]. Estas pueden añadir características de usabilidad como ser la tolerancia de errores y la capacidad de control de sistemas a partir de la interacción persona-ordenador (HCI); debiéndose señalar al respecto, que características funcionales de usabilidad deben considerarse claramente en la definición y construcción de sistemas [Juristo et al, 2007].

En la actualidad la facilidad de uso en un sistema es reconocida como factor central en la aceptación de los mismos por parte de los usuarios, en tal sentido, cabe destacar que existe una necesidad de generar un enfoque sistemático en el uso de modelos de usabilidad e incluir el análisis de usabilidad de software desde las primeras etapas de desarrollo [Cysneiros et al, 2005].

El presente trabajo de Tesis da un nuevo enfoque para la funcionalidad UNDO/REDO, esta propuesta resuelve el subconjunto de casos de operaciones sin estado, en forma eficiente; demostrando la importancia de disponer de una solución automatizada para la inclusión de la funcionalidad en sistemas nuevos o existentes.

Se ha implementado un entorno de trabajo, comúnmente llamado “Framework” por su nombre en inglés, el cual consta de un servicio (SaaS) de UNDO/REDO y de una metodología para su inclusión en forma ordenada y progresiva en una aplicación anfitriona; para este desarrollo se han tomado como idea base Frameworks como ser [Spring, 2012] e [Hibernate, 2012], por último se hizo hincapié en que la aplicación anfitriona debe recibir un conjunto reducido y simple de modificaciones para poder incluir la funcionalidad de UNDO/REDO.

2.2. PATRONES DE USABILIDAD

Los principios de usabilidad son un conjunto de buenas prácticas de software, las cuales hacen que una aplicación tenga una interacción acorde a las características y expectativas del usuario [Ferre et

al, 2003]. Dentro de estos principios se encuentra la funcionalidad de UNDO/REDO, la cual permite que un usuario pueda deshacer o rehacer una acción ejecutada con anterioridad, de aquí en más se nombrará indistintamente al patrón como UNDO/REDO o solo UNDO, debido a un uso extendido en la comunidad de este concepto.

La inclusión de esta funcionalidad en un sistema nuevo o existente no es un proceso trivial, una de las razones es que su inclusión, en términos generales, se realiza en una etapa avanzada del desarrollo de sistemas [Ferre et al, 2003], cuando las decisiones claves de diseñado ya han sido tomadas.

Los sistemas, habitualmente son construidos, por ingenieros de software con escasa experiencia en usabilidad de software; estos a menudo poseen un conocimiento limitado de su uso en forma adecuada en un sistema, como agregado a los conceptos expuestos, la poca estandarización que existe en el área de usabilidad aumenta el problema [Bevan, 2009]. Este hecho, no menor, parece justificar en forma razonable el tener que proveer a los ingenieros de software de un conjunto de herramientas de usabilidad; mostrando qué tipos de características de usabilidad existen, cuándo y cómo los aspectos de usabilidad se deberían tener en cuenta y como ciertas características con impacto positivo y negativo pueden influir en el uso de los principios de usabilidad [Roder, 2012].

El concepto de patrones como "soluciones recurrentes a problemas comunes en un contexto dado" [Alexander, 1979], ha ganado popularidad tanto en la ingeniería del software como en la comunidad de HCI. En el área de HCI se han detectado soluciones probadas para la construcción de interfaces de usuarios, las cuales han sido utilizadas como un medio de comunicar y enseñar el conocimiento adquirido [Roder, 2012].

Estos principios de usabilidad han tomado la forma de patrones, los cuales han sido concebidos con el objetivo que el desarrollo de software sea simple y predecible [Ferre et al, 2004]. Estos patrones se pueden definir como mecanismos utilizados durante el diseño de sistemas para proporcionar al software un conjunto de características específicas de usabilidad [Ferre et al, 2003].

Algunos patrones de usabilidad definidos en la literatura son: Feedback, UNDO/REDO, Cancel, Form/Field Validation, Wizard, User Profile y Help [Juristo et al, 2005]. El principal escollo para la inclusión de estos patrones es la ausencia de un marco de trabajo que contemple todo el proceso de inclusión del artefacto software “Funcionalidad de Usabilidad” en una aplicación nueva o existente, de aquí en más aplicación anfitriona, el cual haga hincapié en aspectos arquitectónicos, de diseño y de performance asociados con los patrones de usabilidad. Esto significa que los patrones, al momento de escribir este trabajo de Tesis, tienen que ser aplicados ad hoc en cada

sistema; lo cual implica que el costo de desarrollo del sistema se incremente como resultado de la mayor carga de trabajo causada por cada diseño e implementación de las funcionalidades de usabilidad. Esto origina que ciertas características de usabilidad quedarán fuera del desarrollo para reducir ese esfuerzo o tiempo de entrega del sistema anfitrión.

El objetivo de este trabajo consiste en desarrollar un Framework para los principales patrones de usabilidad, se ha seleccionado para comenzar el patrón UNDO/REDO, que es un patrón de uso común en la literatura [Abowd & Dix, 1991]; esto justifica la selección del mismo para ser el primer patrón de usabilidad a ser incluido en un marco de trabajo. A la sazón existen otros motivos de carácter técnico que sostienen la decisión de comenzar con el patrón UNDO/REDO, es que este patrón comparte gran parte de su infraestructura con otro patrón que es Cancel y también esto se aplica pero en menor grado a patrones como Feedback y Wizard.

Se ha trabajado con especial interés sobre el patrón UNDO/REDO en el área de los editores de texto [Qin & Sun, 2001], es de destacar la proliferación de patentes sobre este tema, como ser [Bates & Ryan, 2000], en la misma línea de pensamiento se han generado patentes para la implantación de la funcionalidad de UNDO/REDO en los editores de documentos en entornos monousuario como ser en [Baker & Storisteanu, 2001]. Aunque los conceptos subyacentes pueden ser exportables a otros dominios; estas propuestas se definen a un alto nivel, sin una implementación real que demuestre su capacidad de ser reutilizable en diferentes tipos de sistemas, en consecuencia, estas propuestas no resuelven el problema en un sentido amplio que debiera ser considerado para un patrón de software.

Es de mencionar que se han definido soluciones específicas para los editores de texto de uso compartido que soportan la funcionalidad de UNDO/REDO como en [Sun, 2000] y [Chen & Sun, 2001] y [Yang et al, 2004].

La razón de la cantidad de soluciones presentadas para editores de texto es su relativa simplicidad; conceptualmente un editor es un contenedor de objetos con ciertas propiedades (forma, posición, entre otras) por lo tanto, el UNDO es relativamente fácil de implementar, dado que se trata de almacenar el estado del contenedor en unidades de tiempo *i* o por cambios realizados, es de señalar que se suele utilizar una combinación de ambas, estas se van incluyendo en una estructura de pila de la siguiente forma $i+1, ..., i+n$, donde $i+n$ es el último estado almacenado e $i+(n-1)$ es el ante último cambio almacenado y así hasta llegar al último elemento de la pila; luego de este proceso de almacenamiento, cuando el comando UNDO es invocado, el contenedor, en este caso la estructura pila, se ejecuta la inversa $i+n, i+(n-1), \ldots, i$.

Una derivación de las soluciones propuestas para los editores de texto es una implementación alternativa de UNDO/REDO para sistemas de correo electrónico como se define en [Brown & Patterson, 2003], estas soluciones se implementan dentro del editor de texto que posee sistemas de correo electrónico.

Los problemas de UNDO en entornos multiusuario también han atraído atención significativa, tanto [Qin & Sun, 2001] como [Berlage & Genau, 1993] y [Abrams & Oppenheim, 2001] han propuesto mecanismos para el uso de UNDO en entornos distribuidos, en tanto [Abowd & Dix, 1991] proponen un marco formal de referencia para poder definir la funcionalidad de UNDO.

En los entornos distribuidos, la solución tiene que manejar la complejidad de los cambios de los datos compartidos, lo cual se realiza por medio de un archivo histórico de cambios [Berlage & Genau, 1993].

Varios trabajos han proporcionado información sobre los aspectos internos de UNDO, como ser [Mancini et al, 1996] que describe las características del proceso de UNDO, del mismo modo [Berlage, 1994] propone la construcción de un método de UNDO en base a comandos en entornos gráficos; [Burke, 2007] ha trabajado sobre el concepto de una infraestructura de UNDO y [Korenshtein, 2003] da las pautas para realizar un modelo de UNDO selectivo.

Otro aspecto en el cual se ha estado trabajado es la elaboración de un modelo de representación de las acciones realizadas por los usuarios en [Washizaki & Fukazawa, 2002], esta es una estructura dinámica de los comandos ejecutados en forma histórica.

La funcionalidad de UNDO mediante la representación de modelos de grafos ha sido ampliamente desarrollada por [Berlage, 1994], aquí se presenta una distinción entre el UNDO lineal por medio de un archivo histórico y uno no lineal, el cual es representado por un grafo, donde se pueden abrir diferentes ramas de acuerdo a las acciones del usuario. En [Edwards & Mynatt, 1998] también se presenta una estructura de grafo donde a diferencia del trabajo anterior, las ramas del árbol representan un nuevo conjunto de acciones realizadas por el usuario.

En [Dix et al, 1997] se trabaja sobre un grafo en forma de cubo para representar la historia de las acciones realizadas por el usuario. Por su parte [Edwards et al, 2000] modela las acciones de UNDO en forma de hilos paralelo de ejecución.

Todos estas alternativas de representación de las acciones de UNDO en forma de comandos son válidos, pero no es una tarea sencilla de implementar, ya que crear una nueva rama de acción o la unión de dos ramas existentes no es una acción trivial, habida cuenta de que se deben conocer todas

los posibles caminos que puede tomar un usuario; en consecuencia, puede ser recomendable generar una estructura lineal ordenada por tiempo, esta estructura puede ser una cola, la cual resulta fácil de implementar y administrar; un bosquejo de esto se puede observar en [O´Brain & Shapiro, 2004] que han utilizado un registro donde se almacena temporalmente las acciones, este modelo ha sido ampliamente utilizado por su sencillez

Históricamente, se han utilizado para representar el UNDO/REDO el patrón "Command" [Buschmann et al, 1996], [Fayad & Shumidt, 1997] y [Meshorer, 1998] esto sirve para mantener una lista de comandos ejecutados por el usuario, pero no es suficiente para crear un Framework que sea sencillo de incluir en sistemas existentes.

La funcionalidad de UNDO/REDO también se ha asociado a los mecanismos de excepción para revertir una acción que falla como en [Shinnar et al, 2004], este modelo sólo se invoca ante una falla.

Se ha trabajado en patentes, como el método para la construcción de un proceso de UNDO/REDO en un sistema, como en [Keane & Mitchell, 1996] curiosamente, en este trabajo se presenta lo contrario de un proceso de UNDO, es decir, ejecutar nuevamente la acción de deshacer el deshacer. [Nakajima & Wash, 1997] definen un mecanismo para la gestión de múltiples niveles de UNDO/REDO, [Li, 2006] describe un algoritmo de UNDO/REDO, y [Martínez & Rhan, 2000] presentan un método de administración gráfica de UNDO/REDO, basado principalmente en un modelo de interfaz gráfica.

El mayor problema con lo antes referido es, una vez más, la dificultad de ser adoptados en los procesos de desarrollo de software fuera del dominio de editores de textos. La única excepción notable a esto es un patrón a nivel de diseño llamado Memento [Gamma et al, 1994]; este modelo recupera un objeto a un estado anterior y proporciona un mecanismo independiente de la implementación que se pueden integrar fácilmente en un sistema; el inconveniente que presenta este modelo es que este patrón no es fácil de incluir en un sistema existente. Además, Memento sólo restaura un objeto a un estado anterior, no considerando ninguna de las otras opciones que un patrón de UNDO debe incluir.

La funcionalidad UNDO en un sistema de trabajo en grupo tiene que permitir a un usuario revertir los efectos de una operación, incluso si otros usuarios han emitido otras operaciones después de este [Gregory et al, 1992]. El control de concurrencia y el deshacer son cuestiones importantes en el diseño del trabajo en grupo, especialmente para los editores colaborativos. Se ha presentado una versión mejorada de los algoritmos existentes para ambientes distribuidos y el control de

concurrencia que se basan en transformaciones de operaciones. Dado que el uso del algoritmo se basa en una corrección formal, se presenta un conjunto de condiciones necesarias y suficientes para satisfacer y garantizar la coherencia en una arquitectura replicada [Ressel et al, 1996].

Las soluciones presentadas hasta aquí están optimizadas para casos particulares y son difíciles de aplicar a otros dominios, por el otro lado, es necesario incluir una gran cantidad de código asociado a la funcionalidad de UNDO en la aplicación anfitriona.

2.3. ARQUITECTURAS DE USABILIDAD

Existe un conjunto de investigadores que han hecho evolucionar el concepto de patrones a arquitecturas de usabilidad basándose en la experiencia que han adquirido en la definición y educción de patrones de usabilidad [Ferre et al, 2003; 2004; Juristo et al, 2003]. En [Juristo et al, 2005] se aborda la problemática de la usabilidad de software no como un aspecto a tener en cuenta una vez terminado el artefacto software, sino a lo largo de todo el proceso de desarrollo, esta referencia es la primera que se ha encontrado en la búsqueda documental realizada para el presente trabajo de Tesis, esta idea tiene al menos dos aspectos importantes a considerar, en primer lugar su alineación con el sentido común, el cual diría que todo requerimiento de software ya sea central de la aplicación como secundario debe ser contemplado desde un principio; en segundo lugar la idea de considerar a la usabilidad como parte de la arquitectura de un sistema; en los considerandos de los trabajos relacionados también se definen un conjunto de mesuras para evaluar la usabilidad en un artefacto software como ser:

(a) Facilidad de aprendizaje, como los usuarios pueden empezar a interactuar con el sistema,
(b) Eficiencia de uso, es el número de tareas por unidad de tiempo que el usuario puede hacer con el sistema,
(c) Confiabilidad, o también llamado fiabilidad de uso, es la capacidad del sistema para recuperarse de un error dado,
(d) Satisfacción, esta es la visión subjetiva del usuario ante el uso del sistema.

En [Ferre et al, 2003] se continúa con esta idea, una arquitectura donde sea incluida la usabilidad como un elemento a ser considerado desde el inicio del desarrollo, aquí se dan los lineamientos para realizar a través de los requerimientos de software un proceso para detectar características de

usabilidad. El documento citado está relacionado con [Juristo et al, 2005] donde se especifica en detalle el proceso de elicitación de requerimientos relacionados con la usabilidad.

En [Ferre et al, 2004] los autores dan los lineamientos a ser considerados para la integración de las buenas prácticas de usabilidad dentro de un proceso de ingeniería de software; aquí se detallan un conjunto de pasos para la evaluación de las características de usabilidad deseables a ser incluida en la arquitectura de software.

En todos estos trabajo se puede observar que la comunidad de software ha reconocido la necesidad de integrar los requerimientos de usabilidad dentro de la arquitectura general de sistemas y evolucionar hacia un modelo integrador, donde se puedan detectar desde un principio características de usabilidad que deben ser sumadas al desarrollo, reconociendo que estas no pueden ser elicitadas de la misma forma que son detectadas las características funcionales de un sistema.

En este sentido el presente trabajo se alinea a esta corriente de pensamiento e intenta dar un paso más en esa dirección y poder generar una arquitectura totalmente desacoplada del proyecto principal como se verá en el desarrollo de este trabajo.

2.4. SOFTWARE COMO SERVICIO

También denominado como "software bajo demanda", es un modelo de entrega de software en el que los datos y el software asociado están alojados en servidores de Internet, comúnmente referenciados como la nube. Los usuarios acceden al mismo a través de clientes ligeros o un navegador Web. El software como servicio (Software as a Service – SaaS, según sus siglas en inglés) se ha convertido en un modelo de entrega común para muchas aplicaciones, incluyendo contabilidad, gestión de contenidos, relaciones con clientes (CRM), sistemas de información gerencial (MIS), planificación de recursos empresariales (ERP), facturación, gestión de recursos humanos (HRM). Según [Torbacki, 2008]. SaaS se ha incorporado a la estrategia de las principales compañías de software empresarial a nivel global.

SaaS ayuda a las organizaciones que incorporan esta modalidad de software a evitar los gastos de capital pasando a ser la funcionalidad utilizada un gasto operacional [Mehta & Muliok, 2009]; en tal sentido SaaS puede ser considerado como un paradigma de software por reparto (Software Delivery Paradigm – SDP, por sus siglas en inglés) [Muliok, 2009]. Las ventajas de la utilización del SaaS según [Armbrust et al, 2010] son:

- ***Costos más bajos***: El usuario no debe pagar en una solo cuota la licencia del software que utiliza; este sólo abona cuotas periódicas de suscripción de menor valor.

- ***Menores requerimientos de almacenamiento***: El usuario no necesita almacenar el software o los datos en su ordenador, es por esto que no necesita grandes instalaciones de almacenamiento de datos. También está la comodidad de no tener que realizar copias de seguridad de los datos.

- ***Menos recursos en personal***: SaaS reduce la necesidad de personal especialmente entrenados para manejar mantenimiento y actualizaciones de software. El proveedor de SaaS ofrece un equipo dedicado a manejar estas tareas.

Continuando con el conjunto de ventajas que se puede observar de la utilización de SaaS en [TraceOne, 2012] se extienden las mismas al siguiente conjunto:

- ***Reducción de costos***: La inversión inicial es nula o muy baja; además, no es necesario adquirir nuevos equipos, espacio en disco ni otro dispositivo; la construcción de presupuestos resulta mucho más sencilla ya que se sabe, o se tiene una idea muy acertada de cuánto se va a gastar de ante mano.

- ***Reducción de tiempo***: Despliegue rápido en modo de actualización continúa; configuración y funcionamiento más rápidos gracias a que el software está disponible a través de una simple conexión a Internet; esto conlleva a tener accesibilidad en línea permanente desde cualquier punto.

- ***Actualización Continua***: Nuevas versiones las aplica el proveedor del software.

- ***Máximo nivel de reutilización:*** Al encontrarse todo centralizado es más sencillo su rehúso.

- ***Seguridad***: El acceso a datos se restringe al uso de plataformas, certificados, tarjetas inteligentes y espacios de trabajo aislados; en muchos casos, el software SaaS proporciona un nivel de seguridad mayor del que las empresas desearían tener en el software residente en sus propias instalaciones.

- ***Reducción de personal***: Se debe contar con menos personal en la empresa que utiliza el SaaS.

- ***Ecológico***: Un beneficio inesperado del software SaaS es que resulta ser la solución de TI más ecológica gracias a que varios clientes comparten servidor y requisitos de almacenamiento.

SaaS hace referencia a algo que se ofrece sin tener que ser instalado, configurado y mantenido por el administrador del propio sistema; la compañía que ofrece el software, se encarga del mantenimiento y la operación diaria del servicio que prestan a los usuarios [Melendy. 2012].

2.5. EVOLUCIÓN HACIA EL SOFTWARE COMO SERVICIO

La evolución desde los Centros de Datos (Data Center – DC) hasta el SaaS se ha dado gracias a la evolución que ha tenido la computación en los últimos años y la masificación en el uso de Internet tanto sea para empresas como para uso particular. Tomando como base los DC se han ido construyendo un conjunto de capas tecnológicas las cuales interactúan entre sí, estas son el fruto de la evolución a través del conocimiento adquirido por el uso.

Estas distintas capas de tecnología son, a saber:

a) ***IaaS***: En este caso, los recursos informáticos tales como ser capacidad de procesamientos, recursos de almacenamiento y red, se exponen para que los mismos puedan ser accedidos por las personas que posen permiso de acceso a los mismos. En lugar de administrar o controlar la infraestructura subyacente, se provee la infraestructura como un servicio; como ejemplo paradigmático de este servicio se

puede mencionar a el IaaS de Amazon denominado "***Elastic Compute Cloud (EC2)***" [Meier, 2010].

b) ***PaaS***: En este caso a diferencia del anterior se proveen ya no recursos básicos sino herramientas de programación y/o plataformas de desarrollos, como ser Java, Python o NET., y las API para construir aplicaciones basadas en servicios; como ejemplo de esto se puede nombrar a "**Amazon Simple Storage (S3)**" , "***Microsoft Azure***" y "***Force.com***" [Meier, 2010].

c) ***SaaS:*** Aquí las aplicaciones que se exponen son productos los cuales no es necesario que el usuario tenga que realizar algún tipo de configuración o programación compleja ya cuenta con una interfaz de usuario lo suficientemente desarrollada para que pueda ser manipulada directamente [Craig Wood, 2010], como ser "***Google App for Business***".

d) ***SOA:*** En este caso lo que se provee es un método para la construcción de una aplicación [Laplante et al, 2008], este modelo está basado en todas las demás capas antes descriptas y es utilizada como una línea guía para la construcción de software en la modalidad de servicios.

En la figura 2.2 se detallan los distintos niveles de abstracción, sus relaciones y aplicaciones que pertenecen a cada nivel [Craig Wood, 2010]. En dicha figura se puede observar como una capa hace uso de los servicios de la inferior y a su vez provee los recursos para que la superior pueda funcionar.

A nivel de IaaS se puede observar como partiendo desde el nivel eléctrico y mecánico de DC se agrega el nivel de red y los corta fuegos (Firewalls), en esta capa también son considerados los servidores a nivel físico y su virtualización; esta capa provee, como se puede observar, un conjunto de servicios los cuales pueden ser definidos como tangibles, es decir equipos e infraestructura para el funcionamiento de un DC. Sobre esta se encuentra el PaaS en el cual se

encuentran en primer lugar los sistemas operativos y sobre ellos un conjunto de herramientas que sirven para la construcción de aplicaciones como ser bases de datos y lenguajes de programación.

A su vez, la capa de SaaS es un conjunto de servicios de aplicaciones que pueden ser accedidos a través de una red directamente a los equipos de los usuarios.

En la figura descripta anteriormente no se hace referencia a SOA, pues esta se vincula fundamentalmente con un método de integración de servicios más que con una capa de infraestructura en sí.

En la figura 2.3 se detallan los diferentes niveles de servicio, SaaS, PaaS e IaaS y se realiza una comparación con los niveles de acceso que poseen, tanto el proveedor como el usuario, a su vez la figura es dividida horizontalmente en tres niveles: aplicación, plataforma e infraestructura.

Se puede observar como a más bajo nivel, extremo izquierda de la figura, IaaS, el responsable de la mayor parte de la administración es el usuario y hacia la derecha, SaaS, el responsable de toda las capas es el proveedor de servicio [Engine, 2012].

En esta figura se puede observar claramente que el modelo de SaaS es un modelo en el cual los usuarios deben utilizar menor cantidad de recursos para la administración, y por consiguiente, menor nivel de conocimientos para el uso de las aplicaciones provistas por el servicio.

Una vez realizada una breve reseña de las principales características de los distintos modelos y su interrelación, se tratará en detalle de cada uno de ellos haciendo referencia a trabajos que dan sustento a los mismos.

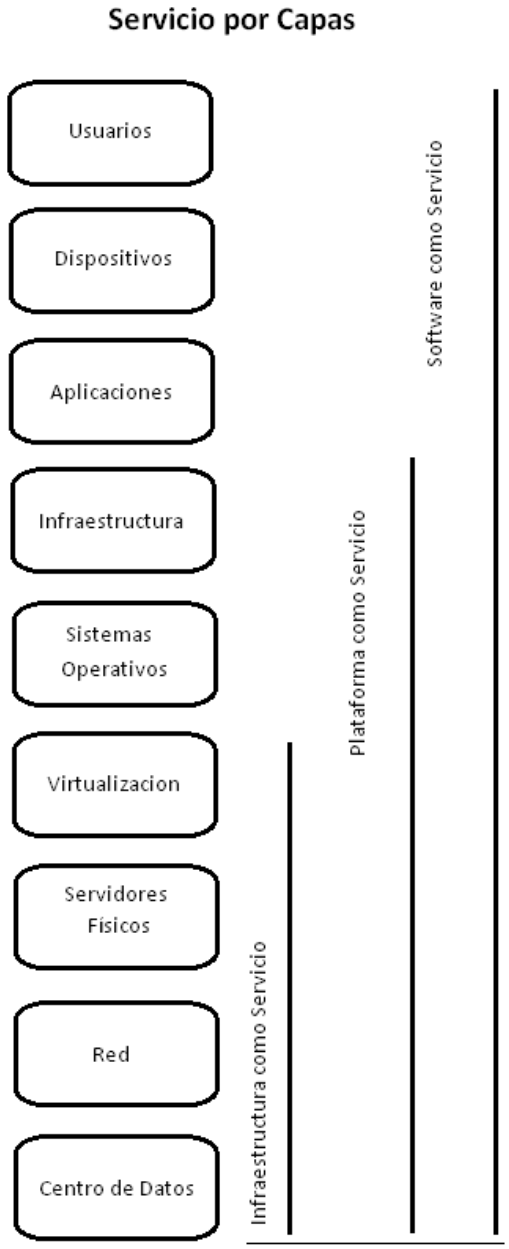


Figura 2.2. Detalle de Abstracción y Aplicaciones.

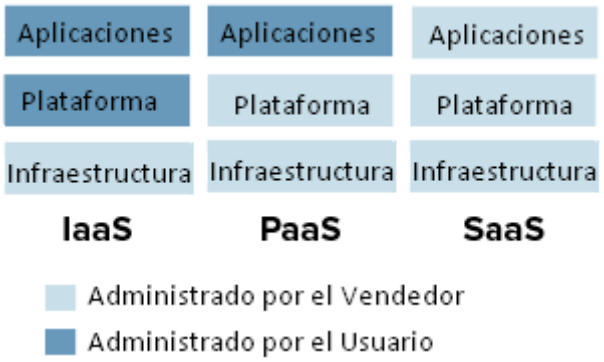


Figura 2.3. Modelo de acceso.

2.5.1. Introducción a la Infraestructura como Servicio

El uso de la Infraestructura como Servicio se ha vuelto frecuente en los últimos años; los usuarios de IaaS se enfrentan al reto de la gestión de los diferentes servicios y aplicaciones que se ejecutan en los diferentes proveedores de IaaS. Extendiendo este concepto, un usuario puede necesitar acceder a un mismo servicio pero a través de diferentes proveedores, esta arquitectura es frecuente en ambientes donde debe existir un modelo de redundancia, lo cual obliga a los consumidores de servicios de IaaS a interactuar con diferentes interfaces de aplicación (Application Programming Interface - APIs), que ofrecen los diversos proveedores, esto trae como consecuencia que la complejidad de la gestión de los servicios sea cada vez mayor.

Para dar una solución a este aspecto problemático de los IaaS se ha propuesto la utilización el servidor de proximidad (Proxy Server) [Shixing et al, 2010], el cual provee una capa de abstracción entre el API publicada por el proveedor del IaaS, en este sentido las invocaciones desde el cliente son unificadas para los distintos proveedores, así se evita la integración para cada proveedor de un servicio de IaaS.

Esta propuesta es innovadora y superadora, pues permite que los consumidores de servicios puedan concentrarse en la orquestación de sus sistemas y no en la complejidad de entender como cada proveedor resolvió su conjunto de APIs. Cabe destacar en esta instancia que la idea expuesta constituye la continuidad del camino que ha transitado la comunidad de bases de datos, la cual luego de años de tratar de imponer cada proveedor de base de datos un estándar, se terminó por trabajar en un conjunto y definir una capa intermedia la cual cada usuario solo debería conocer un solo método de invocación a las distintas funcionalidades que provee un motor de base de datos, este modelo es conocido con OBDC (Object Data Base Connection) [Signore, 1995].

Es necesario hacer referencia que el trabajo de [Shixing et al, 2010] no propone un modelo unificado, aunque si es importante destacar que deja sentadas las bases para que el próximo paso sea lo que ya se ha realizado en la comunidad científica de base de datos. Para el modelado de este proxy los autores han seleccionado el método DMTF CIM (Common Information Model) [DMTF, 2010] este es ampliamente utilizado por la comunidad de objetos como herramienta de modelado para describir las entidades.

Los autores utilizan el CIM como meta-modelo para definir el nuevo servidor de proximidad de IaaS, unificando los diferentes modelos de invocación de las interfaces de los IaaS mediante la definición de un conjunto de operaciones expuestas por el servicio, lo cual proporciona un conjunto de servicios a los que se pueden acceder. Este hecho origina que exista una creciente

comunidad de proveedores de IaaS en Internet lo cual hace que esta solución constituya una alternativa valiosa para las organizaciones que deben evaluar la utilización de servicios de IaaS.

También en el trabajo de [Lee & Hur, 2011] se ha desarrollado una herramienta llamada Infrastructure-as-a-Service Aggregator (IaaSA) para proporcionar a los administradores de tecnología de la información (TI) la posibilidad de suscribirse a los recursos de diferentes proveedores de IaaS y al mismo tiempo utilizar una interfaz común para gestionar los recursos.

El administrador de TI debe ser capaz de dividir los servicios a través de los múltiples proveedores de IaaS de manera transparente. El modelo IaaSA trabaja con DMTF y utiliza un conjunto de servicios genéricos para el IaaS, a través de una abstracción de las operaciones de uso común del IaaS de los principales proveedores. Este hecho provee una cobertura total de funcionalidades para satisfacer las necesidades de los usuarios IaaS.

Estos trabajos han generado interrogantes que han terminado en un trabajo encabezado por un conjunto de empresas agrupado en el Cloud Computing Open Interface Working Group (OCCI-WG), el cual está desarrollando una solución que abarque la definición, el acceso y la monitorización de los servicios en la modalidad de IaaS [OCCIWG, 2012], esto es un avance en la dirección correcta según el entender del autor del presente trabajo.

Como se ha detallado anteriormente, el aumento en el uso de este tipo de servicios ha hecho que se deban tener en cuenta aspectos que en las primeras etapas de la tecnología no se había tratado en detalle; en este sentido, es necesario hacer referencia al trabajos realizado sobre los aspectos de seguimiento y de detección de interrupción del servicio y su relación con los recursos de infraestructura de red de IaaS [Voith, 2010]. Esto está relacionado con la calidad que se pretende obtener de un servicio el cual será utilizado como base de otras aplicaciones.

La información relacionada con el monitoreo de la infraestructura es lo que dará el nivel de calidad de servicio (Quality of Service - QoS), que es el contrato con el cual se firmó el nivel de servicio provisto (Service Lavel Agreement - SLA) hacia la plataforma IaaS se mantiene.

En este trabajo se ha prestado especial atención en crear un conjunto de mediciones y funciones de vigilancia que se ajustan a la gestión actual del middleware para el mantenimiento de la autonomía a nivel de nodo, además se ha prestado especial atención en minimizar el consumo de recursos de la red en la medición, también se ha trabajado en un modelo de una solución escalable.

Es importante hacer notar que la QoS constituye en aspecto central en una infraestructura de este tipo, pues se hace necesario para los usuarios en el modelo de alquiler de servicio que los

proveedores provean las condiciones del SLA del servicio contratado, que puede traer connotaciones económicas contra este, es por ello que el monitoreo sin un nivel de intrusión alto es importante.

Otro punto que se debe tener en cuenta en los servicios IaaS son los aspectos de seguridad relacionados con el acceso a los mismos, pues una empresa que contrata estos servicios de modo externo deberá también exportar parte de sus datos a la infraestructura del servicio de IaaS, entonces aquí ya se cuenta con dos aspectos que deben ser mantenidos y son responsabilidad del proveedor desde la aplicación, la disponibilidad del servicio y la seguridad del mismo.

La madurez de esta tecnología ha permitido que los investigadores y desarrolladores puedan intentar trabajar con nuevas alternativas de implementación, esto se puede ver en [Montero, 2012], donde partiendo de la madurez de los servicio de IaaS en ambientes de Internet estos son ejecutados bajo de la modalidad de Cloud Computing [Nurmi, 2099].

Como el uso de Cloud Computing han sido una solución viable para el aprovisionamiento de recursos de computo bajo demanda, se ha presentado un modelo que tomando la tecnología de IaaS y Cloud Computing para ser llevada dentro de la organización, donde este modelo ha sido denominado “Private Cloud”; aquí también se puede observar una correlación con lo sucedido en la utilización de los recursos de Internet, que han sido utilizados dentro de las empresas en lo que ha sido llamado Intranet [Ferraiolo et al, 1999], con lo que se puede entender que el modelo presentado en este trabajo podrá ser adoptado por las empresas que consumen estos servicios. Aquí es necesario mencionar que esto está dirigido a empresas con capacidad de manejo de la tecnología, pues este modelo va en dirección contraria a la idea original por lo cual fueron creados los servicios de IaaS de reducción de equipos y recursos para su administración.

Llegado este punto se han detallado las principales características y aspecto a tener en cuenta en los servicios de IaaS, además es necesario considerar cual ha sido el aspecto que ha permitido en gran medida la masificación de los IaaS, este aspecto es tratado en [Caraman, 2012] donde se detalla que los avances en la tecnología de virtualización de los DC ha permitido desarrollar características de alta disponibilidad, recuperación de desastres y tolerancia a fallos.

Haciendo referencia a la necesidad que los servicios IaaS que se extienden a través de áreas distribuidas, estas requieren un mecanismo avanzado de tolerancia de desastres; esto se refiere a la capacidad de recupero ante pérdida de los componentes de comunicación o pérdida de la fuente de energía y que el servicio siga prestando su funcionalidad, o poder volver a funcionar en un relativo período corto de tiempo. En este trabajo se hace hincapié nuevamente en la necesidad de proveer

una continuidad en el servicio pese a imprevistos, esta temática vuelve a tratarse en la inmensa mayoría de los trabajos relevados, la necesidad de la continuidad de los servicios.

Como resumen sobre las áreas de mayor investigación en los servicios de IaaS se puede hacer referencia al trabajo [OpenStack, 2012], donde se detallan los ámbito en que se debería hacer mayor hincapié, a saber: en primer lugar transparencia y rapidez en la recuperación de fallos, investigación de modelos para despliegues ISP Border Gateway Protocol [Kent et al, 2000] y el uso de direcciones de IP virtuales. Una segunda área de mejora es representada por las técnicas de detección de fallos y su recuperación y un tercer tema de investigación está representado por la arquitectura descentralizada IaaS donde el proyecto OpenStack es pionero.

2.5.2. Introducción a las Plataformas como Servicio

La plataforma como servicio (PaaS) es un modelo de negocio en la era de la computación en nube, que proporcionan una plataforma de servidor o entorno de desarrollo para desarrolladores, en comparación con la Software como servicio (SaaS) que esta orientado al usuario final, el cual será detallada en el próximo apartado. PaaS es flexible [Lauton, 2008] y en combinación con el Cloud Computing proveen un potencial considerable para ayudar a los desarrolladores empresariales a escribir rápidamente aplicaciones, en general en ambientes Web. Con la maduración de la tecnología detallada en el apartado anterior, la comunidad científica y la industria han orientado sus esfuerzos al desarrollo y maduración de una capa con mayor nivel de abstraccion, es decir PaaS, aquí al igual que en la capa inferior los esfuerzos se orientan a obtener la continuidad del servicio y a facilitar su interacción con el usuario.

En tal sentido el trabajo propuesto por [Xu et al, 2012] presenta una plataforma con un motor de flujo de trabajo que ayuda a los usuarios a utilizar el servicio permitiendo abordar los problemas de procesamiento de datos a gran escala sin la necesidad del conocimiento sobre configuración de la aplicación que soportara el proceso. En la tecnología de modelado basada en flujo de trabajo ha simplificado la complejidad de la ejecución y gestión de las aplicaciones, para esto se ha utilizado un modelo de gráficos dirigidos acíclicos (DAG). Este es un modelo conveniente para representar flujos de trabajo, existiendo gran cantidad de literatura sobre la programación DAG es su relevancia para el problema de la programación de flujo de trabajo como ser en [Dong & Akl, 2007; Topcuoglu et al, 2012; Byun, 2008, Byun, 2011].

Esto se ve acentuado en los últimos años donde hay un resurgimiento del interés en el contexto de los problemas especialmente motivados por la ejecución del flujo de trabajo científico y entornos

heterogéneos, esto se puede ver detallado en los trabajos [Blythe , 2005: Mandal et al, 2005; Sakellariou & Zhao, 2004; Wieczorek, 2005; Yu & Buyya, 2006].

Este trabajo divide la propuesta en dos partes, en primer lugar el "Modelo de Sistema", donde los modelos de flujo de trabajo de un proceso son representados como una serie de actividades simples y se desconecta del flujo de control y de lógica de negocios. La segundo lugar es lo denominado "Arquitectura de la Plataforma", este se compone a su vez de dos capas principales de software. La capa subyacente ofrece una infraestructura de almacenamiento distribuido y el módulo de trabajo en paralelo [Yang et al, 2007]. La capa intermedia es el servicio del motor de flujo de trabajo en sí, que proporciona la gestión de la infraestructura y la gestión de la ejecución; a su vez esta capa se compone de dos módulos, aislación de aplicaciones y tolerancia a fallos.

Como se puede observar el objetivo de este trabajo es la generación de máquinas virtuales; el beneficio de usar un nodo virtualizado para la ejecución del flujo de trabajo, en lugar de tener acceso directo a la máquina física, permite la disminución de potenciales fallas, este modelo de trabajo es conocido como "Sandboxing" [Hansen et al, 2006]. En resumen, se presenta un flujo de trabajo para arquitecturas PaaS junto a una estrategia de virtualización.

El PaaS es un nuevo modelo para los proveedores de software que quieren centrarse principalmente en el ciclo de desarrollo de software y la monetización de nuevas aplicaciones, evitando así la inversión y el mantenimiento de la infraestructura subyacente para diseño de la aplicación, desarrollo, prueba, puesta en producción y alojamiento [Lawton, 2008]; en este trabajo se hace referencia a la utilización de "Business Process Management" (BPM) como centro de TI de la empresa, esta tecnología se ha convertido en un estándar para la implementación de paquetes de software. La infraestructura requerida para el soporte y administración que debe ser soportada se detalla en el trabajo de [Zhou et al, 2010] con un modelo de BPM de múltiples usuarios para permitir ser soportado por una plataforma PaaS, este es denominado "Programable PaaS" (BPPaaS). Esta permite a los usuarios someter sus procesos de lógica de negocio en el código fuente y por este es interpretado por la plataforma.

BPPaaS analizará el código fuente, extraerá las tareas y los procesos de negocios para formar un conjunto de meta-datos, para obtener un conjunto de procesos autónomos que puedan ser ejecutados [Zheng et al, 2012].

Sin embargo, la plataforma PaaS al momento no disponen de tales características, no existe una plataforma madura para la orquestación de procesos PaaS en forma automático; la ejecución de procesos de negocio complejos a gran escala y con conjuntos de datos distribuidos en un entorno de

Cloud Computing ha sido objeto de investigación en los últimos años, se han desarrollado los flujos de trabajo de alto rendimiento en sistemas de redes como lo tratan los trabajos de [Ludscher et al, 2006; Majithia et al, 2004; Oinn, 2002] y sistemas de flujo de datos en cascada como trata [Cascading, 2012]; el objetivo final que apuntan todas estas referencias es a poder extraer los proceso de negocio de las empresas y poder ser ejecutados en plataformas PaaS heterogenias utilizando el algoritmo dividir y sumariar, conocido como "Map-Reduce".

Áreas específicas de la industria se han mostrado interesadas en la utilización de servicios de este tipo, entre estas están los operadores de telefonía celular, en este sentido se puede hacer referencia al trabajo de [Goncalves & Ballon, 2009], donde este hace constar que los entornos de desarrollo de software y suministro de software están cambiando a plataformas basadas en Web con el apoyo de Plataforma/Software como Servicio (PaaS/SaaS) así lo han demostrado los operadores móviles, que se han visto obligados a diversificar su actividad a servicios Web para aplicaciones, es por esto que los proveedores de PaaS tienen incentivos para ofrecer sus servicios de operación.

El servicio PaaS es un enfoque novedoso para los proveedores de software que quieren centrarse principalmente en la ciclo de desarrollo de software y la monetización de nuevo aplicaciones, evitando así la inversión en mantenimiento y de la infraestructura subyacente para el diseño de aplicaciones, desarrollo, prueba, implementación y almacenamiento [Mitchell, 2008]. En este sentido PaaS permite la reducción de costos en el desarrollo dando sustento a las aplicaciones anfitrionas sin la necesidad de conocer en detalle la infraestructura subyacente. Mientras que el mercado se mueve lentamente hacia este modelo, los operadores deben incorporar a su manera el concepto de plataforma; e incorporar el equilibrio adecuado de características que garantizan la adopción de la plataforma coherente y evite la fragmentación de la misma.

Es por lo antes descripto que PaaS es el entorno operativo de la nube, con herramientas que se utilizan bajo demanda para crear y alojar en línea servicios, aplicaciones de software y sitios Web ó móviles. El ahorro de costos es el principal hilo conductor para la adopción de esta tecnología [Mutavdzic, 2012]. La realidad es que las instalaciones informática para soportar este modelo puede ser costosas, hay que adquirir el hardware y software, pagar por la instalación, configurar, probar, mantener, actualizar y asegurar toda la plataforma como servicio [NIST, 2011].

El desarrollo e implementación en un entorno PaaS normalmente se controla y se inicia mediante la utilización de un entorno integrado de desarrollo (Integrado Development Environment - IDE) para crear aplicaciones que permitan maximizar los beneficios, el proveedor de la aplicación debe aplicar patrones de diseño aptos para la nube para construir sistemas que exhiben paralelismo,

multiempresa, autonomía, las interacciones distribuidas, declarativo definiciones, la separación de preocupaciones, y la federación.

Otro aspecto que ha sido de interés para la comunidad científica relacionada con PaaS, es el modelo de escalación y rendimiento de la plataforma, aquí se lo ha relacionado fuertemente con el paradigma de procesamiento en la nube (Cloud Computing); este modelo es compatible con una visión de TI distribuida, donde los servicios de software y aplicaciones son externos a la compañía y se utiliza un modelo de gestión de pago por uso (pay-as-you-go) [Armbrust et al, 2009]. En este trabajo se presentan un conjunto de patrones para diversos modelos de uso de la plataforma en relación con su escalabilidad; en primer lugar se presenta el patrón "Single Platform Pattern" (SPP), el mismo provee un modelo de escalabilidad básica y considera un escenario de un solo inquilino, usuario, en la plataforma.

El segundo patrón considerado es el "Shared Platform Pattern" (ShPP), en él se considera un escenario multiusuario en el que se instala una plataforma única en un conjunto de máquinas virtuales; por último se presenta el patrón "Clustered Platform Pattern" (CPP), esta solución es adoptada por casi todos los principales proveedores de PaaS (por ejemplo, WSO2, Google App Engine), este trabaja en una sola plataforma, a su vez esta da soporte y comparte recursos entre todos los inquilinos, es decir usuarios de la plataforma [Ardagna et al, 2012].

También se detallan combinaciones y extensiones de este conjunto básico de patrones como ser, múltiple instancias de los componentes de la plataforma que pueden ser desplegados en diferentes máquinas del clúster, esto se denomina "Multiple ShPP" (MShPP), esta utiliza tanto la escalabilidad vertical y horizontal; tras un aumento de la carga, los recursos adicionales son tratados con el patrón "Multiple CPP" (MCPP), que es una extensión de CPP y proporciona una escalabilidad horizontal; pero al tiempo de inicialización es utilizada una única plataforma de agrupamiento.

La evaluación experimental de los patrones de escalabilidad ha mostrado que la adición de recursos virtuales proporciona cuasi-linealidad de aceleración sólo en un rango muy limitado, lo que se reduce aún más cuando los mecanismos de seguridad son implementados. En este trabajo se puede observar que el problema de escalabilidad tanto sea horizontal como vertical, es un problema puntual y central para el desarrollo de la tecnología de PaaS, pues su promesa de capacidad de procesamiento hace que las demoras o fallas en el tiempo de respuesta sean inamisibles por los usuarios de la misma.

En este sentido se puede hacer referencia a una plataforma ampliamente utilizada como ejemplo de PaaS esta es "Google App Engine" [Pallis, 2010] que ha sido diseñado para alojar aplicaciones donde muchos usuarios tengan acceso en forma simultánea, esta plataforma asigna automáticamente los recursos para las aplicaciones publicadas, los programadores pueden desarrollar programas con Java, Python, PHP o Go estos incluyen en sus programas las librerías proporcionadas por la plataforma y desarrollan en función de un conjunto simple de lineamientos y luego exportan su aplicación a la plataforma y el resto es administrado automáticamente por el PaaS.

Al igual que lo detallado en el apartado anterior la comunidad de PaaS ha hecho hincapié en la necesidad de proveer un entorno común a los diferentes proveedores de la plataforma, esto se puede observar en el trabajo presentado por [Loutas et al, 2011] donde la arquitectura PaaS puede aumentar su funcionalidad con una capa semántica que alojaría a un conjunto de modelos comunes y sería el enlace entre los distintos proveedores de PaaS con características heterogéneos; este intento se debe a que es altamente complejo poder pasar una aplicación que se encuentra alojada en un proveedor de PaaS a otro proveedor, es por esto, que el trabajo en cuestión plantea que las distintas plataforma provistas deben mantener un nivel mínimo de interoperabilidad semántica para ser alcanzado.

Según [Armbrust et al, 2010], este relaciona a la capa semántica de la plataforma como el componente faltante en el vocabulario para interconectar de plataformas en el Cloud Computing; es por esto que se ha desarrollado el "PaaS Semantic Interoperability Framework" (PSIF) este estándar desarrollado por el "Cloud Computing Foro de Interoperabilidad" (CCIF) tiene como objeto llegar a un modelo de interoperabilidad entre plataformas PaaS provistas por distintos proveedores.

El "Cloud Computing Use Case Group" ha publicado un documento [CCUCDG, 2010] destacando los requisitos que deben ser estandarizados para garantizar la interoperabilidad en los escenarios de interacción basados en Cloud Computing. Sobre la base de estos escenarios, un conjunto de casos de uso se han producido, enfatizando la capa PaaS y sobre todo, la interoperación entre PaaS de los distintos proveedores. Estos modelos contribuyen a la solución semántica de conflictos de interoperabilidad que puedan plantearse durante la implementación o la migración de una aplicación mediante la definición de un el espacio tridimensional, que comprende los siguientes aspectos básicos a tener en cuenta para la integración del PaaS que son: las entidades, los tipos de semántica y los niveles de semántica en conflictos.

Otro aspecto que se ha tratado es la utilización de PaaS para la creación de aplicaciones fuera de línea (offline), esto se puede encontrar en el trabajo de [Lawton, 2008] donde se dan los lineamientos de un entorno de trabajo denominado "Framework for Implementing and Managing PaaS in a Virtual Cloud Computing Lab"; en este trabajo se presenta un entorno de trabajo para administrar la plataforma, el sistema tiene capacidad de expansión y puede mejorar la distribución de los recursos y su utilización; aquí es de mencionar que en futuros trabajos se incluirá imágenes de software y plataformas de hardware, balanceo de carga y completar aprovisionamiento automático de recursos para que el sistema se puede aplicar en gran escala y ambientes distribuidos por último la continuidad del servicio es de fundamental importancia para el éxito de la plataforma.

Siguiendo en esta línea de investigación se ha trabajado para la evaluación de la calidad del servicio y definir un conjunto de trabajos de la arquitectura de servicios, esto incluyendo la ingeniería del ciclo de vida de servicio, diseño, implementación y mantenimiento [Boniface et al, 2010].

Los parámetros QoS en tanto niveles de aplicación e infraestructura representan un desafío y es por esto que tal trabajo ha llamado la atención de la comunidad en cuestión. Este artículo analiza al PaaS ejecutado en el Cloud Computing en relación con el servicio / plataforma / infraestructura (SPI) en capas, este modelo se centra en las características de PaaS y soporte para garantías de QoS.

Se ha descrito una arquitectura PaaS para provisión de servicio en tiempo real orientada a la aplicación en las nubes; este presenta dos aspectos clave del PaaS, a saber, los servicios de ingeniería y gestión de servicios, que muestra cómo la combinación de métodos, herramientas y servicios se puede utilizar para mejorar su uso, facilitando el mantenimiento, eficiencia de los servicios con estricto control del QoS; en segundo lugar proporcionar una capa de integración entre SaaS y IaaS y como tal necesidad de mediar y resolver las diferencias entre distintos proveedores.

2.5.3. Introducción al Software como Servicio

Los cambios dramáticos en el negocio del software en los últimos años tienen implicaciones importantes para los usuarios y los desarrolladores de productos de software y servicios. Las ventas de productos tradicionales y derechos de licencia han disminuido y los ingresos por productos de la compañía se han desplazado a servicios tales como pagos anuales de mantenimiento que dan derecho a los usuarios a recibir soporte ante errores en el programa, actualizaciones de menor importancia y en ciertos casos soporte técnico.

Un cambio está en marcha en la industria de software para empresas con proveedores de servicios establecidos que deben replantear su estructura de ingresos por productos; queda por ver si la dinámica del ciclo de vida y el modelo de negocio de servicio puede ser mantenido a largo plazo [Cusumano, 2008].

En el texto antes mencionado se plantea la visión de cambio de paradigma en el armado de una compañía de software, pues el desafío de proveer servicios de software hace que las compañías deban restar recursos de algunas áreas y agregar en otras, esto se podría graficar de la siguiente manera en áreas como marketing, preventa y venta, se vislumbraría una reducción de sus estructuras, esto se debe a la existencia de nuevos canales de ventas, a saber, Google Play o App Store.

Estas aunque no están dirigidas al mercado de servicios para empresas, han generado un nuevo modelo de comercialización donde los usuarios se dirigen a un lugar especifico en busca de componentes para que en forma automática puedan ser bajados a sus dispositivos para ser utilizados inmediatamente, aunque al momento en la comercialización de servicios no es un modelo puesto en práctica, de seguir la tendencia en el uso y consumo de servicios este será una realidad. Áreas como soporte técnico, arquitectura y todo lo que esté relacionado con la continuidad del negocio deberá ser tomado mucho más en cuenta pues las empresas proveedoras no podrán interrumpir su servicio ante ninguna circunstancia.

Cada vez con más fuerza se hace hincapié en que el software orientado a servicios será la próxima revolución en el desarrollo de software; las capacidades de los servicios Web están en constante expansión desde la utilización para mensajería instantánea hasta la construcción de aplicaciones empresariales complejas. Estos nuevos enfoques orientados al servicio parecen resolver el problema de la rigidez existente en el mantenimiento y evolución de una aplicación de software.

Mientras se solucionan aspectos importantes del problema, es de mencionar que sin embargo, se siguen planteando algunas dificultades. Este cambio hacia la orientación a servicios obliga comprender a las aplicaciones de software mucho más al detalle antes de comenzar su construcción, esto conlleva un potencialmente aumento de costo en la ingeniería de software. En tal sentido el trabajo de [Gold et al, 2004] utiliza un modelo de “On-the-fly Software Service Construction”, en donde se plantean y se dan ciertas guías a los problemas que los ingenieros se enfrentan al trabajar con software orientado a servicios, también se presentan algunas cuestiones nuevas que se deben considerar, entre estas se aborda las dificultades de provisión de servicios y las fallas del mismo. En este trabajo se plantean algunos de los nuevos desafíos que surgen de la utilización de una

nueva tecnología, al considerar del autor del presente trabajo, este es un documento que inicia una nueva línea de investigación pero que no abarca la totalidad de la problemática debido a lo reciente de la tecnología en cuestión.

Como SaaS se vuelve más utilizado, la gestión de la identidad y federación entre aplicaciones SaaS también se convertirse en un factor importante que afecta el crecimiento del ecosistema de los servicios SaaS en tal sentido el trabajo propuesto por [Bin et al, 2009] intenta dar luz sobre esta problemática y presenta una modelo de trabajo en común entre diferentes servicios heterogéneos. En general existen tres funciones principales que se habiliten en la federación de identidades de servicios, a saber:

a. Inicio de sesión único a través de los diferentes servicios.
b. Cuenta de suministro para diferentes servicios.
c. Asegurar las llamadas al núcleo del servicio entre los diferentes servicios.

En la actualidad las plataformas de entrega SaaS proporciona estas funciones en forma ad-hoc, lo que podría limitar el crecimiento del ecosistema SaaS. Para superar las limitaciones, este trabajo propone un marco, que aprovecha el protocolo de identidad abierta como OpenID y OAuth. Por otra parte, un módulo ejecutor de OAuth ha sido propuso para mediar entre las llamadas al núcleo de servicios entre aplicaciones SaaS.

Este modelo puede traer beneficios a todos los roles involucrados en el ecosistema de una manera no intrusiva y centrado en el usuario. También se plantea la utilización de arquitecturas abiertas es un principio bueno de diseño y también la actitud y el espíritu de colaboración permitirán hacer crecer la plataforma. Los autores de este trabajo entienden que un ecosistema SaaS basadas en tecnologías abiertas podría hacer que la composición de los servicios sea sencilla y acelerar la puesta a punto de los proveedores de servicios. Por otra parte, más clientes también pueden ser atraídos por el carácter abierto del ecosistema.

Como se puede observar en el modelos de SaaS también se plantea la problemática existente en otros conjuntos de la industria donde la decisión de software propietario o abierto ha generado diferentes modelos de comercialización con ganadores y perdedores, pero al momento de la redacción de este documento la discusión sobre este tema para los SaaS se encuentra en una etapa embrionaria y es muy difícil vislumbrar una solución inmediata a la problemática en cuestión.

En el universo del SaaS se ha presentado la problemática de cómo soportar múltiples usuarios a través de un mismo servicio, haciendo referencia a la configuración y no a la capacidad de

procesamiento, en tal sentido el trabajo expuesto por [Tsai et al, 2010] propone la generación de un marco de trabajo (Framework - FW) para una arquitectura múltiples usuario.

Para apoyar a un número significativo de usuarios, las aplicaciones SaaS necesitan poder personalizarse para cumplir con los diversos requisitos funcionales y de calidad de los usuarios individuales. Los autores de este trabajo han presentado un sistema unificado e innovador de FW de varias capas de personalización, para apoyar y gestionar la variabilidad de las aplicaciones SaaS para los requisitos específicos de los usuarios. Este FW se base en ontologías para obtener información sobre la personalización y despliegue de las capas usuarios.

Este FW también tiene un motor de recomendación inteligente para apoyar a los nuevos usuarios que permite utilizar la información de las actuales aplicaciones SaaS dando lugar a la simplificación del trabajo de configuración y personalización de los mismos.

Con el rápido desarrollo de Internet y la madurez gradual del modelo de arquitectura orientada servicios (Software Oriented Architecture – SOA), SaaS se ha convertido en un modo de servicio de software popular. En [Shi et al, 2009] se propone un marco flexible para solucionar los problemas causados por la orquestación de procesos de negocios a través de SaaS y las especificaciones de los lenguajes de procesos de negocios (BPEL). El marco es completamente independiente del motor BPEL y servicios Web, donde se puede realizar una serie de funciones como ser personalizar el servicio durante su definición, verificar y comprobar la corrección de la lógica del proceso durante la implementación, modificando dinámicamente, sustitución de servicios y el manejo de situaciones anormales de acuerdo con las normas establecidas por los inquilinos o el FW automáticamente en el tiempo de ejecución del proceso.

En el trabajo de [Liu, 2010] se propone un modelo de referencia independiente de plataforma SaaS. En primer lugar, se intenta dar un marco teórico y una discusión sobre la arquitectura SaaS. En segundo lugar, las responsabilidades de la plataforma SaaS se analizan y los componentes de la infraestructura independiente son presentados. Nuevamente en este trabajo se propone una arquitectura basada en el concepto de múltiples usuarios; donde el usuario y el proveedor paguen por el uso de esta especificación de plataforma bajo demanda. Por último, se da un resumen sobre las ventajas independientes plataforma SaaS.

Lo que intenta detallar este trabajo que mediante una plataforma SaaS independiente varios vendedores se beneficiarán en gran medida y los usuarios reciben los mejores servicios de software. Este trabajo plantea un concepto novel en el cual tanto proveedores como clientes deban pagar por el uso de una arquitectura unificada, la principal ventaja que se obtendría para los usuarios es una

rápida puesta en producción, a los proveedores cuentan con un modelo de integración más simple por utilizar un estándar y una reducción en los costos de alquiler en los mismo, es de esperar que propuestas en este sentido sean tenidas en cuenta en el futuro próximo por tratarse de un proyecto unificador y que redundará en menores costos de construcción de software.

También se ha tratado la utilización de SaaS en dominios especiales de la industria del software, en tal sentido el trabajo de [Liyang et al, 2010] hace referencias al rápido desarrollo de la tecnología Web, sumado a un creciente número de empresas que han estado buscando un método para facilitar el proceso de toma de decisiones empresariales, el poder de la línea de fondo, y lograr una organización totalmente coordinada, llamado "Business Intelligence" (BI). El BI tradicional tiende a ser inmanejable, arriesgado y costoso, especialmente para las pequeñas y medianas empresas (PYME). El surgimiento de la Cloud Computing y SaaS ofrece una solución rentable, estas aplicaciones de inteligencia de negocios que son provistas a través de SaaS son denominadas como "Business Intelligence as a Service (SaaS BI)", esto ha demostrado ser la próxima generación en el mercado de BI; sin embargo, como SaaS BI se encuentra en una etapa embrionaria, este trabajo intenta dar un marco de referencia para la adopción de un modelo de trabajo unificado. Aquí se puede observar como el modelo de servicio es tomado en cuenta por diferentes áreas de especialización como una metodología de trabajo válida, tanto sea para la reducción de costos como para la simplificación de las aplicaciones de usuarios.

El SaaS a menudo adopta la arquitectura de múltiples usuarios (multi-tenancy - MTA). Sin embargo, la construcción de una aplicación SaaS MTA requiere un esfuerzo significativo, ya sea que fuese construida desde cero o utilizando plataformas ya existentes, como "Force.com" o "Google App Engine". Es por esto que en el trabajo de [Tsai et al, 2011] se presenta a "EasySaaS" un FW de desarrollo SaaS diseñado para simplificar la construcción de aplicaciones SaaS. "EasySaaS" ofrece dos alternativas para construir una aplicación SaaS:

a. Permite publicar los requisitos de la aplicación, así como los scripts de prueba y dejar a los proveedores de SaaS la personalización de sus soluciones SaaS para satisfacer las necesidades de los usuarios.

b. Permite a los usuarios componer la aplicación utilizando plantillas proporcionadas en "EasySaaS".

Este FW reduce la carga de trabajo de los desarrolladores en las empresas consumidoras de SaaS, y proporciona un método sencillo para la personalización según las necesidades de los proveedores en un espíritu de colaboración. La mayoría de los servicios de aplicaciones independiente como el

conocimiento del dominio se almacenan las ontologías para apoyar el desarrollo de varios dominios. Este trabajo trata de presentar un concepto de software colaborativo a los entornos de SaaS que permitirán, en el caso de crecer, que proveedores no solo sean sustentadores del servicio sino que puedan empezar a tener conocimiento del dominio en cuestión para permitir dar servicios más personalizados y a menor costo.

En el trabajo realizado por [Zhang et al, 2009] se presenta una plataforma SaaS habilitada para la Web y SOA, que se denomina "GridSaaS"; aquí se puede observar que la integración de el SaaS más la utilización de tecnologías Web permiten dar una solución más completa y compleja a los usuarios.

El SaaS se está convirtiendo en un modelo de entrega de software muy común, especialmente para el software empresarial, el cambio al modelo SaaS está ocurriendo rápidamente; en el trabajo de [Tang et al, 2010] se hace referencia a el advenimiento de ecosistemas SaaS más complejos, incluyendo el servicio de gestión de ciclo de vida, relaciones comerciales y gestión de procesos, cobro y facturación para todas las funciones en el ecosistema y así sucesivamente. Este artículo describe las investigaciones sobre SaaS para su gestión y propone un marco de apoyo a los diferentes modelos de negocio, así como arquitecturas de múltiples aplicaciones. El FW abarca tanto los requisitos técnicos y de negocios y no toma en cuenta las consideraciones funcionales, tales como la disponibilidad, seguridad, escalabilidad y rendimiento.

El proceso de monitoreo de aplicaciones en modo SaaS es un aspecto en el cual se ha trabajado y se han planteado ciertos modelos como en [Lee & Hur, 2011], este plantea que el monitoreo ayuda a comprender la utilización de recursos en la computación en nube y detectar el patrón de uso de los usuarios de servicios diversos. Como el SaaS permite a los usuarios personalizar servicios de software según sus necesidades, estos tienden a volverse cada vez más complejos; debido a la complejidad de servicios personalizados, la necesidad de información en tiempo de ejecución y la soportar ambientes de múltiples usuarios va en incremento. Aquí se propone una arquitectura para el seguimiento y la gestión de SaaS en tiempo de real, este intenta presentar un modelo de seguimiento en el cual se permita la personalización del servicio y donde el proveedor del mismo pueda seguir manejando el modelo.

Es de mencionar que el procedimiento de administración y gestión de una plataforma SaaS, tiene la necesidad de tener que adaptarse a las diferentes personalizaciones creadas por el usuario de los servicios y a la vez mantener una interfaz común simple para su uso.

Recurrentemente en la bibliografía relevada se observa la necesidad de mantener un nivel SLA adecuado, pues el éxito de este modelo de servicios solo puede existir en función de la certeza por parte de los usuarios que al construir una solución basada en servicios externos que utilicen el Cloud Computing debe estar siempre disponible.

En tal sentido en el trabajo propuesto por [Pervez et al, 2010], hace referencia a la disponibilidad de potencia de procesamiento y la capacidad almacenamiento en el Cloud Computing; es por esto que los grandes jugadores de la industria informática como ser Google y Microsoft están proporcionando su próxima generación de productos a través Internet. Con el éxito de Amazon Cloud Computing toda la industria está poniendo atención en adoptar una solución de este tipo para ser provista para sus usuarios.

El aumento de las capacidades de procesamiento de los servidores no significa que los servicios alojados pueden manejar infinita carga de ejecución, este procesamiento creciente y la capacidad de almacenamiento debe ser utilizada con eficacia, es por esto que los autores de este trabajo intentan detallar los beneficios de su uso en aplicaciones en el mundo real.

Al tratarse de un nuevo modelo de trabajo, hay una necesidad de establecer pautas para este nuevo paradigma y haciendo el esfuerzo por dejar de lado ciertos mitos acerca de la capacidad ilimitada de procesamiento y almacenamiento, que podrían generar falsas expectativas y en consecuencia malos desarrollos de aplicaciones; se ha propuesto un FW de validación que ayudará a limitar la carga de procesamiento en servicios del modelo SaaS alojados en un Cloud Computing, este FW funciona mediante la validación de la solicitud de entrada en la semántica del contexto lo que ayuda en la entrega del proveedor de servicios de calidad de servicio, ajustándose su SLA.

Esta temática de reducción de costos es también tratada en [Lee & Choi, 2012], donde se detallan las características de SaaS como un modelo de entrega de software en el que los recursos de software son accedidos en forma remota por los usuarios; las empresas podrán encontrar en SaaS una alternativa válida para bajar sus costos. Pero es necesario hacer notar que esta tecnología requiere compartir servidores de aplicaciones entre múltiples usuarios y poder mantener por parte de los proveedores bajos costos de operación, además de la distribución de los servidores de aplicaciones, se necesitan satisfacer las necesidades de cada inquilino.

Es de hacer notar que en este trabajo se describe FW un marco Web de múltiples usuarios para el uso de SaaS. El marco propuesto admite personalizaciones de tiempo de ejecución de interfaces de usuario y la lógica de negocio mediante el uso de espacios de nombres a nivel de archivo, herencia y polimorfismo; pero la principal ventaja de esto, es que toda su funcionalidad es manejada a través

de tecnologías Web, por esta razón que este FW resulta de interés para la comunidad involucrada en el desarrollo de aplicaciones en base a SaaS.

Siguiendo esta línea de investigación y haciendo referencia a la reducción de costos, el trabajo propuesto por [Cai et al, 2009] hace mención al SaaS como una innovación tecnológica, así como un modelo de negocio que ofrece nuevas oportunidades a las pequeñas y medianas empresas (PYME); aquí se proponen diferentes métodos de uso de los servicios SaaS dependiendo del perfil del cliente, el coste de los recursos necesarios y el precio que los clientes están dispuestos a pagar.

La tecnología SaaS permite la utilización del servicio a través de múltiples clientes, lo que significa la necesidad de dar soporte a los múltiples clientes sobre la plataforma SaaS al mismo tiempo. En [Cai et al, 2009] se describe una metodología, junto con un conjunto de herramientas que soporta la granularidad fina para el trabajo con múltiples usuarios, esto permite utilizar una aplicación de instancia única para apoyar múltiples usuarios, un enfoque que podría abaratar los recursos de infraestructura del Cloud necesarios para apoyar un gran volumen de clientes. Además, la metodología puede disminuir el nivel de entrada de los operadores SaaS, así como las tasas de suscripción de usuario.

En el trabajo antes referenciado se hace la introducción a un tema que resulta ser central en la infraestructura necesaria para el SaaS, es la capacidad por parte de los proveedores de los servicios de poder generar una aplicación con una única instancia y dar servicios a múltiples usuarios, esto trae aparejado la reducción significativa de la cantidad de hardware involucrado para el un nivel de servicios acorde a lo definido en el contrato de SLA firmado con el usuario.

Esta línea también se sigue en [Guo et al, 2007], donde se hace referencia a la tecnología para múltiples usuarios haciendo notar que es una de las competencias clave para los servicios a través de Internet, para lograr un mayor margen de beneficio económico mediante el aprovechamiento de las economías de escala.

Este artículo explora las necesidades y los retos de la utilización de patrones que soporten multiplex usuarios y detalla el potencial de servir a un gran volumen de clientes al mismo tiempo. Ofreciendo un marco de servicios de múltiples clientes comunes, para ayudar a las personas a diseñar e implementar una alta calidad aplicaciones concurrentes en forma más eficiente. Debido al requisito esencial para garantizar la calidad del servicio con eficiencia de alta participación, se presentan enfoques y principios para apoyar mejores aislamientos entre los usuarios relacionados a la seguridad, el rendimiento, la disponibilidad y administración.

El SaaS ofrece a los proveedores de software de aplicación un modelo basado en Web entrega para gran cantidad de clientes con una infraestructura de múltiple usuario y una arquitectura basada en la compartición de aplicaciones con el fin de obtener un gran beneficio de la economía de escala.

Aunque la aplicación SaaS se suele desarrollar con todas las funciones de software altamente estandarizados para servir a tantos clientes como sea posible, muchos clientes siguen pidiendo para las variantes de función de acuerdo a sus necesidades empresariales únicas a través de una fácil configuración y personalización. Es por esto que el trabajo de [Sun et al, 2008] hace referencia a un modelo basado en suscripción, los proveedores de SaaS necesita tener una estrategia bien diseñada para permitir auto-servicio de configuración y personalización por parte de sus clientes sin tener que cambiar la aplicación SaaS código fuente para cualquier cliente individual.

Aquí se presentan los problemas de configuración y personalización y desafíos para los proveedores de SaaS, aclarar la diferencia entre configuración y personalización permite sientan las bases para un modelo de competencia y un marco metodológico que ayuda a los proveedores de SaaS para planificar y evaluar sus capacidades y estrategias para la configuración del servicio y personalización.

Otro aspecto importante donde la comunidad científica relacionada al SaaS ha trabajado, es la privacidad de los datos, al tratarse de servicios de uso extensivo a las empresas, estas son reticentes a exportarlos, es por esto que el trabajo realizado por [Zhang et al, 2009] se menciona la confiabilidad como el mayor desafío en la amplia aceptación de SaaS, ante la falta de confiabilidad en las aplicaciones SaaS, esta confiabilidad está dada en el no interrupción del servicio y la privacidad de datos, este último siendo el tema principal y el más importante para los usuarios.

Cómo proteger la privacidad de los datos cuando el servicio de software y base de datos están albergados en servidores del proveedor es todavía una cuestión abierta. En este trabajo se introduce el tema de la personalización de aplicaciones SaaS y base de datos compartida, se presenta un modelo de datos de almacenamiento compartido donde se define tres tipos de restricciones de seguridad comunes a todos los usuarios y luego propone una restricciones de seguridad personalizable basado en la preservación de la privacidad de datos mediante la combinación de cifrado de datos y la información de disociación.

La cuestión del manejo de datos también se puede observar en [Rimal & El-Refaey, 2010] donde se hace referencia a crecimiento volumétrica de los datos está aumentando día a día. La necesidad de computo se volviendo masivo y se necesita de sistemas escalables y eficientes, la gestión de datos es una etapa importante para acelerar el de procesamiento. Aquí se hace una diferencia entre el

trabajo de datos en ambientes científicos y empresariales, es por esto que se hace referencia aquí a la programación de los algoritmos de flujo de datos a los procedimientos de operaciones y procesamiento masivo de datos.

El estudio de los flujos de trabajo en ambientes científicos en contexto de múltiple usuarios permite la orquestación particular de un control de flujo y de datos, nuevos requisitos en el desarrollo de sistemas han hado paso a la generación de nuevos servicios. Se ha explorado un marco de flujo de trabajo científico para múltiple usuarios medio ambiente Cloud Computing y servicios SaaS; el aporte de mayor significación de este trabajo es el reconocimiento de características particulares para el procesamiento de datos en ambientes científicos y dando un marco de trabajo acorde a los lineamientos detectados.

El manejo de licenciamientos es otro aspecto tratado, como se puede ver en [Liao, 2009] donde se menciona la diferencia entre el modo de licencia de software para el modo tradicional de licencia perpetua de uso y las características de los usuarios de SaaS donde podrían utilizar cualquier software que ellos necesitan a través de las suscripciones de los proveedores de servicio o software. En este trabajo por medio del método de análisis comparativo, entre una arquitectura de software basada en SaaS y una arquitectura no basada en servicios se intenta dar luz sobre las principales características de este nuevo esquema de licenciamiento, haciendo hincapié en los nuevos desafíos que deben enfrentar los proveedores de servicios para mantener un esquema de licencias acorde a lo esperado por los usuarios.

En [Zhang et al, 2009] se presenta un modelo SaaS de aplicación que se ha diseñado y entregado en el modelo de madurez de alto nivel, a fin de realizar la configuración, los metadatos se utiliza para definir todos los puntos de variabilidad de la aplicación; mientras se utiliza un modelo para gestionar los metadatos, estos pueden ser cambiados en caliente y desplegados inmediatamente en tiempo de ejecución; la escalabilidad se discute tanto en la capa de aplicación y la capa de datos. Aquí se puede observar como los mecanismos de control no solamente de servicio en sí, sino de la infraestructura son considerados en el modelo de SaaS con la utilización de metadatos del ambiente.

El flujo de trabajo para SaaS permite a las organizaciones de todos los tamaños construir, ejecutar, gestionar y desarrollar sus propias aplicaciones orientadas a los procesos de negocio. En [Xiao et al, 2010] se muestra un proceso de personalización flexible de flujo de trabajo basado en SaaS para resolver el problema causado por la orquestación de procesos de negocio basado en procesos de definición de lenguaje XML (XPDL); permite a múltiples usuarios personalizar los procesos y la asignación de tareas manuales durante el momento de personalización es el centro de este trabajo;

en tiempo de ejecución del proceso, el sistema puede obtener de forma dinámica las reglas y asignar la tarea al participante automáticamente de acuerdo con las normas establecidas por el usuario del sistema.

El manejo del pago de los servicios ha sido tratado en [Pathirage et al, 2011], este modelo de servicio promete un ahorro de costos para los usuarios finales, permitiéndoles externalizar sus funciones no críticas de negocio a terceros al estilo "pay-as-you-go". Sin embargo, para poder trabajar con este modelo es necesaria una capa intermedia de aplicación que maximice la participación y colaboración entre usuarios. Ambientes de múltiple usuarios que permitan a los arrendatarios múltiples, es decir usuarios, poder compartir una única instancia de aplicación de forma segura, es un factor clave para la construcción de esta capa intermedia de aplicación.

En este trabajo se presenta el diseño y la arquitectura de un motor de flujo de trabajo múltiple usuarios mientras se discuten en detalle los casos de uso potencial de este tipo de arquitectura. Se puede hacer notar que el aporte principal de este trabajo está direccionado al flujo de trabajo en ambientes múltiple usuarios y el diseño e implementación del motor de flujo de trabajo múltiple usuarios que permite a varios clientes para gestionar sus flujos de trabajo de forma segura dentro de la instancia del mismo motor.

En los últimos años el desarrollo de la industria de SaaS se ha mantenido muy activa y con el desarrollo y la promoción de la computación en nube, cada vez más personas llegan a creer SaaS se convertirá en el modelo futuro de la industria del software. Como se trata en [Zhao & Lui, 2011] actualmente la arquitectura y marco de desarrollo son de propiedad, faltan normas uniformes, además las aplicaciones son difíciles de personalizar y montar. En vista de esta situación, en este trabajo se describe un marco orientado a componentes SaaS de integración basado en el lenguaje de negocios abierto de la comunidad de Apache (OFBiz).

En un servicio basado en múltiple usuarios para una aplicación SaaS, se presenta el problema que el número de servidores en los casos de servicios Web se implementan son limitados y los usuarios comparten la misma aplicación y servicios. Con el propósito de reducir los costos de propiedad elevadas economías de escala, hay que resolver el problema de cómo colocar de manera óptima los usuarios para maximizar el número total de usuarios sin violar su acuerdo SLA, este es el enfoque que se ha abordado en [Yang et al, 2011], aquí se propone un enfoque híbrido para resolver la colocación de los usuarios que se llama estrategia de "Colocación de Usuarios (TPS)"; el TPS utiliza una combinación del modelo de estimación de consumo de recursos, selección de servicios

con el algoritmo genético (GA), el enfoque de razonamiento basado en casos (CBR) y heurística. CBR se propone para hacer coincidir los planes existentes de ejecución que se generan por la GA.

Con el fin de hacer pleno uso de todos los tipos de recursos de los servidores, un enfoque heurístico se propone para seleccionar el plan de ejecución óptima en función de la distancia del vector inquilino de consumo de recursos y el vector de recursos del servidor residual. Los resultados de los experimentos simulados muestran que la estrategia propuesta en este trabajo es eficaz en la colocación de los inquilinos. Este trabajo encara el uso de métodos no convencionales de programación para la asignación de recursos, haciendo de este un trabajo novel y abriendo una nueva línea de investigación.

2.5.4. Introducción a la Arquitectura Orientada a Servicios

Surge una gran confusión en la distinción entre el software como servicio (SaaS) y arquitectura orientada a servicios (SOA) [Laplante et al, 2008]; en este trabajo se apela al modelo Zachman para ayudar a dar sentido los entornos de servicios Web y los servicios que constituyen la base tanto para SOA y SaaS. La diferencia entre SaaS y SOA es que el primero es un modelo de software de aplicación, mientras que el segundo es un modelo para la construcción de software y puede utilizar servicios SaaS.

Una manera de detectar las diferencias entre estos dos conceptos es utilizar el modelo arquitectónico Zachman. En este trabajo se da un marco donde se comparan los conceptos de SOA y SaaS, usamos el modelo Zachman para diferenciar los dos enfoques arquitectónicos; debido a que el modelo Zachman es intuitivo, el enfoque que tomado para describir las diferencias entre SaaS y SOA es un método valido para su uso.

Como se ha detallo anteriormente el SOA es un modelo de construcción de software que se asienta sobre los servicios SaaS, este concepto permite ahondar en la reducción de costos en relación a la infraestructura que la TI incurre como ser hardware, licencias de software, costos de mantenimiento y supervisión, administración y mantenimiento de la infraestructura de TI.

En [Azeez et al, 2010] se hace referencia a esta posibilidad de reducción de costos en relación a la utilización de SOA a través de servicios externos que se encuentran en Internet, específicamente el modelo denominado Cloud Computing, este modelo ofrece algunas perspectivas tangibles en la reducción de algunos de esos costos, sin embargo, las abstracciones proporcionados por el Cloud

Computing son a menudo insuficientes para proporcionar un importante ahorro de costos a través de la infraestructura de TI del ciclo de vida.

El punto de vista abordado en este trabajo es de sumo interés pues intenta remplazar para múltiples clientes, utilizar una sola aplicación para emular múltiples instancias de aplicaciones; al compartir una aplicación a través de muchos usuarios (multi-tenancy) se intenta remplazar muchas instancias de aplicaciones pequeñas con uno o pocas aplicaciones reduciendo así el costo total de la infraestructura de TI. Aquí se presenta una arquitectura para lograr multi-tenancy a nivel de SOA, que permite a los usuarios ejecutar sus servicios y otros artefactos SOA en un contexto multi-tenancy, así como proporcionar un entorno para construir aplicaciones multi-tenancy, se discute la arquitectura, las decisiones de diseño y los problemas encontrados junto con las posibles soluciones en caso de corresponder.

En estas últimas referencias se puede observar como la evolución que ha tenido los servicios accesibles a través de Internet han llegado a este momento en el punto donde se provee un modelo de construcción de aplicaciones basado en servicios, es decir SOA; pues el aumento de disponibilidad de nuevos servicios y mayor complejidad hace que los arquitectos de sistemas deban tener guía para la repetición de las buenas prácticas de técnicas probadas, esto como se detalló en párrafos anteriores es muy similar a la definición de patrones, con lo cual se puede concluir que el proceso de aprendizaje de la comunidad de sistemas es cíclico ante un nuevo avance se realizan los ajuste necesarios y sobre esto se construye la nueva capa hasta llegar a un punto de madurez de la tecnología en donde se puede empezar a repetir los procesos exitosos; esto es el concepto subyacente en la construcción de patrones que pueden esta definidos como es el caso del presente trabajo a nivel arquitectónico utilizando el estado del arte con respecto a la utilización de SaaS y SOA.

2.6. DISCUSIÓN MARCO

En esta sección se presenta la evolución de los patrones a las arquitecturas (sección 2.6.1.) y el software como servicio y usabilidad (sección 2.6.2).

2.6.1. EVOLUCIÓN DE LOS PATRONES A LAS ARQUITECTURAS

Como se ha definido en [Cysneiros & Kushniruk, 2003] las técnicas probadas en usabilidad han pasado a constituir un conjunto de patrones. Siguiendo con esa línea de pensamiento los patrones pueden evolucionar a modelos más complejos; estos modelos en la actualidad están orientados al SaaS, esto se debe al conjunto de ventajas que se ha detallado en [TraceOne, 2012].
A su vez las empresas que comienzan a migrar sus aplicaciones a instalaciones que soportan el modelo Software como Servicio (SaaS), aquí surgen nuevos retos de integración, estas aplicaciones deben interactuar entre sí y con las aplicaciones existentes. Para hacer frente a los desafíos de la integración del software existente y el SaaS se debe contar con una arquitectura acorde a las nuevas necesidades [Kolb, 2009].

2.6.2. SOFTWARE COMO SERVICO y USABILIDAD

En este apartado se detalla la utilización de servicios en la modalidad de SaaS para proveer funcionalidades de usabilidad en aplicaciones anfitrionas.
En función de lo relevado por el autor del presente trabajo, esta idea resulta ser novel pues no ha sido posible detectar trabajos anteriores en relación a la funcionalidad de UNDO/REDO dentro del marco de usabilidad de software, siendo esta extraída de la aplicación para ser depositada en un servicio externo y que la aplicación en cuestión haga uso del mismo según sea su conveniencia.
Es válido mencionar que durante el proceso de búsqueda se ha encontrado referencias relacionadas con la usabilidad de los servicios en la modalidad de SaaS. Estos trabajos lo que intentan dar luz es sobre cual es el mejor modelo de usabilidad que debería ser utilizado para presentar un servicio a un usuario, aquí se puede trazar un paralelo con los trabajos existentes con aplicaciones Web de uso común.

3. DESCRIPCIÓN DEL PROBLEMA

En este Capítulo se presentan el contexto del problema (sección 3.1), se define el problema (sección 3.2), y se plantean las correspondientes preguntas sumario de investigación (sección 3.3).

3.1. CONTEXTO DEL PROBLEMA

Los principios de usabilidad son un conjunto de buenas prácticas de software, las cuales contribuyen a que una aplicación posea un grado de interacción acorde a las características y expectativas del usuario. Salvo excepciones, a este conjunto de buenas prácticas se las considera en un estadio avanzado del ciclo de desarrollo de software; con lo cual, se las suele dejar de lado y no incluirlas en la aplicación o posponer su implementación para próximas versiones del artefacto o producto software.

En los casos en los que se las incluye, esto suele hacerse en forma parcial debido a la complejidad de las mismas, generando de esta manera un incremento en los costos del proyecto al sumar nuevos requerimientos hacia el final del ciclo de desarrollo.

3.2. DESCRIPCIÓN DEL PROBLEMA

En el contexto de los patrones de usabilidad este trabajo se centra en la funcionalidad de UNDO/REDO. Para un sistema software la inclusión de la funcionalidad UNDO/REDO puede ser una funcionalidad central o ser una funcionalidad deseable.

En caso de que la funcionalidad sea de carácter central, un diseñador contempla la construcción de esta en el cuerpo principal de la aplicación; debiéndose destacar, que estas son aplicaciones que no pueden ser concebidas sin la existencia de la funcionalidad de UNDO/REDO. Cabe incluir dentro de esta categoría a los procesadores de texto, un usuario de este tipo de sistemas espera la existencia de la misma. En este sentido, se considera de interés hacer referencia a todo un conjunto de

aplicaciones software que enmascaran un procesador de textos, tales como planillas de cálculo, gestores de correos electrónicos, sistemas de mensajería instantánea, entre otros.

En caso de que la funcionalidad sea de carácter deseable, el diseñador se enfrenta con la dificultad de tener que incluir esta funcionalidad para que la aplicación sea completa y robusta desde el punto de vista de la usabilidad. Asimismo cabe señalar, que esto insume tiempo y esfuerzo para una funcionalidad que no es el núcleo de la aplicación. Dentro de esta categoría se enrolan aplicaciones como gestores de administración de base de datos, sistemas de carga de datos con interfaz de usuario, entre otros. Tal como se puede observar, en esta categoría los usuarios del sistema manejan el ingreso de datos, en consecuencia es altamente probable que se generen errores en el proceso de carga de los mismos, por lo antes mencionado, los usuarios desearían poder corregir esto rápidamente, con lo cual se podría inferir que la funcionalidad deseable.

En función de lo expuesto, es necesario incluir la funcionalidad UNDO/REDO, en todas las aplicaciones que manejen algún tipo de interfaz de usuario para que la aplicación sea robusta; no solo desde el punto de vista de la usabilidad de software, sino desde la ingeniería del software en general, dado que la calidad de los datos hace al núcleo de cualquier sistema.

Por consiguiente, el problema abierto que se pretende abordar en este trabajo de tesis en referencia a la funcionalidad de UNDO/REDO, consiste en la falta de un proceso definido para su inclusión dentro de un artefacto software.

Esta falta de proceso origina que la aplicación en cuestión, si bien puede estar en condiciones de llevar a cabo las tareas para las cuales fue diseñada, no posea la usabilidad adecuada; en otras palabras, se resiente la facilidad con la que el usuario debe interactuar con el sistema, no pudiéndose obtener todo el potencial que la aplicación podría darle.

En esta instancia se estima que es necesario desarrollar un breve ejemplo de la importancia que la usabilidad posee en el uso o no de un sistema software. En tal sentido, se hace referencia al período que comprende desde mediados de la década del 80 hasta principios de los 90.

Por esos años, el sistema operativo para computadores de rango medio y chico era el Unix, específicamente para los procesadores Intel la versión distribuida por Santa Cruz Operation (SCO) [SCO, 2012]. Dentro de este período, hace su aparición la primera versión del sistema operativo de la empresa Microsoft DOS (MS-DOS) [MS-DOS, 2012], el cual prestaba muchas menos funcionalidades que SCO, y las que proveía, no contemplaban la totalidad de alternativas de SCO. A este hecho cabe añadir, que también es posible hacer mención a cuestiones de estabilidad y

robustez del sistema operativo, SCO era un sistema probado en cientos de instalaciones medianas donde eran soportadas aplicaciones críticas para las organizaciones.

En síntesis, de ante mano se podría hacer mención a que SCO posee importantes ventajas frente a MS–DOS, sin considerar cuestiones de mercadotecnia. No obstante, y de acuerdo al entender del autor del presente trabajo, el factor usabilidad fue lo que inclinó la balanza en equipos de rango chico y medio hacia el MS-DOS, dado que éste, a pesar de su muy reducida funcionalidad, era sumamente sencillo de instalar y su uso era también mucho más simple y claro que el SCO. La característica citada, hacía posible que un usuario con poca experiencia pudiera trabajar con él sin la necesidad de tener que realizar un curso de capacitación para su uso.

Por su parte, el MS-DOS se reducía a un conjunto simple de tareas con pocos parámetros, lo cual permitió a Microsoft constituirse en la compañía que es hoy; es decir, comprender el concepto de usabilidad, el cual se basa en consideraciones intuitivas, lo que como consecuencia trae aparejado simpleza de uso permitiendo al usuario tener una curva de aprendizaje no pronunciada.

Continuando con este proceso de observación y análisis de los productos que la compañía Microsoft ha desarrollado durante todos esto años, cabe señalar que su filosofía de potenciar la usabilidad en sus productos no ha sufrido cambios hasta el día de hoy; y si lo extendemos a la actualidad, se puede observar que el sistema operativo Linux [Linux, 2012], en sus distribuciones más populares, expone como uno de sus principales argumentos su similitud de interfaz a el sistema operativo Windows.

A modo de cierre, se estima de interés citar el caso de otra compañía de relevancia, tal como es el caso de Apple [Apple, 2012]. Cabe resaltar, que esta empresa no ha inventado en términos generales la tecnología que ha puesto en sus productos, pero si les ha proporcionado un orden que hasta el momento no había sido visto; además de generar nuevos canales de venta y nuevos mercados, estos dos últimos aspectos que no son relevantes a este trabajo.

La característica más saliente de los productos de Apple, como ser iPhone, iPad, entre otros, consiste en el diseño del producto, el cual se sustenta sobre dos pilares: cuestiones estéticas de diseño y cuestiones de usabilidad de software; todos sus productos son de carácter intuitivo en lo que se refiere a su uso, siendo este uno de los principales argumentos de la usabilidad.

Se han presentado dos ejemplos que permiten dar justificación al problema abordado; y tal como se ha detallado anteriormente, la usabilidad es de suma importancia para una aplicación software, y se detecta que aspectos de la usabilidad tan elementales como la funcionalidad UNDO/REDO no

poseen el grado de madurez esperado para su inclusión en una aplicación. En este sentido, es de suma importancia construir la pirámide de la usabilidad en el modo correcto, solucionando primero los aspectos básicos, como ser dar solución a la funcionalidad UNDO/REDO entre otros, para poder avanzar luego hacia un proceso de usabilidad unificado.

3.3. PREGUNTAS SUMARIO DE INVESTIGACIÓN

En el presente trabajo se intentará dar respuesta a las siguientes preguntas de investigación disparadores:

Pregunta 1: ¿Es posible desarrollar un proceso para la inclusión de la funcionalidad UNDO/REDO que sea simple y rápido de incluir en una aplicación nueva o existente?

Pregunta 2: ¿Es posible definir una abstracción para el modelo de datos a ser manejada por la funcionalidad de UNDO/REDO?

4. SOLUCIÓN PROPUESTA

En este capítulo se presenta la solución propuesta al problema introducido. Se dan las aproximaciones de solución consideradas (Sección 4.1) entre las que se discuten; Aproximación Ad Hoc (Sección 4.1.1), Aproximación Basada en Patrones (Sección 4.1.2), Aproximación Basada en Software como Servicio (Sección 4.1.3), Aproximación Basada en Micro-Framework (Sección 4.1.4), Aproximación Basada en Procesos (Sección 4.1.5), Procesos y Servidor de Proximidad (Sección 4.1.6); introducción de definiciones (Sección 4.2), Definición de Artefacto de Usabilidad UNDO/REDO (Sección 4.2.1), Descripción formal (Sección 4.2.2), se introduce el concepto de Unidad Lógica de Cambio (Sección 4.2.3) y los Puntos de No Retorno (Sección 4.2.4), se introduce el proceso propuesto para la Inclusión de Servicios en Aplicaciones Anfitrionas Basado en Patrones de Usabilidad (Sección 4.3) para cada fase de este se definen las entradas, las salidas, potenciales problemas que se pueden presentar, y soluciones propuestas. Se presentan las técnicas propuestas asociadas a las tareas del proceso (Sección 4.4).

4.1. APROXIMACIONES DE SOLUCION CONSIDERADAS

En esta sección se detallan distintas aproximaciones para satisfacer la funcionalidad de UNDO/REDO, en esta se realiza una comparación entre ellas y se fija posición ante cada una, las misma son: Ad Hoc (Sección 4.1.1), Patrones (Sección 4.1.2), Software como Servicio (Sección 4.1.3), Micro-Framework (Sección 4.1.4), Procesos (Sección 4.1.5), Procesos y Servidor de Proximidad (Sección 4.1.7).

4.1.1. Aproximación Ad Hoc

Es la inclusión de la funcionalidad dentro de una aplicación escribiendo todo el código necesario, no se reutiliza código generado anteriormente, ni son utilizadas las lecciones aprendidas al construir otras aplicaciones, como consecuencia de esto se dimensiona la funcionalidad UNDO/REDO según las necesidades propias de cada aplicación en desarrollo. El punto a favor de esta alternativa es que la funcionalidad para cada aplicación es única y se pueden realizar todos los considerandos particulares, obteniendo de esta forma una relación código y funcionalidad óptimos, es decir, solo se agrega estrictamente los requerimientos de la funcionalidad UNDO/REDO propios de la aplicación en desarrollo, el punto en contra de la misma es, cada vez que se realiza una

aplicación se debe rescribir todo el código nuevamente, esto provoca una mayor propensión a la generación de errores de codificación, a su vez la experiencia adquirida de otras aplicaciones no es transferida en su totalidad pues el desarrollador intenta adaptar la funcionalidad de UNDO/REDO a cada nueva aplicación, perdiendo en muchos casos valiosas lecciones aprendidas.

Esta es una metodología de trabajo ampliamente utilizada para la inclusión de requisitos de usabilidad dentro de una aplicación, siendo esta la razón por la cual el uso de estructuras y procesos sólidos en el área de la usabilidad de software no se ha extendido como otras áreas de la ingeniería del software en general.

4.1.2. Aproximación Basada en Patrones

Se incluye la funcionalidad dentro de una aplicación siguiendo los lineamientos de una solución probada, todo el código se incluye en la aplicación anfitriona. La utilización de patrones es un avance sustancial con respecto a la solución Ad Hoc, pues la experiencia adquirida en desarrollos anteriores ha sido sistematizada y normalizada en un esquema para que la funcionalidad pueda ser utilizada en todas las aplicaciones, realizando las modificaciones mínimas que sean necesarias. La desventaja de esta alternativa de diseño consiste en la inclusión de toda la complejidad de la funcionalidad UNDO/REDO, lo cual implica un considerable incremento en la cantidad de líneas de código de la aplicación en cuestión. Esta alternativa es viable en los casos en que la funcionalidad de UNDO/REDO se erige en un elemento central de la aplicación en cuestión. En tal sentido, se puede citar, como ejemplo para este caso a un procesador de texto, así mismo cabe destacar que una aplicación sin esta clase de de funcionalidad sería de difícil uso.

Esta aproximación intenta acercar el campo de la usabilidad a las reglas establecidas en otras áreas de la ingeniería del software; en este sentido, los patrones que implementan la funcionalidad UNDO/REDO, quizás provean más funcionalidad de la necesaria para la aplicación. Esto hace que se tienda a modificar el patrón para adaptarlo a la aplicación en desarrollo convierte al modelo de patrones a un modelo Ad Hoc; esto implica que su puesta a punto demande tiempo y esfuerzo por parte de los desarrolladores de la aplicación, diluyendo la principal característica que se le atribuyen a los patrones, que es confiabilidad y rapidez de implementación.

4.1.3. Aproximación Software como Servicio

En la aproximación software como servicio (SaaS) la aplicación anfitriona incluye solamente una invocación para determinada funcionalidad a través de un servicio externo, toda la complejidad del mismo se encuentra encapsulada dentro de este servicio y no en la aplicación.

Esta solución tiene la ventaja de reducir la cantidad de código que es necesario dentro de la aplicación anfitriona, minimizando la cantidad de posibles errores de codificación, además permite invocar las características requeridas para cada aplicación, pues el servicio tiene el conjunto máximo de características de la funcionalidad de UNDO/REDO y mediante una consola de administración se puede configurar que características de la funcionalidad necesita la aplicación.

Dentro de las ventajas que se pueden observar en esta aproximación existe la posibilidad de administrar perfiles de usuarios con lo cual una misma aplicación anfitriona podría permitir a cierto grupo de usuarios acceder a la funcionalidad de UNDO/REDO y a otro grupo no. También se puede redefinir las características de la funcionalidad de manera sencilla a través de la interfaz que provee el servicio.

Como consecuencia de la administración centralizada de la funcionalidad de UNDO/REDO se posible mantener un historial de las modificaciones realizadas por interfaz de usuario, de manera tal que el servicio esté en condiciones de añadir un módulo de reconocimiento de patrones, el cual podría generar agrupaciones de ocurrencias similares en el uso de la aplicación para la funcionalidad de UNDO/REDO. Dichas agrupaciones serían las encargadas de orientar a los administradores para generar ciertas correcciones pos implementación del sistema o detectar la necesidad de reforzar o generar nueva capacitación a los usuarios del mismo.

Esta aproximación permitiría contemplar la posibilidad de tener un almacenamiento de pilas de modificaciones persistentes; es otras palabras, los usuarios poden mantener las modificaciones realizadas a través de distintas sesiones de acceso. En tal sentido, es necesario resaltar que dicho almacenamiento está relacionado a las posibilidades de la aplicación anfitriona con respecto a la persistencia de sus datos y la validez temporal de los mismos.

Otra extensión de este modelo es la generación de patrones de la funcionalidad UNDO/REDO asociado con determinadas interfaces de usuario; hecho este, que puede ser explicado de la siguiente forma: al asociar interfaces de usuario con requisitos de usabilidad de UNDO/REDO esto

permite que a medida que se utiliza el servicio, la configuración de nuevas aplicaciones sea más rápida.

Como toda solución posee sus desventajas, la misma está asociada a un quiebre con respecto al método de inclusión que será utilizado en la aplicación anfitriona. Esto consiste en utilizar un modelo que se enrole estrictamente en la arquitectura SaaS, en virtud de la cual, el desarrollador de la aplicación debe realizar las llamadas al servicio según su leal saber y entender. Cabe señalar que el modelo presenta su punto débil, dado que es necesaria de una capacidad no auditada por el mismo para que este pueda funcionar.

4.1.4. Aproximación Basada en Micro-Frameworks

Los Micro-Framework, o mínimo entorno de trabajo, son un conjunto de rutinas encapsuladas dentro de una interfaz programable de aplicaciones (API por su siglas en inglés), los cuales tienen un cierto orden de invocación; este es denominada micro pues su alcance está limitado a uno o pocas funcionalidades dentro de una aplicación a diferencia de otros Frameworks que abarcan la totalidad de la construcción de una aplicación.

En este caso pueden ser combinados dentro de esta estrategia la utilización de patrones y SaaS sumado a un orden de llamadas a las rutinas del SaaS, el avance con respecto a las demás alternativas es que se vale de otras aproximaciones, además estructurar la forma de trabajo dejando un menor conjunto de acciones a ser realizadas por parte del desarrollador.

Como ejemplo y justificación de la validez de esta aproximación, se puede tomar como caso de estudio la utilización de APIs para el acceso de una base de datos, el motor de base de datos podría ser comparado con el servicio que encapsula la funcionalidad de UNDO/REDO. El API incluye un conjunto de patrones conocidos para el acceso de dase de datos, consultas y transformación de objetos a tablas relacionales, sumado a esto se provee de un instructivo para la correcta secuencia de llamada a las rutinas. Esto es lo mismo que se estaría aplicando al API donde se encapsula la funcionalidad de UNDO/REDO.

Esta línea de razonamiento oculta un error sutil, pero con grandes consecuencias en el proceso de inclusión de la funcionalidad UNDO/REDO dentro de la aplicación anfitriona; esta es el grado de madurez de la tecnología en cuestión, es decir la utilización de bases de datos, además de ser un

tema que lleva varios años en la ingeniería de software, es a la sazón un área muy tenida en cuenta en los planes educativos de las carreras de sistemas y la bibliografía existente es abundante.

Esto no es así en el dominio de la usabilidad de software, no se esta diciendo que no existan planes ni bibliografía, solo se está haciendo notar que el acceso a una base de datos es un tema que un desarrollador promedio conoce en mayor o menor profundidad, lo que no sucede con un desarrollador promedio con relación a la usabilidad de software en general y en particular con la funcionalidad de UNDO/REDO.

Esto hace que la utilización de un API unida a las demás aproximaciones antes referenciadas en este apartado sea una mejora con respecto a la utilización aislada de las otras aproximaciones, pero deja igualmente librada a la conciencia del desarrollador ciertos aspectos de diseño que pueden implicar el éxito o fracaso del modelo de inclusión del servicio.

4.1.5. Aproximación Basada en Procesos

Esta estrategia abarca además de la utilización de un Micro-Framework, incluye la utilización de un conjunto de pasos para detectar las características de UNDO/REDO necesarias para la aplicación anfitriona y un conjunto de tareas post implementación para el mejoramiento perfectivo de la misma; esta arquitectura podría ser enrolada como software orientado a servicios (SOA por sus siglas en inglés), esta es una manera de dividir el software en componentes, donde estos se comunican a través de la capa de red con el cliente y con otros servicios, de esta forma los servicios se pueden combinar en aplicaciones compuestas. En esta primera etapa se ha implementado el modelo de servicio para la funcionalidad UNDO/REDO de extender a otras funcionalidades del dominio de la usabilidad como ser el proceso de cancelar (Cancel) y el asistente (Wizard) se debería considerar este como un SOA para usabilidad de software.

En relación con la aproximación de procesos, la misma permite que el desarrollador deba seguir un plan de tareas, el cual de ser seguido, minimizaría la posibilidad de errores en la inclusión de la funcionalidad de UNDO/REDO dentro de una aplicación anfitriona; se entiende que no se reduce la probabilidad a cero, pero si se la reduce considerablemente.

Este proceso de inclusión sumado a la API, los patrones y al servicio generan un conjunto ordenado que permite que un desarrollador no experimentado en cuestiones de usabilidad, pueda tener las probabilidades a su favor durante el proceso de inclusión, pues este fue realizado en función de

casos de éxito para aplicaciones que anteriormente han usado el servicio, es decir se ha reutilizado no solamente código sino experiencia adquirida. Es esta aproximación donde se enrola la propuesta presentada en este trabajo de Tesis.

Se ha seleccionado un conjunto de características deseables para una implementación de la funcionalidad de UNDO/REDO en una aplicación anfitriona y se las ha comparado (tabla 4.1).

Estas características deseables han sido seleccionadas del ideal de la ingeniería de software tiene con respecto a una herramienta de desarrollo, estas son que permita ser repetible ni mayores esfuerzos, que pueda ser escalable tanto en aplicaciones pequeñas como grandes, que sea auditable que durante el proceso de construcción la solución propuesto incluya puntos de validación y la trazabilidad de todas sus operaciones, que sea perfectible, es decir que la aplicación mediante algún mecanismo, en este caso un proceso de minería de datos, pueda analizar su uso y mejoras al sistema, que sea transferible, esto está relacionado a la capacidad de que su uso no sea una ciencia solo conocida por un grupo pequeño de personas sino que el pasaje de conocimientos tanto se da en su uso como en su aplicación sea una cuestión trivial y por último integrable a otras plataformas y tecnologías tanto sea existentes como por venir en el futuro próximo.

Alternativa	Repetible	Escalable	Auditable	Perfectible	Transferible	Integrable
Ad Hoc	NC	NC	CP	CP	NC	NC
Patrones	CT	CP	CT	CT	CT	CP
SaaS	CT	CT	NC	NC	CT	CP
Framework	CT	CT	CP	CP	CT	CT
Proceso	CT	CT	CT	CT	CT	CT

Tabla 4.1. Comparación. (Escala= NC no cumple la condición, CP cumple parcialmente, CT cumple totalmente la condición)

Como se puede observar la implementación de un Proceso es la alternativa más sólida para el UNDO/REDO. La construcción de la funcionalidad de UNDO/REDO bajo la modalidad de servicio genera que toda la complejidad de la funcionalidad UNDO/REDO se ocultada por el servicio, la ventaja de esta aproximación es que permite construir sobre un servicio la complejidad que se desee, pues la misma puede ser reutilizada a través de distintas aplicaciones, con lo cual el esfuerzo de su construcción queda justificado.

4.1.6. Aproximación Basada en Procesos y Servidor de Proximidad

Como se definió anteriormente la implementación mediante un proceso, que a su vez utiliza un servicio es una opción válida; un aspecto que debe ser contemplado en la aproximación por proceso es la seguridad, este trabajo trata de presentar una solución de aplicación real en el mundo de la ingeniería de software; es por esto, que se debe hacer referencia al tipo de aplicaciones que serán desarrolladas que puedan utilizar un servicio de usabilidad en la modalidad servicio Web.

En la actualidad es ampliamente utilizada las tecnologías Web para el desarrollo de aplicaciones para Internet como así para Intranet, estas son ejecutadas dentro de un contenedor denominado Browser, este interpreta HTML y lo presenta en forma gráfica, además puede ejecutar programas en JavaScript y realizar invocaciones a los servidores de aplicación en la modalidad de asíncrona conocida comúnmente como AJAX.

En este entorno de trabajo un aspecto de seguridad que contemplan los Browsers es el conocido como la secuencia de comandos en sitios cruzados (XSS de sus siglas en inglés), esta falla de seguridad permite que al inocular en una página HTML a través de JavaScript una referencia a otro dominio, dirección URL, se pueda enviar información sensible a un sitio no autorizado sin que el usuario de la página se dé cuenta; es por esto, que los Browsers validan esta opción por defecto cerrando el camino para que se hagan referencias a distintos sitios del invocado en la página original.

Este aspecto de seguridad puede ser modificada en la opción de configuración de los Browsers, pero generaría un agujero de seguridad por lo cual muchas empresas descartarían la utilización del servicio Web de usabilidad por considerarlo no seguro.

Como respuesta a este problema se ha desarrollado la alternativa de uso de un servidor de proximidad de usabilidad, el cual es incluido en el proceso y puede ser utilizado o no según los lineamientos de seguridad que posea la empresa que consuma el servicio.

Como aspecto extra del uso de esta alternativa es, con la proliferación de servicios en la Web, se podría plantear el remplazo de proveedores de servicios sin mayores cambios en la aplicación. Esto que en teoría es una cuestión simple, en la práctica puede presentar problemas en su implementación, al no existir un modelo normalizado de comunicación y mensajería entre los diversos servicios que proveen distintos proveedores de aplicaciones, en el caso que nos incumbe servicios de usabilidad UNDO/REDO.

Para contemplar esta posibilidad se presenta la utilización de un servidor de proximidad el cual media entre la interfaz de la aplicación y los servicios que utiliza; esto trae como ventaja, la posibilidad de generar una separación física real entre el modo en que es mostrada la información al usuario y la aplicación en sí.

Comúnmente un servidor de proximidad es utilizado en las empresas para reducir el tráfico desde y hacia Internet desde la red local; en él se almacenan las páginas más accedidas, cuando un usuario intenta acceder a una página que ha sido recientemente accedida por otro usuario, este en vez de acceder a la página real le provee al usuario la que tiene almacenada, haciendo mucho más rápido el acceso y reduciendo el ancho de banda utilizado en la red, aquí se validan cuestiones tiempos de acceso, que son contemplados por los servidores de proximidad.

Como es mencionado en el capítulo anterior, existen algunos intentos para integrar IaaS bajo la modalidad de servidor de proximidad, de esa idea se puede extender el concepto de crear un servidor de proximidad especializado para la usabilidad de software. Este servidor integraría interfaces de usuario con funcionalidad del núcleo de la aplicación y los servicios de UNDO/REDO antes descripto.

En consecuencia un servidor de proximidad basado en usabilidad complementa el proceso antes descripto, permitiendo intercambiar servicios de distintos proveedores, sin afectar la interfaz de usuario ni el núcleo de la aplicación y mantiene un estándar de seguridad.

Esta alternativa ha mostrado otras ventajas, que se suman a la antes descripta, permite utilizar en el desarrollo de aplicaciones diversas tecnologías en el núcleo de la aplicación, siendo el integrador el servidor de proximidad. Los módulos de la aplicación propietaria pueden especializarse sin la necesidad de pensar en el modelo de interfaz.

4.1.6.1 Especificación del Servidor de Proximidad

En esta sub sección se detallan las características de tiene implementada el servidor de proximidad y como cada una de ellas soporta y simplifica un aspecto del desarrollo de software.

- **Mensajería**: El servidor de proximidad debe mantener la comunicación en ambos sentidos hacia la interfaz de usuario y hacia el núcleo de la aplicación.

- **Seguridad**: El servidor de proximidad permite implementar una arquitectura más robusta de seguridad pues puede implementar varios modelos de seguridad según sea la interfaz de usuario que se conecte y el servicio al cual se deba conectar.
- **Ubicuidad**: El servidor de proximidad permite crear aplicaciones que se alinean con los principios de ubicuidad, es decir, que la aplicación responda en función del lugar de acceso y el tipo de dispositivo que la invoca.
- **Orquestación**: El servidor de proximidad da la opción de ordenar la secuencia y el modo de en que una aplicación será presentada al usuario.
- **Modularidad**: El servidor de proximidad permite tener un mayor grado de abstracción en la aplicación generando la posibilidad de crear módulos independientes relacionados a la funcionalidad específica de la aplicación.

De lo antes mencionado de desprenden tres ventajas adicionales, rapidez en la creación de aplicaciones, pues los principios de ubicuidad y orquestación son firmemente utilizados; simplicidad, en función de lo antes expuesto en la creación de una aplicación el arquitecto debe centrarse en funcionalidades atómicas y el servidor permitirá orquestar las mismas; por último, todas estas características permiten una integración simple y rápida de las aplicaciones.

Se puede definir al servidor de proximidad como un enrutador de funcionalidades de una aplicación, dentro de las cuales, las funcionalidades de usabilidad son de primordial importancia por el peso que tienen las mismas al momento que un usuario utilice una aplicación.

Otro aspecto a ser considerado es entender cuando debe ser utilizado el servidor de proximidad en una aplicación, o cuando simplemente debe ser usado el proceso de integración de la funcionalidad de UNDO/REDO. Para responder esta pregunta se ha formalizado un conjunto de métricas el cual permite obtener un simple estudio de viabilidad para el uso del servidor de proximidad, en la tabla 4.2 se puede observar el estudio de viabilidad para la inclusión.

ID	Pregunta asociada	Peso	Umbral
1	Existen restricciones de seguridad	10	Alto
2	Deben ser integrados varios servicios	9	Alto
3	Existe la posibilidad de cambio en los proveedores de servicios	7	Medio
4	Se accederá a la aplicación desde múltiples dispositivos	6	Medio
5	Existen cambios frecuentes en la aplicación	7	Medio
6	Se desea contar con mejoras pos implementación	4	Bajo

Tabla 4.2. Evaluación de Viabilidad.

Para las preguntas asociadas se definen los siguientes atributos.

- **ID**: Es el identificador de la pregunta.
- **Pregunta**: Describe la condición a evaluar.
- **Peso**: Describe la importancia relativa a cada característica.
- **Umbral**: Indica que valor de la característica debe superar.

A continuación se especifica la conversión del lenguaje coloquial al esquema de valores difusos de pesos de la tabla. En una escala del 0 al 10, está dada por los valores de los pesos, se detallan los siguientes intervalos difusos (Tabla 4.3).

Para cada valor extraído de la característica se lo convierte al rango difuso y seguido a esto al valor de umbral asociado; con esto se calcula el promedio de los valores de umbral si este es medio o alto, es decir un valor superior a 5, es aconsejable la utilización de un servidor de proximidad para la aplicación en cuestión.

Expresión	Intervalo	Valor
No	0	Bajo
Nada	0; 3	Bajo
Poco	3; 5	Bajo
Regular	5; 7	Medio
Algunos	5; 7	Medio
Mucho	7; 9	Medio
Todo	8; 10	Alto
Existe	8; 10	Alto
Si	10	Alto

Tabla 4.3. Intervalos Difusos

De lo antes mencionado se puede conformar un servidor de proximidad con los siguientes módulos que permitirán la interacción entre usuarios y aplicación de forma simple y rápida.

- **Módulo de Seguridad:** actúa como gestor de accesos entre el usuario y los distintos servicios que utiliza la aplicación, este módulo almacena la relación entre usuario de aplicación y permisos que este tiene de acceso a los servicios internos o externos que accede la aplicación.

- **Módulo de Orquestación:** este actúa como controlador en una arquitectura de Modelo Vista Controlador, dando un mayor grado de abstracción y en consecuencia poder manejar más complejidad, es por eso que implementa un motor de flujos de tarea que permite alinearse a la teoría de modelado de procesos de negocios.

- **Módulo de Acceso a Servicios Externos:** es el nexo entre el servidor de proximidad y el servicio externo o interno, en el se encapsula toda la complejidad de la comunicación con los servicios, ya que al momento de escribir este documento no existe una normalización en la mensajería, por último este módulo posee la característica de ser ejecutado en un proceso independiente al servidor de proximidad, esto es hecho así, pues en caso de generarse una falla en la comunicación con los servicios y este quedar en un estado indefinido, solo es necesarios reiniciar el proceso en cuestión y no el servidor de proximidad en su totalidad.

- **Módulo de Trazabilidad:** es el encargado de llevar un registro de todas las operaciones que realiza un usuario, haciendo especial hincapié en los tiempos y caminos de acceso. Esta información será de valor para otros módulos del servidor.

- **Módulo de Administración:** es el que configura el servidor para que pueda dar acceso a los usuarios y acceder a los servicios, es donde se realiza la orquestación de la aplicación también, en resumen este módulo interactúa en una u otra medida con todos los demás módulos.

- **Módulo de Adaptabilidad:** se compone por algoritmos provenientes del dominio de la inteligencia artificial, en primer lugar utiliza el modelo SOM (Self Organization Map), también conocido como modelo de Kohonen, el cual categoriza a los usuarios por tipos de acceso, tiempos y flujos de trabajo; esta información es obtenida del módulo de trazabilidad. Con estas categorías generadas un sistema de aprendizaje automático se encarga de insertar

en forma automática nuevos valores al sistema de producción de reglas que permitirá configurar según el usuario la estructura de aplicación más apropiada según sus usos, esto último el sistema de producción de reglas lo realiza interactuando con el orquestador del servidor de proximidad.

4.2. DEFINICIONES

En esta sección se introducen las definiciones de Artefacto de Usabilidad UNDO/REDO (sección 4.2.1) dando una Descripción Formal (sección 4.2.2), se introduce el concepto de Unidad Lógica de Cambio (Sección 4.2.3) y los Puntos de No Retorno (Sección 4.2.4).

4.2.1. Definición de Artefacto de Usabilidad UNDO/REDO

En el contexto del presente trabajo es necesario definir el artefacto de usabilidad UNDO/REDO, este es un artefacto que permite revertir un trabajo hecho y de ser necesario una vez que se ha revertido, poder volver a un estado descartado en el proceso de reversión. En un aspecto teórico todo sistemas puede ser revertido a un estado anterior o ser devuelto a un estado posterior, pues un sistema puede ser representado como un conjunto discreto de estados finitos; en la práctica esto en ciertos casos es inviable por el esfuerzo que demandaría de tiempo y recursos. Se puede dar como resumen a lo antes mencionado, que no todo sistema puede utilizar la funcionalidad de UNDO/REDO, como derivación de esto un sistema puede incluir la funcionalidad de UNDO/REDO en cierta parte del mismo y en otra no ser viable.

Para poder trabajar con el artefacto de UNDO/REDO es necesario definir la información que debe ser almacenada para su recupero, esto se detalla en el apartado Unidades Lógicas de Cambio, por otra parte es necesario definir hechos por los cuales no puede ser vuelto hacia atrás un sistema, esto se define en los Puntos de No Retorno, la definición de estos puntos puede dividirse en dos grandes grupos por un lado cuestiones prácticas y por otro lado cuestiones de recursos compartidos entre usuarios de sistemas.

Con la definición de estos aspectos de la funcionalidad de UNDO/REDO el lector podrá entender en forma estricta los conceptos vertidos en la solución de la presente tesis.

4.2.2. Descripción Formal de UNDO/REDO

Las definiciones y proposiciones de esta sección se utilizan para probar que el patrón UNDO/REDO tomado como transformación algebraica construida bajo ciertas restricciones, es la transformación inversa de la transformación DO.

Definición 1. Sea $E = \{\varepsilon_j^i / \varepsilon_j$ es una estructura de datos$\}$ es el conjunto de todas las estructuras de datos.

Definición 2. Sea ε_j^i es la instancia *i* de la estructura de datos ε_j perteneciente a *E.*

Definición 3. Sea $\varepsilon_j^C = \{ \varepsilon_j^i / \varepsilon_j^i$ es la instancia *i* de la estructura $\varepsilon_j\}$ es el conjunto de todos los posibles casos de estructura de datos. ε_j.

Definición 4. Sea $o_\tau^{\varepsilon j}$ es la transformación donde se verifica $o_\tau^{\varepsilon j}$: $\varepsilon_j^C \rightarrow \varepsilon_j^C$ y $o_\tau^{\varepsilon j}(\varepsilon_j^i) = \varepsilon_j^{i+1}$.

Definición 5. Sea ε_j^{Cr} el límite de ε_j^C definida como $\varepsilon_j^{Cr} = \{ \varepsilon_j^i / \varepsilon_j^i$ es la instancia *i* de la estructura ε_j que verifica $o_\tau^{\varepsilon j} (\varepsilon_j^{i-1}) = \varepsilon_j^i\}$

Proposición 1. Si $o_\tau^{\varepsilon j}$: $\varepsilon_j^C \rightarrow \varepsilon_j^{Cr}$ entonces $o_\tau^{\varepsilon j}$ es bijectiva.

Prueba: $o_\tau^{\varepsilon j}$ es injectiva por definición 4, $o_\tau^{\varepsilon j}$ es subjectiva por definición 5, entonces $o_\tau^{\varepsilon j}$ es bijectiva por ser injectiva y subjectiva. QED.

Proposición 2. Si $o_\tau^{\varepsilon j}$: $\varepsilon_j^C \rightarrow \varepsilon_j^{Cr}$ entonces es inversa.
Prueba: Sea $o_\tau^{\varepsilon j}$ es bijectiva por proposición 1, entonces por propiedad del algebra $o_\tau^{\varepsilon j}$ tiene inversa. QED.

Definición 6. Sea o_τ es el conjunto de las transformaciones $o_\tau^{\varepsilon j}$.

Definición 7. Sea Φ la operación de composición definida por propiedades del algebra transformacional.

Definición 8. Sea Σ esta definida por las estructuras $< E^\Sigma , o_\tau^\Sigma, \Phi >$ donde $E^\Sigma \subseteq E$ y $o_\tau^\Sigma \subseteq o_\tau$.

Definición 9. Sea $X = o_\tau^{\varepsilon j1} \Phi o_\tau^{\varepsilon j2} \Phi ... \Phi o_\tau^{\varepsilon jn}$ es la composición de la transformación donde se verifica $o_\tau^{\varepsilon ji}$: $\varepsilon_j^C \rightarrow \varepsilon_j^{Cr}$ para todo i: 1...n. Por construcción del algebra X: $\varepsilon_j^C \rightarrow \varepsilon_j^{Cr}$.

Proposición 3. La composición de la transformación X es inversa y bijectiva.

Prueba: Sea X = $o_\tau^{\varepsilon j1}$ Φ $o_\tau^{\varepsilon j2}$ Φ ... Φ $o_\tau^{\varepsilon jn}$. Para toda i: 1...n verifica $o_\tau^{\varepsilon ji}$ que es inversa por composición 2. Sea $[o_\tau^{\varepsilon ji}]^{-1}$ es la transformación inversa de $o_\tau^{\varepsilon ji}$, por propiedades del algebra $[o_\tau^{\varepsilon ji}]^{-1}$ es bijectiva. Entonces es posible la composición de transformación $X^{-1} = [o_\tau^{\varepsilon jn}]^{-1}$ Φ$[o_\tau^{\varepsilon jn-1}]^{-1}$ Φ ... Φ$[o_\tau^{\varepsilon j1}]^{-1}$. La transformación X^{-1} es bijectiva por transformación de bijectividad. Entonces la transformación X^{-1}: $\varepsilon_j^{Cr} \rightarrow \varepsilon_j^{C}$ *existe y es inversa en* X. QED.

Definición 10. Sea UNDO/REDO la X^{-1} transformación de X.

4.2.3. Unidad Lógica de Cambio

Para ahondar en el entendimiento del concepto de las Unidades Lógicas de Cambio (ULC) se da un ejemplo: el campo teléfono, el mismo está compuesto por código de país, código de área y número telefónico y en algunos casos extensión. Estos datos pueden ser ingresados en campos separados, pero a efectos del proceso de UNDO/REDO, se debería trabajar como una unidad que de ser necesario regresar a un valor anterior el servicio retorne la información de todos los campos a un momento determinado. Esto es porque los campos antes descriptos, a nivel de interfaz de usuario están fuertemente relacionados, código de país, código de área, número telefónico y extensión son percibidos por el usuario de sistemas como un único campo, denominado teléfono.

Sobre lo antes expuesto, se pueden realizar las siguientes inferencias, asociar como una unidad indivisible un conjunto de campos está relacionado con el dominio de acción del sistema; es decir, que en dos sistemas con conjuntos de datos similares relacionados a distintos dominios, las ULC definida para cada uno puede ser distinta.

Definir la granularidad con la que se debe trabajar en un sistema es algo conocido en la ingeniería de software, el nivel de desagregación y la relación con que se deben almacenar los datos es un concepto común en el mundo de las bases de datos; este mismo concepto no lo es en lo referido a la funcionalidad de UNDO/REDO dentro de la usabilidad de software.

El concepto de ULC se puede relacionar con el rotulado de una versión dentro de un sistema de versionado de software, pero a nivel de datos específicamente; cuando se define una ULC el

servicio lo trata como una unidad indivisible la cual debe ser almacenada completa y restaurada de la misma forma.

4.2.4. Definición de Puntos de No Retorno

Los puntos de no retorno (PNR) son estados del sistema que por motivos propias al domino o a limitaciones prácticas del sistema no se puede retroceder más allá de este, estos suelen ser comunes en ambientes multiusuario donde la modificación de un usuario hace que la pila de modificaciones realizada por otro usuario queda invalidad.

El concepto de PNR ha sido tratado con anterioridad y no representa un aporte novel del presente trabajo, se puede hacer una analogía con la terminología relacionada al dominio de base de datos y decir que las ULC representan los registros que serán almacenados en una tabla y los PNR son el proceso de “Transaction & Rollback”; proceso por el cual, se vuelve hasta el inicio de la transacción, descartándose todas las transacciones hechas, de esta manera los PNR nos dan la profundidad que podrá tener una pila de cambios realizados por el sistema.

Situaciones donde se producen PNR pueden presentarse cuando el usuario salga de la interfaz en la cual estaba trabajando, esto trae aparejado que los datos que se estaban apilando se limpien y se produzca un PNR, pues no hay nada que volver hacia atrás si el usuario retorna a la interfaz de usuario en la que estaba trabajando, en este caso se podría decir que el PNR es del tipo temporal.

Otro tipo de PNR son los físicos, si un usuario estaría trabajando en forma continua sobre una misma interfaz y generando cambios, aunque poco probable, esta llegaría a un punto en el cual no se podría volver hacia un estado anterior pues supero la capacidad física de almacenamiento temporal del servicio.

Por último están los PNR del tipo lógico los cuales que por razones de dominio no se puede volver hacia atrás, como ejemplo se puede citar en ambientes colaborativos si dos usuarios están utilizando una misma interfaz de usuarios y esta permite a través del servicio el compartir pilas de cambios, en el momento que uno de los usuarios de por aceptado los cambios; esto implicará, que el otro no podrá seguir realizando por el momento cambios en la misma interfaz pues el proceso de compartir cambios se había visto truncado, este segundo usuario debería reiniciar el proceso de cambios en el servicio para poder continuar con su trabajo.

En resumen las ULC y los PNR son dos características necesarias para la creación de un servicio que implemente la funcionalidad de UNDO/REDO y estas deberán ser utilizadas como referencia para la construcción del mismo.

4.3. PROCESO PROPUESTO

El proceso de Inclusión de Servicios en Aplicaciones Anfitrionas Basado en Patrones de Usabilidad que se propone en esta tesis está estructurado en cuatro fases: Modelado del Servicio, Pruebas de la Aplicación, Creación del Servicio, y Pruebas del Servicio. Las tareas propuestas para cada fase se muestran en la Tabla 4.4.

El objetivo de la fase de Modelado del Servicio es identificar cuáles son las características de usabilidad que posee la aplicación anfitriona para poder iniciar el proceso de inclusión del servicio dentro de la misma. A los efectos de poder alcanzar el objetivo se realizan las siguientes tareas: Análisis de Usabilidad, Detección de Unidades Lógicas de Cambio, Detección de Puntos de No Retorno (PNR) y Análisis de uso de Servidor de Proximidad (SP).

Fase	Tareas
Modelado del Servicio (F1)	Análisis de Usabilidad (F1-T1) Detección de Unidades Lógicas de Cambio (F1-T2) Detección de Puntos de No Retorno (F1-T3) Análisis de uso de Servidor de Proximidad (F1-T4)
Pruebas de la Aplicación (F2)	Prueba de Usabilidad de la Aplicación (F2-T5) Prueba de Carga de la Aplicación (F2-T6)
Creación del Servicio (F3)	Especificación de la Configuración de Servicio (F3-T7) Implementación de la Invocación del Servicio (F3-T8) Implementación de la Inclusión del Servicio (F3-T9)
Pruebas del Servicio (F4)	Prueba de Usabilidad del Servicio (F4-T10) Prueba de Carga del Servicio (F4-T11) Evaluación (F4-T12)

Tabla 4.4. Fases y Tareas del Proceso de Inclusión de Servicios en Aplicaciones Anfitrionas Basado en Patrones de Usabilidad

La Tarea Análisis de Usabilidad intenta detectar las características propias que posee la aplicación anfitriona en relación a la usabilidad de software; cabe mencionar, que este análisis no intenta dar una evaluación de la usabilidad de un sistema sino que intenta que la apreciación que se ha tenido

de la usabilidad dada por el método de evaluación sea la misma una vez finalizada la inclusión de la funcionalidad de UNDO/REDO.

La Tarea Detección de ULC busca encontrar en las interfaces de usuarios los conjuntos de datos que deberían ser considerados como conjuntos indivisibles de información, esta tarea debe ser realizada en conjunto con el desarrollador de la aplicación, analizando la documentación del sistema y observando la interfaz de usuario de la interacción de estos tres deberá surgir el conjunto de ULC para cada interfaz de usuario.

La Tarea Detección de PNR busca encontrar donde se encuentran los mismos en la aplicación, el modelo de búsqueda es similar a proceso realizado en la tarea anterior y pueden ser ejecutadas en paralelo, la idea es detectar los sucesos que desde la interfaz de usuario impiden seguir volviendo hacia a tras una aplicación, aquí es necesario enfatizar que el análisis que se debe hacer esta relacionado al dominio de la usabilidad de software; es por eso, que se nombra específicamente a la interfaz de usuario y no deben incluirse por tratarse de dominios por fuera de la usabilidad a situaciones relacionadas, como por ejemplo con la base de datos.

En la fase Análisis de uso de Servidor de Proximidad se intenta determinar si es necesario incorporar al proceso de integración del servicio el servidor de proximidad, para ello se siguen los lineamientos para el estudio de viabilidad del servidor de proximidad.

El objetivo de la fase de Pruebas de la Aplicación es tomar una foto del estado de situación de la aplicación anfitriona antes de ser incluida la funcionalidad de UNDO/REDO; esto se hace, pues la inclusión de el servicio no debería traer cambios sustanciales el proceso de aprendizaje de utilización del sistema ni en el tiempo de respuesta con el que responde ante el usuario; para alcanzarlo, se realizan las siguientes tareas: Prueba de Usabilidad de la Aplicación y Prueba de Carga de la Aplicación.

La Tarea Prueba de Usabilidad de la Aplicación la evaluación por parte de un usuario nuevo del sistema, el cual se enfrenta al sistema por primera vez poder tener una evaluación que tiempo promedio que le llevaría reconocer las funcionalidades del mismo y en definitiva reconocer la interfaz de usuario del mismo, la cual permitirá obtener una apreciación sobre esta métrica.

La Tarea Prueba de Carga de la Aplicación busca reconocer cual es el tiempo de respuesta ante una carga dada de usuarios, esta prueba no busca evaluar si el sistema esta correctamente dimensionado para la cantidad de usuarios con los cuales trabajara, no es el objetivo de la misma, sino tener una

medida de tiempo de respuesta para ser comparada con la misma prueba luego de agregar la invocación al servicio.

El objetivo de la fase de Creación del Servicio es el momento donde se empieza a trabajar físicamente con el proceso que llevará como resultado final la inclusión del mismo en la aplicación anfitriona, para alcanzarlo se realizan las siguientes tareas: Especificación de la Configuración de Servicio, Implementación de la Invocación del Servicio e Implementación de la Inclusión del Servicio.

La Tarea Especificación de la Configuración de Servicio busca a través de un conjunto de interfaces de usuario en el servicio, las cuales permiten dar formato y configurar según los requerimientos de la aplicación anfitriona el servicio.

La Tarea Implementación de la Invocación del Servicio busca una vez definido las características que serán soportadas por el servicio, lo cual fue hecho en la etapa anterior, definir cuál es la técnica más apropiada para invocar al servicio desde la aplicación anfitriona, esta surgirá de evaluar con que tecnología ha sido construida la aplicación anfitriona y las características provistas para el servicio.

La Tarea de Implementación de la Inclusión del Servicio busca ahondar y supervisar el método de inclusión de las llamadas o invocaciones al servicio desde la aplicación anfitriona, este toma como salida el resultado de la etapa anterior e implemente este en el sistema.

El objetivo de la fase de Pruebas del Servicio es volver a realizar el conjunto de pruebas hechas en la Fase 2 para poder comparar el rendimiento antes y después de incluir el servicio en cuestión. Para alcanzarlo se realizan las siguientes tareas: Prueba de Usabilidad del Servicio, Prueba de Carga del Servicio, y Evaluación.

La Tarea Prueba de Usabilidad del Servicio busca evaluar la usabilidad del mismo una vez incluido el servicio, el detalle del método se ha descripto en la Fase 2.

La Tarea Prueba de Carga del Servicio busca evaluar cual es el tiempo de respuesta del sistema una vez incluida la funcionalidad de UNDO/REDO.

La Tarea Evaluación busca como finalización del proceso comparar las métricas de usabilidad antes y después de la inclusión del sistema, y el tiempo de respuesta del mismo antes y después de la inclusión del servicio. Es de esperar que las evaluaciones antes y después de la inclusión tiendan a

ser muy similares lo que daría como conclusión que la inclusión de la funcionalidad de UNDO/REDO al ser insertada en la aplicación no ha generado alteraciones en la misma.

El flujo del proceso de Inclusión de Servicios en Aplicaciones Anfitrionas Basado en Patrones de Usabilidad se muestra en la Figura 4.1. Las fases del proceso propuesto con descripción de tareas, entradas y salidas se presentan en la Tabla 4.5.

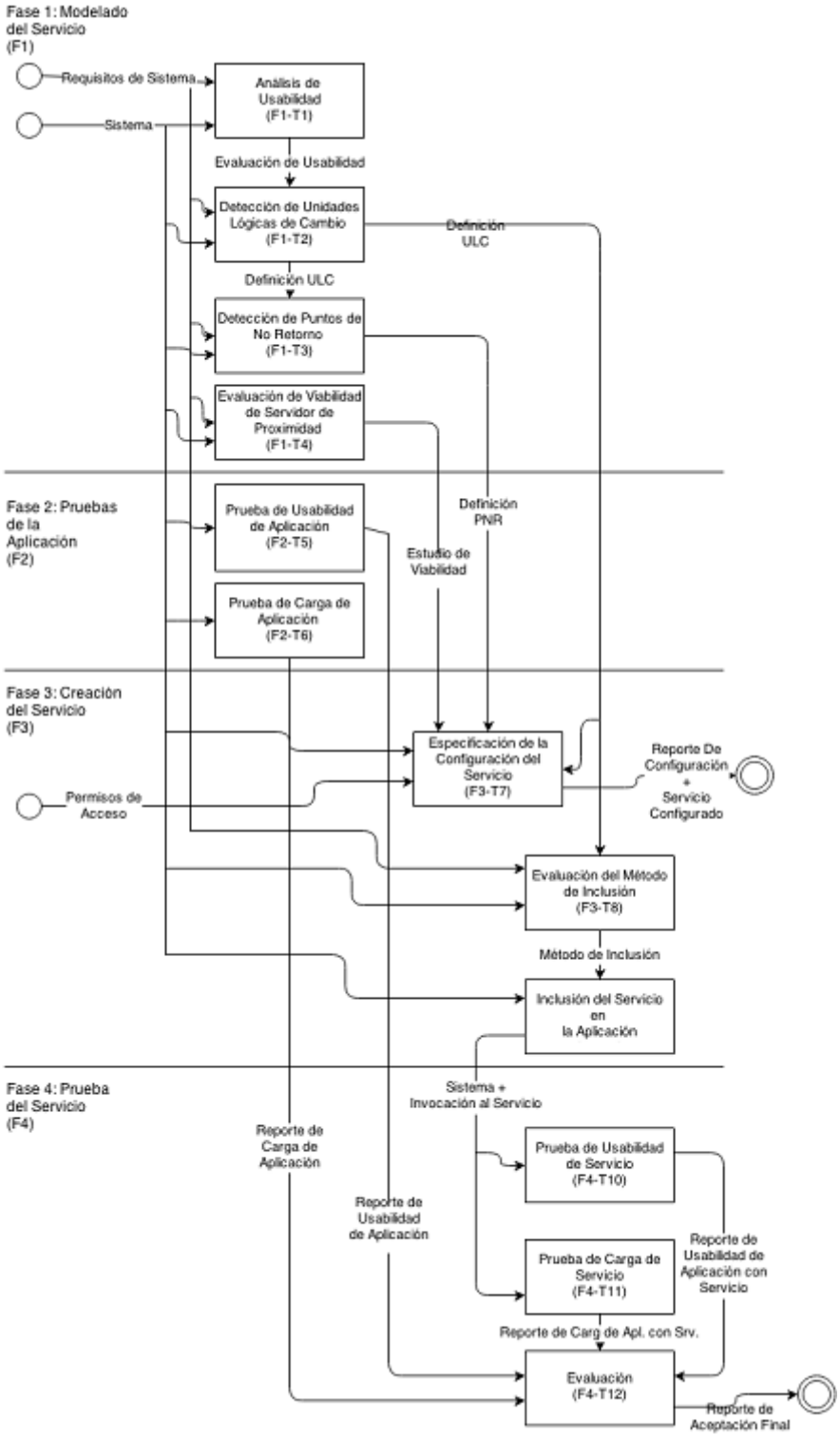


Figura 4.1. Flujo de proceso.

FASE	TAREA	ENTRADA		TÉCNICA	SALIDA	
		DENOMINACIÓN	REPRESENTACIÓN		DENOMINACIÓN	REPRESENTACIÓN
MODELADO DEL SERVICIO (FI)	Análisis de Usabilidad (F1-T1)	Requerimientos de Sistema + Sistema	Documento + Sistema	Detección y Evaluación de Requisitos de Usabilidad. (F1-T1-T)	Evaluación de Usabilidad. (F1-T1-R)	Documento.
		A partir de la documentación del sistema se intenta detectar los requisitos. En caso de existir el sistema se trabaja también sobre la interfaz.			Se obtienen los requisitos de usabilidad definidos para el sistema	
	Detección de Unidades Lógicas de Cambio (ULC) (F1-T2)	Requisitos de Sistema + Sistema + Evaluación de Usabilidad (F1-T1-R)	Documento + Sistema	Detección de ULC. (F1-T2-T)	Definición de ULC. (F1-T2-R)	Documento.
		Del análisis de esta documentación y del sistema de existir se obtendrán las ULC			Conjunto de información que se tratara como una unidad	
	Detección de Puntos de No Retorno (PNR) (F1-T3)	Requisitos de Sistema + Sistema + Diseño de ULC (F1-T2-R)	Documento	Detección de PNR. (F1-T3-T)	Definición de PNR. (F1-T3-R)	Documento.
		El conjunto de los Requisitos y las ULC permiten detectar los PNR.			Hitos en la utilización del sistema que no permiten volver a tras al mismo.	
	Evaluación de Viabilidad de Servidor de Proximidad (F1-T4)	Requerimientos de Sistema + Sistema	Documento + Sistema	Estudio de Viabilidad. (F1-T4-T)	Estudio de Viabilidad (F1-T4-R)	Documento.
		A partir de la documentación del sistema se intenta detectar la viabilidad de uso del servidor de proximidad. En caso de existir el sistema se trabaja también con este.			Se obtiene el estudio de viabilidad para el uso del servidor de proximidad.	
PRUEBAS DE LA APLICACIÓN (F2)	Prueba de Usabilidad de Aplicación (F2-T5)	Sistema	Software	Prueba de usabilidad de Aplicación. (F2-T5-T)	Reporte de Usabilidad de Aplicación (F5-T5-R)	Documento.
		Se le aplica al sistema original la validación de usabilidad.			Se obtiene un reporte con el análisis de usabilidad del mismo.	
	Prueba de Carga de Aplicación (F2-T6)	Sistema	Software	Prueba de Carga de Aplicación. (F2-T6-T)	Reporte de Carga de Aplicación (F2-T6-R)	Documento.
		Se le aplica al sistema original la validación de carga.			Se obtiene un reporte con el análisis de carga del mismo.	
CREACIÓN DEL SERVICIO (F3)	Especificación de la Configuración de Servicio (F3-T7)	Diseño de ULC + Definición de PNR + Permisos de Acceso + Sistema + Estudio de Viabilidad (F1-T2-R) (F1-T3-R)	Software + Documento	Configuración de Servicio. (F3-T7-T)	Reporte de Configurado + Servicio Configurado. (F3-T7-R)	Documento + Software.
		Ser recolecta la información necesaria para la configuración del sistema.			Se obtiene el servicio configurado para el sistema anfitrión	
	Evaluación del método de inclusión (F3-T8)	Definición de ULC + Documentación de Sistemas + Sistema (F1-T2-R)	Documento + Software	Evaluación de Métodos de Inclusión. (F3-T8-T)	Método de Inclusión. (F3-T8-R)	Documento.
		A partir de las ULC detectadas se analiza el mejor método para incluir el servicio en el sistema.			Definición del mejor método para inyectar el servicio en el sistema.	
	Inclusión del Servicio en la Aplicación (F3-T9)	Sistema + Método de Inclusión (F3-T8-R)	Software + Documento	Programación o rotulado de archivos fuentes (F3-T9-T)	Sistema + Invocación al Servicio. (F3-T9-R)	Software.
		Se procede a incluir en el sistema las invocaciones al servicio.			Invocaciones al servicio desde el sistema anfitrión.	
PRUEBAS DEL SERVICIO (F4)	Prueba de Usabilidad del Servicio (F4-T10)	Sistema + Invocación al Servicio (F3-T9-R)	Software.	Prueba de usabilidad de sistemas. (F4-T10-T)	Reporte de Usabilidad de Aplicación con Servicio (F4-T10-R)	Documento.
		Se lleva a cabo una nueva prueba de usabilidad.			El resultado de esta no debería variar del anterior sumado el servicio.	
	Prueba de Carga del Servicio (F4-T11)	Sistema + Invocación a Servicio (F3-T9-R)	Software.	Prueba de Carga. (F4-T11-T)	Reporte de Carga de Aplicación con Servicio (F4-T11-R)	Documento
		Se lleva a cabo una nueva prueba de carga.			El resultado de esta no debería variar del anterior sumado el servicio.	
	Evaluación (F4-T12)	Reporte de Usabilidad + Reporte de Carga (F2-T5-R) (F4-T10-R) (F2-T6-R) (F4-T11-R)	Documento.	Evaluación de Resultados. (F4-T12-T)	Reporte de Aceptación Final (F4-T12-R)	Documento.
		Evaluación final de los documentos generados en las pruebas.			Aceptación final del proceso.	

Tabla 4.5. Fases del proceso propuesto.

4.4. TÉCNICAS PROPUESTAS ASOCIADAS A LAS TAREAS DEL PROCESO

Para desarrollar las actividades correspondientes a cada fase del proceso propuesto, se crearon las siguientes técnicas: Técnica de Detección de Requerimientos de Usabilidad (F1-T1-T), Técnica de Detección de ULC (F1-T2-T), Técnica de Detección de PNR (F1-T3-T), Técnica de Estudio de Viabilidad (F1-T4-T), Técnica de Prueba de Usabilidad de Aplicación (F2-T5-T), Técnica de Prueba de Carga de Aplicación (F2-T6-T), Técnica de Configuración de Servicio (F3-T7-T), Técnica de Evaluación de Métodos de Inclusión (F3-T8-T), Técnica de Programación o rotulado de Archivos Fuentes (F3-T9-T), Técnica de Prueba de Usabilidad de Sistemas (F4-T10-T), Técnica de Ejecución de la Prueba de Carga (F4-T11-T), y Técnica de Evaluación de Resultados (F4-T12-T).

4.4.1. Técnicas Utilizadas para las Actividades de la Fase de Modelado del Servicio

Este grupo de técnicas comprende: Técnica de Detección de Requerimientos de Usabilidad (sección 4.3.1.1), Técnica de Detección de ULC (sección 4.3.1.2), Técnica de Detección de PNR (sección 4.3.1.3) y Técnica de Análisis de uso de Servidor de Proximidad (sección 4.3.1.4).

4.4.1.1. Detección y Evaluación de Requisitos de Usabilidad (F1-T1-T)

La norma ISO define la usabilidad como la capacidad que tiene un producto para ser usado por determinados usuarios con el fin de alcanzar unos objetivos concretos con efectividad, eficiencia y satisfacción dentro de un contexto de uso específico [ISO 9241-210:2010]. Con esto se puede tener una idea clara del objetivo de esta actividad es la evaluación de cómo el sistema interactúa con el usuario del mismo. La razón primordial de incluir esta actividad es la de poder evaluar a posteriori de la inclusión del servicio si la funcionalidad de UNDO/REDO, si se generó alguna diferencia en la apreciación de la usabilidad del sistema, esta evaluación debería ser neutra es decir que no se han producido modificaciones antes y después de la inclusión de la funcionalidad o en el mejor de los casos advertir una mejora en la interacción usuario-sistema.

Como resultado anexo a lo antes expresado se contara con una evaluación general de usabilidad para el sistema la cual podrá ser usada para hacer ajustes no solamente con la funcionalidad en cuestión sino con otros aspectos de la usabilidad de software. Los pasos para desarrollarla se presentan en la Tabla 4.6.

Detección y Evaluación de Requisitos de Usabilidad
Entradas: Requerimientos de Sistema + Sistema (en caso de estar en producción) Salidas: Evaluación de Usabilidad
Paso 1. Definir documento de evaluación para cada interfaz. 1.1. Nombre de la interfaz a evaluar. 1.2. Descripción general de la interfaz. Paso 2. Preparación del material y selección del equipo evaluador. 2.1. Selección los usuarios y/o expertos que evaluarán el sistema. 2.2. Disponer de los equipos necesarios para realizar la prueba. Paso 3. Desarrollo del método de Inspección Heurística de Molich y Nielsen [Granollers et al, 2005]. El método de evaluación heurística es un método informal en el cual se le pide a un conjunto de evaluadores que den su opinión sobre la facilidad de uso de una aplicación siguiendo un conjunto de lineamientos específicos. Para cada interfaz se debe evaluar: 3.1. Los cuadros de diálogos son simples y naturales. 3.2. La aplicación habla el lenguaje del usuario. 3.3. Se minimiza la carga de memoria en el usuario. 3.4. Hay consistencia en los datos pedido por la interfaz. 3.5. La interfaz proporciona información de proceso. 3.6. La interfaz proporciona métodos de salida. 3.7. La interfaz proporciona caminos abreviados. 3.8. La interfaz proporciona mensajes de error. 3.9. La aplicación previene errores. Paso 4. Unificar opiniones de los evaluadores. 4.1. Se procederá a generar un documento unificado con el resumen de los distintos evaluadores, el cual será el documento deliberable de la etapa. Para cada punto de la etapa 3 (3.1 a 3.9) se le dará un rango de 1 a 5 donde 1 es la más baja calificación y 5 la más alta.

Tabla 4.6. Detección y Evaluación de Requisitos de Usabilidad (F1-T1-T)

4.4.1.1.1 Herramienta de documentación para detección y evaluación de requisitos de usabilidad

Este apartado presenta un breve ejemplo del tipo de documentación generada en esta etapa. La documentación de entrada de la etapa está dada por la documentación del sistema, en este caso se debe hacer referencia a los apartados referidos a la interfaz de usuario; haciendo un esfuerzo para dejar de lado una argumentación puramente teórica, el autor de este trabajo se referirá a aspectos más prácticos, los cuales no siempre se encuentran alineados a las buenas prácticas de la ingeniería del software, pero ayudaran a concluir la tarea de generación de la información requerida en la etapa. La pregunta a responder en esto es, de no existir la documentación apropiada como se puede recolectar esta para formalizar la etapa. Si el sistema se encuentra en producción o en una etapa

avanzada de prueba, la solución simple es la de colocarse frente al sistema y recolectar la información necesaria a través de las interfaces del propio sistema. En caso de no encontrarse el sistema funcionando se debe contactar a los diseñadores y desarrolladores para que provean las interfaces que se tienen al momento y de ser necesario finalizarlas para que puedan ser evaluadas al menos en la modalidad de maqueta (mockup por el termino en ingles).

En la tabla 4.6.1 se detalla la secuencia de pasos a seguir según la información con la que se cuenta.

Orden	Recurso a evaluar
1	Documentación del sistema
2	Sistema en producción o evaluación
3	Sistema en desarrollo
4	Construcción de maquetas

Tabla 4.6.1. Orden de precedencia

Esta secuencia de pasos definidos anteriormente no invalida que el responsable de recolectar la información pueda utilizar un conjunto de estas para poder obtener un mejor modelo a evaluar.

A continuación en la figura 4.2 se detalla la información que contendrá cada documento de evaluación para cada interfaz a evaluar. Este documento pude ser impreso o en formato electrónico, este último es más apropiado, pues su versionado es sencillo comparado con la copia papel.

A continuación se presenta el documento resumen para cada interfaz y que engloba la evaluación realizada por todos los evaluadores (figura 4.3.).

En este se vuelcan las opiniones promedio, haciendo truncamiento de los resultados en caso de poseer dígitos decimales, y es firmado por el auditor líder de la etapa, con esto se finaliza la evaluación y se contaría con un documento por interfaz evaluada. Como extensión de este tópico se puede generar un documento resumen de todas las interfaces para obtener una ponderación general de la usabilidad del sistema, esto no es propio de la metodología implementada pero es un desprendimiento lógico luego de realizar el esfuerzo de evaluación de las diversas interfaces.

Documento de Evaluación de Interfaz			
Fecha		Responsable	
Nombre de Sistema			
Nombre de Interfaz		Nombre Clave	
Descripción General			
Puntos a Evaluar		Valor (Rango de 1-5) 1=Malo - 5=Excelente	
1.	Los cuadros de diálogos son simples y naturales.		
2.	La aplicación habla el lenguaje del usuario.		
3.	Se minimiza la carga de memoria en el usuario.		
4.	Hay consistencia en los datos pedido por la interfaz.		
5.	La interfaz proporciona información de proceso.		
6.	La interfaz proporciona métodos de salida.		
7.	La interfaz proporciona caminos abreviados.		
8.	La interfaz proporciona mensajes de error.		
9.	La aplicación previene errores.		
Firma			
Documento Versión			

Figura 4.2. Documento de Evaluación.

Resumen de Evaluación de Interfaz			
Fecha		Auditor Líder	
Nombre de Sistema			
Nombre de Interfaz		Nombre Clave	
Descripción General			
Puntos a Evaluar		Valor (Rango de 1-5) 1=Malo - 5=Excelente	
1.	Los cuadros de diálogos son simples y naturales.		
2.	La aplicación habla el lenguaje del usuario.		
3.	Se minimiza la carga de memoria en el usuario.		
4.	Hay consistencia en los datos pedido por la interfaz.		
5.	La interfaz proporciona información de proceso.		
6.	La interfaz proporciona métodos de salida.		
7.	La interfaz proporciona caminos abreviados.		
8.	La interfaz proporciona mensajes de error.		
9.	La aplicación previene errores.		
Firma			
Documento Versión			

Figura 4.3. Documento de Evaluación Resumen.

4.4.1.1.2 Ejemplo de uso de la herramienta

A continuación se da una versión ya completa del documento antes mencionado (figura 4.4)

Documento de Evaluación de Interfaz			
Fecha	2012-10-05	Responsable	Juan Godoy
Nombre de Sistema	Sistema Contable General		
Nombre de Interfaz	Alta de plan de cuenta	Nombre Clave	I01-AL-PC
Descripción General			
Esta interfaz es utilizada para la carga de los planes de cuenta de los diferentes contadores que utilizan el sistema, esta interfaz la utilizada principalmente en el momento de parametrización de sistemas y es poco usada durante la operación del mismo. Si es de destacar que el uso en este periodo es intensivo y deben ser ingresados los planes de cuenta completo de cada contador.			

Puntos a Evaluar		Valor (Rango de 1-5) 1=Malo - 5=Excelente
1.	Los cuadros de diálogos son simples y naturales.	3
2.	La aplicación habla el lenguaje del usuario.	4
3.	Se minimiza la carga de memoria en el usuario.	2
4.	Hay consistencia en los datos pedido por la interfaz.	3
5.	La interfaz proporciona información de proceso.	1
6.	La interfaz proporciona métodos de salida.	3
7.	La interfaz proporciona caminos abreviados.	2
8.	La interfaz proporciona mensajes de error.	3
9.	La aplicación previene errores.	1

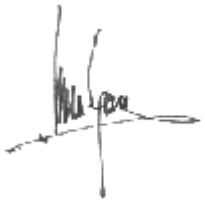
Firma	
Documento Versión	0.0.1

Figura 4.4. Documento de Evaluación Completo.

Esta documentación se debe completar para todas las interfaces de usuario que serán relevadas para la funcionalidad UNDO/REDO.

A continuación se presenta en la figura 4.5., el documento resumen para la interfaz completad en la figura 4.4.

Como se puede observar este documento es muy similar al antes descripto, pero aglutina toda la información de la interfaz, en caso de existir otro evaluador se ponderará también al otro evaluador.

Resumen de Evaluación de Interfaz			
Fecha	2012-10-05	Auditor Líder	Alberto Pérez
Nombre de Sistema	Sistema Contable General		
Nombre de Interfaz	Alta de plan de cuenta	Nombre Clave	I01-AL-PC
Descripción General			
Esta interfaz es utilizada para la carga de los planes de cuenta de los diferentes contadores que utilizan el sistema, esta interfaz la utilizada principalmente en el momento de parametrización de sistemas y es poco usada durante la operación del mismo. Si es de destacar que el uso en este periodo es intensivo y deben ser ingresados los planes de cuenta completo de cada contador.			

Puntos a Evaluar		Valor (Rango de 1-5) 1=Malo - 5=Excelente
1.	Los cuadros de diálogos son simples y naturales.	3
2.	La aplicación habla el lenguaje del usuario.	4
3.	Se minimiza la carga de memoria en el usuario.	2
4.	Hay consistencia en los datos pedido por la interfaz.	3
5.	La interfaz proporciona información de proceso.	1
6.	La interfaz proporciona métodos de salida.	3
7.	La interfaz proporciona caminos abreviados.	2
8.	La interfaz proporciona mensajes de error.	3
9.	La aplicación previene errores.	1

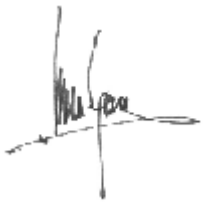
Firma	
Documento Versión	0.0.2

Figura 4.5. Resume de Evaluación de Interfaz Completo.

4.4.1.2. Técnica de Detección de ULC (F1-T2-T)

El proceso de detección de ULC intenta definir los componentes indivisibles de información dentro de una interfaz de usuario; para esta tarea se tomarán los requisitos de sistema sumado a la evaluación de usabilidad y de existir el sistema se lo analizará también. Como deliberable de esto se obtendrá la definición de las ULC, que será parte central en el proceso de configuración del servicio de UNDO/REDO. Los pasos para desarrollarla se presentan en la Tabla 4.7.

Técnica de Detección de ULC
Entradas: Requerimientos de Sistema + Sistema (en caso de existir) + Evaluación de Usabilidad Salidas: Definición de ULC
Paso 1. Reutilizar la información generada en el paso 1 de la etapa 1 Definir documento de evaluación de interfaz. 1.1. Nombre de la interfaz a evaluar 1.2. Descripción general de la interfaz Paso 2. Preparación del material y selección del equipo evaluador 2.1. Selección del arquitecto de sistemas que evaluarán el sistema. 2.2. Disponer de los equipos necesarios para realizar la evaluación. Paso 3. Análisis de interfaces El método que se utilizara es una derivación del proceso de evaluación heurística que ha sido detallado en la etapa 1 paso 3, esto se debe a su facilidad de uso y adaptabilidad a distintos dominios de aplicación. Para cada interfaz se debe evaluar: 3.1. Clasificar los datos según el tipo campo de ingreso. Ejemplo: campo de texto, combo, selección múltiple, etc. 3.2. Evaluar cada campo según la siguiente escala. 3.2.1. El campo es atómico. 3.2.2. Está relacionado con otro campo. 3.2.3. El campo puede ser modificado. Paso 4. Construcción del documento deliberable 4.1. Descartar los campos que no pueden ser modificados. 4.2. Denominar a cada campo relacionado con un identificador de interfaz y de ULC. 4.3. Para el resto de los campos atómicos también darle una identificación de interfaz y ULC.

Tabla 4.7. Detección ULC (F1-T2-T)

4.4.1.2.1 Herramienta de documentación para detección y evaluación de ULC

Aquí se presenta el formato de documento necesario para recolectar la información relacionada a la ULC. En este apartado al igual que en el resto del capítulo se hace hincapié a los aspectos prácticos relacionados con la detección de las ULC, en este punto es menester, pues al ser un concepto innovador no se cuenta con la documentación necesaria que las identifique y debe ser educidas de la documentación general de sistemas y/o del propio sistema. Al igual que en la etapa anterior se debe recolectar información de varias fuentes, el orden de precedencia de búsqueda de información está dado en la tabla 4.7., antes mencionada donde se detalla la secuencia de pasos a seguir según la información con la que se cuenta.

A continuación en el formulario 4.6 se detalla la información que contendrá cada documento de evaluación para cada interfaz a evaluar. Este documento pude ser impreso o en formato electrónico, este último es más apropiado pues su versionado es sencillo comparado con la copia papel.

Documento de Evaluación de ULC			
Fecha		Responsable	
Nombre de Sistema			
Nombre de Interfaz		Nombre Clave	
Nombre de ULC			
Descripción General			
Cantidad de Campos de la interfaz			
Para cada Campo de la Interfaz Evaluar			
1	Tipo de Campo		
2	Identificador de Campo		
3	El campo puede ser modificado		
4	El campo es atómico.		
Agregar tantos campos como sea necesario			
Firma			
Documento Versión			

Figura 4.6. Documento de Evaluación de ULC.

4.4.1.2.2 Ejemplo de uso de la herramienta

La fuente de la información para esta etapa proviene de la documentación del sistema y de la aplicación en sí de existir. A modo de ejemplo se redactará parte de un documento perteneciente a un caso de uso (figura 4.7.), documentación habitual en el desarrollo de sistemas y se presentará una pantalla con un ABM simple para entender el concepto de ULC y como se aplica a la metodología en cuestión (figura 4.8.).

> *"…el usuario carga los datos de los profesores que podrán ser considerados como tutores de tesis de grado para el próximo año…"*

Figura 4.7. Caso de Uso.

A continuación se observa la pantalla que representaría en forma simple lo antes descripto.

Figura 4.8. ABM que representa el Caso de Uso.

En una situación real, uno puede basarse en cualquiera de las dos figuras antes presentadas, en el modelo ideal deberían ser evaluadas ambas para tener un mayor grado de certeza de la definición que se dará sobre la ULC.

Aquí se puede observar que los campos “legajo-nombre-apellido” deberían ser tratados como un conjunto indivisible, esta afirmación se debe a que un número de legado debería corresponder a un solo profesor y la relación nombre-apellido también corresponde a una única persona, en consecuencia ambos datos deberían ser mantenidos como uno solo a nivel lógico para la funcionalidad UNDO/REDO. Es válido mencionar que la explicación antes dada se relaciona a situaciones comunes de sistemas y que en determinados dominios la misma podría ser no válida por las características propias del sistema en cuestión; es por esto que la definición de las ULC está relacionada al dominio de trabajo y a la apreciación que el arquitecto/diseñador que está trabajando con la funcionalidad entiende sobre conjunto de datos.

Aunque esta aparente amplitud en los modos de evaluar las ULC puede hacer pensar al lector en una primera apreciación lo subjetivo del mecanismo, el concepto subyacente que existe en poder definir conjuntos de datos lógicos para la funcionalidad en cuestión hace que el modelo sea sólido y adaptable a diversas situaciones y dominios de aplicación.

Como paso siguiente se presentara la documentación completa para la ULC (figura 4.9.)

4.4.1.3. Técnica de Detección de PNR (F1-T3-T)

Esta técnica se basa en la detección de situaciones en la cual el sistema no podrá volver a un estado anterior, esto puede ser dado por razones de diseño, de concurrencia o practicas, es decir por el tiempo que llevaría volver a una situación anterior al sistema. Los pasos para desarrollarla se presentan en la Tabla 4.8.

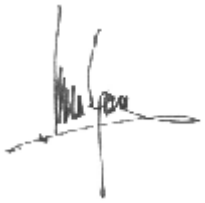
Documento de Evaluación de ULC				
Fecha	2012-10-05		Responsable	Juan Godoy
Nombre de Sistema	Sistema Administración Curricular			
Nombre de Interfaz	Alta de profesores tutores		Nombre Clave	I01-AL-PT
Nombre de ULC	ULC01-AL-PT			
Descripción General				
Esta interfaz es utilizada para la carga de los profesores que serán habilitados para poder dirigir tesis de grado.				
Cantidad de Campos de la interfaz		3		
Para cada Campo de la Interfaz Evaluar				
Campo Legajo				
1.1	Tipo de Campo	Numérico		
1.2	Identificador de Campo	Legajo		
1.3	El campo puede ser modificado	No (generación automática)		
1.4	El campo es atómico.	No		
Campo Nombre				
2.1	Tipo de Campo	Alfanumérico		
2.2	Identificador de Campo	Nombre		
2.3	El campo puede ser modificado	Si		
2.4	El campo es atómico.	No		
Campo Apellido				
2.1	Tipo de Campo	Alfanumérico		
2.2	Identificador de Campo	Apellido		
2.3	El campo puede ser modificado	Si		
2.4	El campo es atómico.	No		
Agregar tantos campos como sea necesario				
Firma				
Documento Versión	0.0.1			

Figura 4.9. Documento de Evaluación de ULC.

Técnica de Detección de PNR
Entradas: Requerimientos de Sistema + Sistema (en caso de existir) + Definición de ULC Salidas: Definición de PNR
Paso 1. Reutilizar la información generada en el paso 1 de la etapa 1 Definir documento de evaluación de interfaz. 1.1. Nombre de la interfaz a evaluar 1.2. Descripción general de la interfaz Paso 2. Preparación del material y selección del equipo evaluador 2.1. Selección del arquitecto de sistemas que evaluarán el sistema. 2.2. Disponer de los equipos necesarios para realizar la evaluación. Paso 3. Análisis de PNR El método que se utilizara es una derivación del proceso de evaluación heurística que ha sido detallado en la etapa 1 paso 3, esto se debe a su facilidad de uso y adaptabilidad a distintos dominios de aplicación. Para cada interfaz se debe evaluar: 3.1. Existe algún requisito de sistema que impida volver a un estado anterior. 3.2. Clasificar la probabilidad de modificaciones de la ULC. 3.3. Clasificar la profundidad de la pila de cada ULC. Paso 4. Construcción del documento deliberable 4.1. Si se detectaron requisitos definir un PNR. 4.2. Si la probabilidad de modificación es alta y la profundidad de pila es alta definir un PNR. 4.3. Para todos los demás casos no definir un PNR.

Tabla 4.8. Detección PNR (F1-T3-T)

4.4.1.3.1 Herramienta de documentación para detección y evaluación de PNR

Aquí se presenta el formato de documento necesario para recolectar la información relacionada a la PNR. En este apartado al igual que en el resto del capítulo se hace hincapié a los aspectos prácticos relacionados con la detección de los PNR. Se debe evaluar documentación general de sistemas y/o al propio sistema y las ULC (figura 4.10.).

Es necesario aclarar algunos ítems de la figura antes referenciada, en particular los referidos a los tipos de PNR, a saber “Condición externa” hace referencia que la ULC referida está relacionada a una operatoria que no es propia del usuario que se encuentra trabajando en este momento con el sistema para esta ULC, esto puede ser dado por el uso que otro usuario le está dando al sistema y luego de determinado uso del mismo hace que los cambio no puedan llevarse a cabo o a una situación propia del sistema, como ser fallas del mismo. En términos generales estos tipos de situaciones se relacionan a ambientes colaborativos de los sistemas.

Documento de Evaluación de PNR					
Fecha				Responsable	
Nombre de Sistema					
Nombre de Interfaz				Nombre Clave	
Nombre de ULC					
Nombre de PNR					
Tipo de PNR					
1	Condición Externa				
2	Profundidad de Cola		Profundidad		
3	Tiempo de Persistencia				
Descripción del a PNR					
Firma					
Documento Versión					

Figura 4.10. Documento de Evaluación de PNR.

Con respecto a la "Profundidad de cola", esta es el PNR por defecto de todas la ULC, para cada ULC se deberá definir si existen PNR y qué tipo son; es de destacar que por cuestiones prácticas para toda ULC existe un tipo de PNR, preséntese esta situación, una aplicación podría almacenar cambios para una ULC en forma infinita, esto es posible en el campo teórico aplicado a la funcionalidad de UNDO/REDO, pero en el aspecto práctico esto no es aceptado, es por esto que se puede concluir que para toda ULC al menos existe un tipo reconocido de PNR.

En consecuencia en este tipo de PNR se deberá definir la cantidad de cambios permitidos como máximo para la ULC tratada, al llegarse al máximo establecido el nuevo valor por almacenar eliminara al valor más antiguo almacenado.

Para finalizar el tipo "Tiempo de persistencia", está relacionado al tiempo que los datos quedarán almacenados en el mecanismo de persistencia temporal del servicio que provee la funcionalidad de UNDO/REDO. En términos generales los datos almacenados temporalmente tienen la duración de la sesión de usuarios que esté utilizando el sistema principal; es decir, cuando este abandona la aplicación principal se deberían marcar los datos almacenados temporalmente como datos a borrar; pero en ciertas situaciones y aquí se vuelve a los ambientes colaborativos, puede ser que al ser

compartida una misma cola de persistencia temporal por varios usuarios los datos puedan ser marcados como aptos para ser borrados luego que el último de los usuarios que comparten esa cola temporal abandone el sistema. Otro caso en el que se puede presentar una situación en donde sea necesario tener una opción temporal, son los sistemas relacionados a traducción de documentos, estos son utilizados por los usuarios traductores, con un amplio grado de cambios que realizan al texto que fue traducido, una vez terminada la primera versión se pasa a realizar un conjunto de nuevas interpretaciones que hacen que sea muy habitual los cambios en los mismos. A su vez estas revisiones pueden ser hechas a lo largo de varios días de trabajo de corrección, este es otro caso donde podría presentarse esta situación, aquí el servicio que presta la funcionalidad de UNDO/REDO podría ser entendido como una especialización particular de un control de versiones de documentos.

4.4.1.3.2 Ejemplo de uso de la herramienta

A continuación se dará un ejemplo en donde se muestra la forma de trabajar los PNR y la relación existente entre estos y la ULC. La fuente de información son la documentación del sistema y los reportes de evaluación de las ULC. La documentación base son los reporte de las ULC, como segunda fuente ante dudas se recurrirá a la documentación del sistema (figura 4.11.).

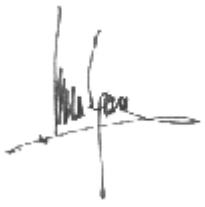
<table>
<tr><td colspan="6">Documento de Evaluación de PNR</td></tr>
<tr><td>Fecha</td><td colspan="3">2012-10-05</td><td>Responsable</td><td>Juan Godoy</td></tr>
<tr><td>Nombre de Sistema</td><td colspan="5">Sistema Administración Curricular</td></tr>
<tr><td>Nombre de Interfaz</td><td colspan="3">Alta de profesores tutores</td><td>Nombre Clave</td><td>I01-AL-PT</td></tr>
<tr><td>Nombre de ULC</td><td colspan="5">ULC01-AL-PT</td></tr>
<tr><td>Nombre de PNR</td><td colspan="5">PNR01-AL-PT</td></tr>
<tr><td colspan="6">Tipo de PNR</td></tr>
<tr><td>1</td><td>Condición Externa</td><td colspan="4">No</td></tr>
<tr><td>2</td><td>Profundidad de Cola</td><td>Si</td><td>Profundidad</td><td colspan="2">50</td></tr>
<tr><td>3</td><td>Tiempo de Persistencia</td><td colspan="4">No</td></tr>
<tr><td colspan="6">Descripción del a PNR</td></tr>
<tr><td colspan="6">No se han detectado ninguna restricción para la ULC= ULCC01-AL-PT, en consecuencia se le asigno una profundidad de cola máxima por razones de performance.</td></tr>
<tr><td>Firma</td><td colspan="5">  </td></tr>
<tr><td>Documento Versión</td><td colspan="5">0.0.1</td></tr>
</table>

Figura 4.11. Documento de Evaluación de PNR completo.

Aquí se observa como se ha completado el último documento referente a la recolección de información que servirá para generar el servicio, de aquí en mas se pasara a realizar pruebas de calidad y performance del sistema antes de incluir la funcionalidad UNDO/REDO.

4.4.1.4. Técnica Estudio de Viabilidad (F1-T4-T)

Esta técnica intenta determinar si es necesario la inclusión del un servidor de proximidad para la inclusión del servicio UNDO/REDO. Los pasos para desarrollarla se presentan en la Tabla 4.9.

Técnica de Evaluación de Viabilidad
Entradas: Especificación para el estudio de Viabilidad del servidor de Proximidad Salidas: Estudio de Viabilidad
Paso 1. Cumplimentar el proceso definido como Evaluación de Viabilidad 1.1. Se ejecutan los pasos definidos en el modelo. Paso 4. Construcción del documento deliberable 4.1. Se crea el documento con la evaluación de viabilidad.

Tabla 4.9. Evaluación de Viabilidad (F1-T4-T)

4.4.1.4.1 Herramienta de documentación para el estudio de Viabilidad.

Aquí se presenta el formato de documento necesario para recolectar la información relacionada a los servidores de proximidad. En este apartado al igual que en el resto del capítulo se hace hincapié a los aspectos prácticos relacionados con la detección del servidor de proximidad. Se debe evaluar documentación general de sistemas y/o del propio sistema (figura 4.12.).

Documento Estudio de Viabilidad			
Fecha		Responsable	
Nombre de Sistema			
Recomendación Final			
Observaciones			
Firma			
Documento Versión	0.0.1		

Figura 4.12. Documento de Evaluación Final de Servidor de Proximidad.

4.4.1.4.2 Ejemplo de uso de la herramienta

A continuación se dará un ejemplo en donde se muestra la forma de trabajar al servidor de proximidad. La fuente de información es la documentación del sistema y/o el sistema, luego de llevar a cabo el proceso descripto en 4.1.6.1 (Especificación del servidor de Proximidad) en la figura 4.13 se muestra el detalle del documento final.

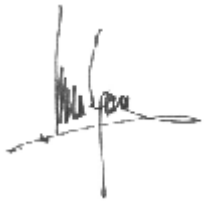
Documento Estudio de Viabilidad			
Fecha	2013-08-06	Responsable	Juan Fernández
Nombre de Sistema	Sistema Administración Curricular		
Recomendación Final	Aceptado		
Observaciones			
Como el valor obtenido es mayor a 5 se recomienda la inclusión de un servidor de proximidad.			
Firma			
Documento Versión	0.0.1		

Figura 4.13. Documento de Evaluación Final de Servidor de Proximidad Completo.

4.4.2. Técnicas Utilizadas para las Actividades de la Fase de Pruebas de la Aplicación

Este grupo de técnicas comprende: Técnica para Prueba de usabilidad de sistemas (sección 4.3.2.1) y Técnica de Ejecución de la Prueba de Stress (sección 4.3.2.2).

4.4.2.1. Técnica de Prueba de Usabilidad de Aplicación (F2-T5-T)

Esta técnica define el método como será realizada la prueba de usabilidad de sistema, esta fase solo será hecha en el caso que el sistema anfitrión se encuentre en producción. Esta prueba será contrastada con una que se realizara una vez incluido el servicio en la aplicación anfitriona. Los pasos para desarrollarla se presentan en la Tabla 4.9.

Prueba de usabilidad de sistemas
Entradas: Sistema Anfitrión Salidas: Evaluación de Usabilidad
Paso 1. Tomar el documento especificado en Fase 1. 1.1 Evaluar la aplicación interfaz por interfaz. Para cada punto de la fase 1 etapa 3 (3.1 a 3.9) se le dará un rango de 1 a 5 donde 1 es la más baja calificación y 5 la más alta. Paso 2. Unificación de las opiniones de los evaluadores 2.1. Se procederá a generar un documento unificado con el resumen de los distintos evaluadores, el cual será el documento deliberable de la etapa.

Tabla 4.10. Prueba de Usabilidad de Aplicación (F2-T5-T)

4.4.2.1.1 Herramienta de documentación para detección y evaluación de Prueba de Usabilidad

Las herramientas de documentación de esta sección se han detallado en el apartado: Herramienta de documentación para detección y evaluación de requisitos de usabilidad (4.3.1.1.1.).

4.4.1.1.2 Ejemplo de uso de la herramienta

Las herramientas de documentación de esta sección se han detallado en el apartado: Ejemplo de uso de la herramienta (4.4.1.1.2.).

4.4.2.2. Técnica de Prueba de Carga de Aplicación (F2-T6-T)

En esta etapa se evalúa el tiempo de respuesta del sistema anfitrión en función de la capacidad de carga esperada. Esta prueba será repetida una vez terminada el proceso de inclusión de la funcionalidad y servirá para saber si la aplicación anfitriona sigue manteniendo el mismo comportamiento sobre stress de carga con y sin el servicio. Los pasos para desarrollarla se presentan en la Tabla 4.10.

Prueba de Carga de Aplicación (F2-T6-T)
Entradas: Requerimientos de Sistema + Sistema. ***Salidas:*** Reporte de Carga de Aplicación (**F2-T6-R**)
Paso 1. Detectar en los requisitos de sistemas: 1.1. Concurrencia máxima. 1.2. Tiempo máximo de respuesta. Paso 2. Construcción del programa para la prueba de carga: 2.1. Detectar el mecanismo de comunicación entre la aplicación y la interfaz. 2.2. Construir un programa que simule la concurrencia máxima de usuarios. 2.3. Obtener los datos que serán necesarios para la prueba. Paso 3. Preparación del material y selección del equipo evaluador: 3.1. Selección del experto que evaluará el sistema. 3.2. Disponer de los equipos necesarios para realizar la prueba. Paso 4. Ejecución de la prueba: 4.1. Repetir el proceso de pruebas al menso 3 veces en distintos horarios y días. Paso 5. Generar deliberable de la tarea: 5.1. Se procederá a generar un documento unificado con el resumen de la prueba.

Tabla 4.11. Prueba de Carga de Aplicación (F2-T6-T)

4.4.2.2.1 Herramienta de documentación para detección y evaluación de Pruebas de Carga

En este apartado se presenta el modelo de documentación necesaria para la realización de la prueba de stress del sistema anfitrión. Al igual que en todos los pasos anteriores se hará hincapié en los aspectos prácticos de la recolección de información para generar la prueba de usabilidad.

La primera fuente de información a relevar es la documentación de sistema, en ella debería existir en forma clara y precisa la estimación de usuarios concurrentes que espera el sistema como máximo, pues de esa estimación o certeza en caso de ser aplicaciones no expuestas en Internet, se habría definido la arquitectura del sistemas, en caso de no existir y de encontrarse la aplicación en producción se debería acceder a los archivos de sucesos del sistema para intentar detectar la concurrencia máxima, es un modelo de análisis no intrusivo, también se puede recurrir a herramientas que evalúan en tiempo real la concurrencia que tiene un sistema y así poder detectar la máxima exigencia que posee el sistema. Luego de obtener el “número mágico” de máxima exigencia de la aplicación se procederá a realizar una aplicación que simule la concurrencia máxima del sistema para poder obtener la performance del sistema bajo stress.

En este caso la primera fuente de información para detectar el método de comunicación y la mensajería existente entre la interfaz de usuario y el núcleo del sistema, se ha dejado de lado en esta consideración aplicaciones comúnmente denominadas “espagueti” en las cuales no existe una

separación entre las distintas capas de la aplicación, de no existir la información requerida se deberá realizar un proceso de ingeniería inversa para obtener el modelo de comunicación para poder ser simulado a través de una aplicación. En la figura 4.14 se detalla el modelo de documentación.

Documento de prueba de stress			
Fecha		Responsable	
Nombre de Sistema			
Nombre de Interfaz		Nombre Clave	
Concurrencia Máxima			
Tiempo Esperado		Tiempo Obtenido	
Observaciones			
Repetir para todas las interfaces			
Firma			
Documento Versión			

Figura 4.14. Documento de Prueba de Carga

4.4.2.2.2 Ejemplo de uso de la documentación

Una vez seleccionada la herramienta para realizar la prueba de stress y desarrollada la aplicación que generara la carga sobre la aplicación anfitriona se completara la información detallada a modo de ejemplo en la figura 4.15. En un solo documento se debe agrupar todas las pruebas de stress para ser luego comparadas con el mismo conjunto de pruebas luego de ser agregado el servicio.

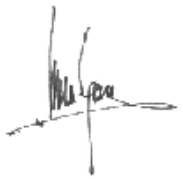
Documento de prueba de stress			
Fecha	2012-10-05	Responsable	Juan Godoy
Nombre de Sistema	Sistema Administración Curricular		
Nombre de Interfaz	Alta de profesores tutores	Nombre Clave	I01-AL-PT
Concurrencia Máxima	50		
Tiempo Esperado	2.3 segundos por usuario	Tiempo Obtenido	2.5
Observaciones			
El tiempo de respuesta aunque no es el esperado por el relevado de en la documentación se encuentra dentro de la tolerancia consultada con los administradores del mismo.			
Repetir para todas las interfaces			
Firma			
Documento Versión	0.0.1		

Figura 4.15. Documento Prueba de Carga Completo.

4.4.3. Técnicas Utilizadas para las Actividades de la Fase de Creación del Servicio

Este grupo de técnicas comprende: Técnica de Configuración de Servicio (sección 4.3.3.1), Técnica de Evaluación de Métodos de Inclusión (sección 4.3.3.2), y Técnica de Programación o rotulado de archivos fuentes (sección 4.3.3.3)

4.4.3.1. Técnica de Configuración de Servicio (F3-T7-T)

En esta etapa se procede a generar el servicio que dará soporte a la aplicación anfitriona para la funcionalidad UNDO/REDO. Como resultado final de esta fase se obtiene el servicio personalizado para la aplicación en cuestión. Los pasos para desarrollarla se presentan en la Tabla 4.11.

Configuración del Servicio
Entradas: Sistema Anfitrión + Diseño de ULC + Definición de PNR + Permisos de Acceso + Estudio de Viabilidad Salidas: Reporte de Configuración
Paso 1. Definir accesos al servicio. 1.1. Con los permisos del sistema anfitrión se debe definir la estructura de accesos. 1.1.1 Se puede optar por no generar una estructura de acceso. 1.2. Definir Documento con Estructura de Permisos. Como Documento intermedio de la fase se crea la estructura de permisos. Paso 2. Cargar Información para el Servicio. 2.1. A través de las pantallas de configuración del sistema cargar. 2.1.1 El detalle obtenido de las ULC. 2.1.2 El detalle obtenido de los PNR. 2.1.3 El detalle obtenido de los Permisos. 2.1.4 El detalle obtenido de Estudio de Viabilidad. Paso 3. Generación del reporte de configuración del servicio Con esto finaliza el proceso de configuración del servicio.

Tabla 4.12. Configuración de Servicio (F3-T7-T)

4.4.3.1.1 Herramienta de documentación de Configuración de Servicio

En este apartado se presenta el modelo de documentación necesaria para concretar la etapa de configuración de servicio y definición de permisos de sistema (figura 4.16).

Documento de Permisos de Servicio					
Fecha			Responsable		
Nombre de Sistema	Sistema de Inscripción				
Nombre de Interfaz			Nombre Clave		
Definir para cada Perfil					
Perfil	Tipo de Acceso.				
	Lectura		Escritura		
Observaciones					
Firma					
Documento Versión	0.0.1				

Figura 4.16. Documento Permisos de Servicios.

A continuación se detalla el formato de documento para el reporte final de la configuración de servicio (figura 4.17).

Documento de Configuración de Servicio			
Fecha		Responsable	
Nombre de Sistema			
Nombre de Interfaz		Nombre Clave	
Seguridad			
ULC			
PNR			
Proxy			
Observaciones			
Firma			
Documento Versión	0.0.1		

Figura 4.17. Documento Configuración de Servicio Final.

4.4.3.1.2 Ejemplo de uso de la documentación

Una vez seleccionada la herramienta para realizar la definición de permisos del servicio (figura 4.18) y el reporte final (figura 4.19).

Documento de Permisos de Servicio					
Fecha	2013-04-02			Responsable	Pedro Sánchez
Nombre de Sistema	Sistema de Inscripción				
Nombre de Interfaz	Alta de Alumnos			Nombre Clave	SI_01_AA
Definir para cada Perfil					
Perfil	Tipo de Acceso.				
Administrador	Lectura	Si		Escritura	Si
Profesor	Lectura	Si		Escritura	No
Alumno	Lectura	Si		Escritura	Si
Observaciones					
Firma					
Documento Versión	0.0.1				

Figura 4.18. Documento Permisos de Servicios Completo.

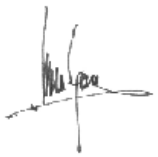
Documento de Configuración de Servicio			
Fecha	2013-04-02	Responsable	Pedro Sánchez
Nombre de Sistema	Sistema de Inscripción		
Nombre de Interfaz	Alta de Alumnos	Nombre Clave	SI_01_AA
Seguridad	Hereda la estructura del sistema anfitrión.		
ULC	Especificada en el documento F1-T2-R.		
PNR	Especificada en el documento F1-T3-R.		
Proxy	Especificada en el documento F1-T4-R. – No Utiliza.		
Observaciones			
El servicio ha sido configurado satisfactoriamente.			
Firma			
Documento Versión	0.0.1		

Figura 4.19. Documento Configuración de Servicio Final Completo.

4.4.3.2. Técnica de Evaluación de Métodos de Inclusión (F3-T8-T)

Aquí el equipo a cargo de la inclusión de la funcionalidad UNDO/REDO se encargara de definir cuál es el mejor modelo de inclusión en la aplicación anfitriona. Como resultado de este esfuerzo se

obtendrá la mejor técnica a criterio del equipo de inclusión de la funcionalidad UNDO/REDO en la aplicación anfitriona. Los pasos para desarrollarla se presentan en la Tabla 4.12.

Evaluación de Métodos de Inclusión
Entradas: Sistema Anfitrión + Diseño de ULC + Documentación de Sistemas Salidas: Método de Inclusión
Paso 1. Analizar la documentación del sistema anfitrión. 1.1. En este paso se debe evaluar cual fue el modelo seleccionado para la construcción del sistema anfitrión. 1.2. Verificar que la documentación este alineada con la versión de producción del sistema anfitrión. Paso 2. Definir alternativas para la inclusión 2.1. Definir el conjunto de alternativas para la inclusión del servicio en función de la tecnología utilizada en la aplicación anfitriona. Paso 3. Entrega de Deliberable 3.1. Como deliberable de esta etapa se obtiene el documento con el método de inclusión definido por el equipo de trabajo.

Tabla 4.13. Evaluación del Método de Inclusión (F3-T8-T)

4.4.3.2.1 Herramienta de documentación para evaluar metodología de inclusión

A continuación se definirá la documentación a presentar para el proceso el método de inclusión de la funcionalidad dentro de la aplicación anfitriona, esto se detalla en la figura 4.20.

Documento metodológica de inclusión			
Fecha		Responsable	
Nombre de Sistema			
Opción Seleccionada			
Observaciones			
Firma			
Documento Versión			

Figura 4.20. Documento Metodología de Inclusión.

4.4.3.2.2 Ejemplo de uso de la documentación

Se presenta en este apartado la documentación de ejemplo relacionada a la selección del método de inclusión. Al igual que el resto de la metodología se presenta una visión estrictamente práctica de la inclusión de la metodología dentro de la aplicación anfitriona (figura 4.21).

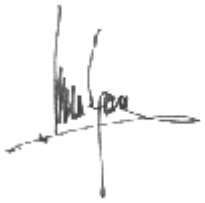
Documento metodológica de inclusión			
Fecha	2012-10-05	Responsable	Juan Godoy
Nombre de Sistema	Sistema Administración Curricular		
Opción Seleccionada	Ajax – Json.		
Observaciones			
El modelo de comunicación seleccionado entre la aplicación el servicio es Json por ser un cuasi-estándar de las comunicaciones de internet y por contar con desarrolladores que lo conocen en profundad. Para agregar las llamadas dentro de la aplicación anfitriona se utilizara Ajax por ser el modelo natural de trabajo junto a Json y puesto que la aplicación anfitriona ya trabaja con la Ajax no se vislumbran problemas en el proceso de integración.			
Firma			
Documento Versión	0.0.1		

Figura 4.21. Documento Metodología de Inclusión Completa.

4.4.3.3. Técnica de Programación o Rotulado de Archivos Fuentes (F3-T9-T)

En esta etapa se procede a incluir las llamadas al sistema en la aplicación, como resultado de esta etapa se obtendrá el sistema anfitrión con sus respectivas llamadas al servicio. Los pasos para desarrollarla se presentan en la Tabla 4.13.

Inclusión del Servicio en la Aplicación (F3-T9-T)
Entradas: Sistema + Método de Inclusión ***Salidas:*** Sistema + Invocación a Servicio (**F3-T9-R**)
Paso 1. Definición de la documentación requerida. 1.1. Definir el modelo de mensajería a utilizar. 1.2. Generación del plan de tiempos. Paso 2. Proceder a la inclusión según el plan definido anteriormente. 2.1. Proceder al proceso de programación. Paso 3. Generar deliberable de la tarea: 3.1 Generar Reporte con la mensajería del Servicio.

Tabla 4.14. Inclusión del Servicio en la Aplicación (F3-T9-T)

Aquí se deben seguir los lineamientos de cada empresa para el proceso de modificación de software, la tabla antes referenciada se provee a modo de ejemplo ilustrativo del conjunto de pasos que deberían ser hechos.

4.4.3.3.1 Herramienta de documentación para inclusión de servicio

A continuación se definirá la documentación a presentar para el proceso de inclusión de servicio figura 4.22.

Documento de Inclusión del Servicio			
Fecha		Responsable	
Nombre de Sistema	Sistema Administración Curricular		
Nombre de Interfaz		Nombre Clave	
Definir para cada Mensaje			
Mensaje			
Observaciones			
Firma			
Documento Versión	0.0.1		

Figura 4.22. Documento Inclusión de Servicio.

4.4.3.2.2 Ejemplo de uso de la documentación

Se presenta en este apartado la documentación de ejemplo relacionada a la selección del método de inclusión. Al igual que el resto de la metodología se presenta una visión estrictamente práctica de la inclusión de la metodología dentro de la aplicación anfitriona (figura 4.23).

4.4.4. Técnicas Utilizadas para las Actividades de la Fase de Pruebas del Servicio

Este grupo de técnicas comprende: Técnica de Prueba de usabilidad de sistemas (sección 4.3.4.1), Técnica de Ejecución de la Prueba de Carga (sección 4.3.4.2), y Técnica de Evaluación de Resultados (sección 4.3.4.3).

4.4.4.1. Técnica de Prueba de Usabilidad de Sistemas (F4-T10-T)

En este caso se repite lo hecho en 4.3.2.1., se repite la misma técnica con los mismos parámetros pero en este punto ya se ha incluido en el sistema anfitrión la funcionalidad. La única salvedad es que el grupo de usuarios y/o expertos que realiza la prueba debe ser distinto para que el conocimiento previo que han adquirido durante la ejecución de la prueba anterior no contamine la prueba actual.

Documento de Inclusión del Servicio			
Fecha	2013-04-20	Responsable	Reinaldo Álvarez
Nombre de Sistema	Sistema Administración Curricular		
Nombre de Interfaz	Alta de profesores tutores	Nombre Clave	I01-AL-PT
Definir para cada Mensaje			
Mensaje	APL. ->SRV.	SRV. ->APL.	
Inicio de sesión	Mensaje: alfanumérico de 5. Token: alfanumérico de 20. Usuario: alfanumérico de 10.	Aceptación: booleano	
Envío de datos	Mensaje: alfanumérico de 5. Token: alfanumérico de 20. Usuario: alfanumérico de 10. Interfaz: alfanumérico de 10. Dato: alfanumérico variable (max. 512)	Aceptación: booleano	
Pedido de datos	Mensaje: alfanumérico de 5. Token: alfanumérico de 20. Usuario: alfanumérico de 10. Interfaz: alfanumérico de 10.	Profundidad: numérico Cantidad de Datos: numérico Página: numérico Dato: alfanumérico variable (max. 512)	
Más datos	Mensaje: alfanumérico de 5. Token: alfanumérico de 20. Usuario: alfanumérico de 10. Interfaz: alfanumérico de 10. Página: numérico	Profundidad: numérico Cantidad de Datos: numérico Página: numérico Dato: alfanumérico variable (max. 512)	
Cierre de sesión	Mensaje: alfanumérico de 5. Token: alfanumérico de 20. Usuario: alfanumérico de 10.	Aceptación: booleano	
Observaciones			
Mensajería Json.			
Firma			
Documento Versión	0.0.1		

Figura 4.23. Documento Inclusión del Servicio Completa.

Prueba de Usabilidad del Servicio (F4-T10-T)
Entradas: Sistema + Servicio Incluido. ***Salidas:*** Reporte de Usabilidad de Aplicación con Servicio (**F4-T10-T**)
Paso 1. Repetir el proceso definido en F1-T1-T. Paso 2. Generar deliberable de la tarea

Tabla 4.15. Prueba de Usabilidad del Servicio (F3-T10-T)

4.4.4.1.1 Herramienta de documentación para detección y evaluación de Prueba de Usabilidad

Las herramientas de documentación de esta sección se han detallado en el apartado: Herramienta de documentación para detección y evaluación de requisitos de usabilidad (4.3.1.1.1.).

4.4.4.1.2 Ejemplo de uso de la herramienta

Las herramientas de documentación de esta sección se han detallado en el apartado: Ejemplo de uso de la herramienta (4.3.1.1.2.).

4.4.4.2. Técnica de Ejecución de la Prueba de Carga (F4-T11-T)

Aquí se repite la prueba de carga realizada en 4.3.2.2., a diferencia del apartado anterior no existe ninguna salvedad se debe ejecutar tal cual se realizo la primera prueba de carga. El deliberable de la fase debe nombrar que es la prueba realizada a posterior de la inserción del servicio en el sistema anfitrión.

Prueba de Carga del Servicio (F4-T11-T)
Entradas: Sistema + Invocación a Servicio ***Salidas:*** Reporte de Carga de Aplicación con Servicio (F4-T11-R)
Paso 1. Tomar el programa construido para la ejecución de la prueba en la F2-T6-T Paso 2. Paso 2. Preparación del material y selección del equipo evaluador 2.1. Selección del experto que evaluará el sistema. 2.2. Disponer de los equipos necesarios para realizar la prueba. Paso 3. Ejecución de la prueba 3.1. Realizar una prueba de validación. Paso 4. Generar deliberable de la tarea

Tabla 4.16. Prueba de Carga del Servicio (F3-T11-T)

4.4.4.2.1 Herramienta de documentación para detección y evaluación de Pruebas de Carga

En este apartado se repetirá el proceso lo hecho en la sección 4.3.2.2.1 Herramienta de documentación para detección y evaluación de Pruebas de Stress.

4.4.4.2.2 Ejemplo de uso de la documentación

Para ver el ejemplo de llenado de documentación observar el apartado 4.3.2.2.2 Ejemplo de uso de la documentación

4.4.4.3. Técnica de Evaluación de Resultados (F4-T12-T)

En este punto se realiza la comparación de las distintas evaluaciones y se analiza si el proceso de inclusión ha sido bien concluido y el sistema anfitrión puede seguir en producción. Los pasos para desarrollarla se presentan en la Tabla 4.14.

Evaluación (F4-T12-T)
Entradas: Reporte de Usabilidad + Reporte de Carga (F2-T5-R) (F4-T10-R) (F2-T6-R) (F4-T11-R) ***Salidas:*** Reporte de Aceptación Final (**F4-T12-R**)
Paso 1. Comparar los Deliberables Detectar F2-T5-R con F4-T10-R. Paso 2. Comparar los Deliberables Detectar F2-T6-R con F4-T11-R. Paso 3. Evaluación final: 3.1. Si ambas comparaciones son satisfactorias se dará por concluido el proceso. 3.2. En caso contrario se iterara desde F3-T7-T. Paso 4. Generar deliberable de la tarea: 4.1. Este deliberable es la aceptación del proceso de inclusión.

Tabla 4.17. Evaluación (F4-T12-T)

4.4.4.3.1 Herramienta de documentación para detección y evaluación final

En este apartado se presenta la documentación asociada al reporte final de evaluación sobre el proceso de inclusión de la funcionalidad UNDO/REDO (figura 4.24.).

Documento de evaluación final			
Fecha		Responsable	
Nombre de Sistema			
Recomendación Final			
Observaciones			
Firma			
Documento Versión			

Figura 4.24. Documento de Evaluación Final

4.4.4.3.2 Ejemplo de uso de la documentación

Se presentara un ejemplo de documentación completa sobre la evaluación final del sistema incluida la funcionalidad UNDO/REDO (figura 4.25.).

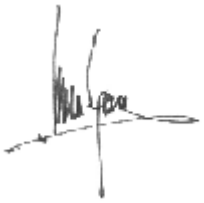
Documento de evaluación final			
Fecha	2012-10-05	Responsable	Juan Godoy
Nombre de Sistema	Sistema Administración Curricular		
Recomendación Final	Aceptado		
Observaciones			
De la comparación de las pruebas realizadas no se observan cambios significantes antes y después de la inclusión de la funcionalidad de Undo/Redo. En consecuencia se ha logrado el objetivo de agregar la funcionalidad sin modificar la performance y la usabilidad existente del sistema anfitrión.			
Firma			
Documento Versión	0.0.1		

Figura 4.25. Documento de Evaluación Final Completo Aceptado.

También se puede presentar el caso en que luego de la evaluación se vuelve hacer el proceso y se dan sugerencias a este respecto (figura 4.26.), según las recomendaciones se precederá según lo convenido y se repetirán las etapas a mejorar.

Este proceso se iterara hasta obtener la aceptación del proceso de inclusión y se proceda con la puesta en producción del sistema anfitrión incluida la funcionalidad UNDO/REDO.

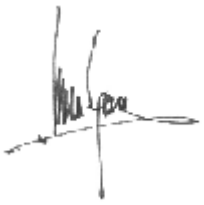
Documento de evaluación final			
Fecha	2012-10-05	Responsable	Juan Godoy
Nombre de Sistema	Sistema Administración Curricular		
Recomendación Final	Rechazado		
Observaciones			
De la comparación de las pruebas realizadas ha sido satisfactoria la prueba de stress pero sobre la evaluación de usabilidad se ha detectado un decremento del mismo, el cual es inamisible para la aplicación, se recomienda evaluar nuevamente el método de inclusión y realizar nuevamente las pruebas de usabilidad y stress con la funcionalidad incluida.			
Firma			
Documento Versión	0.0.1		

Figura 4.26. Documento de Evaluación Final Completo Rechazado.

5. CASOS DE VALIDACIÓN

En este capítulo se presenta un conjunto de casos seleccionados para demostrar la viabilidad del modelo propuesto. Se da un primer caso donde se implementa una aplicación Web y se agrega el servicio de UNDO/REDO (Sección 5.1.), como segundo caso se implementa sobre el primer caso una extensión que posibilita acceder a la aplicación desde plataformas móviles (Sección 5.2.), como tercer caso se presenta una reingeniería completa, para que el primer caso puede ser ejecutado en un ambiente de Cloud Computing (Sección 5.3.) y por último a modo de resumen de los casos validados.

5.1. CASO 1: APLICACIÓN WEB

El motivo de la selección de una aplicación Web como caso de validación del proceso de inclusión de la funcionalidad UNDO/REDO se debe a que este tipo de desarrollo representa un amplio conjunto de las aplicaciones desarrolladas en la actualidad. En esta sección se detalla: descripción de la aplicación anfitriona (Sección 5.1.1.) y el proceso de inclusión del servicio en la aplicación (Sección 5.1.2).

5.1.1. Descripción de la Aplicación Anfitriona

En esta sección se presentan en el Cuadro 5.1 partes del documento de relevamiento de las necesidades de la organización pertinentes al desarrollo de la aplicación web.

"… El sistema al cual se le integrará la funcionalidad de UNDO/REDO es un sistema que administra los recursos de una universidad, donde hay profesores, estos toman cursos por día y horario, los cursos son dictados en aulas, los mismos pueden tener requerimientos especiales como ser proyectores o computadoras, además el aula determina la cantidad de alumnos que podrán cursar. Se cuenta con una interfaz para la administración de alumnos los cuales deben registrarse para poder acceder a los cursos que dictan los profesores. Existen tres tipos de perfiles de usuarios para la aplicación, una es el administrador con permiso general para acceder a todo el sistema, el perfil profesor que puede definir el curso que dictará, el día y el horario del mismo y si este necesita algún requerimiento especial para su dictado. Con esto se asigna automáticamente las aulas para los cursos. Con la grilla de cursos y horarios el tercer perfil, alumnos, que puede darse de alta e inscribe hasta el tope máximo que permite cada aula… "
"… Surge la necesidad por parte del usuario que la aplicación tenga una interfaz amigable y sencilla orientada a usuarios con diferentes niveles de experticia en el manejo de computadoras. A su vez el sistema tiene picos de uso, los cuales se presentan en el momento donde los alumnos se inscriben, el tiempo de respuesta estipulado para todo el sistema es de menos de 5 segundos y 100 usuarios como máximo.

Cuadro 5.1.a. partes del documento de relevamiento de las necesidades de la organización pertinentes al desarrollo de la aplicación web.

La documentación que posee el sistema se reduce a una descripción general del sistema, donde se detalla las características y se definen un conjunto de objetivos que deberá cumplir el sistema; sumado a esto se cuenta con un conjunto de casos de uso de las principales acciones del mismo…"

"… Este sistema se alinea con las aplicaciones de una sola página (Single Page Applications), estas son aplicaciones donde dentro de un solo marco que es representado en HTML se ejecuta todo el sistema , la aplicación así contenida evita saltos entre página y página, intentando simular una aplicación de escritorio, todo esto administrado por Ajax…"

"… En la figura se ilustra un fragmento de la aplicación en cuestión, la misma es una tabla donde los alumnos cargan el número de matrícula, nombres, apellido y mail, esta simple interfaz es representativa para el proceso de inclusión del servicio de UNDO/REDO…"

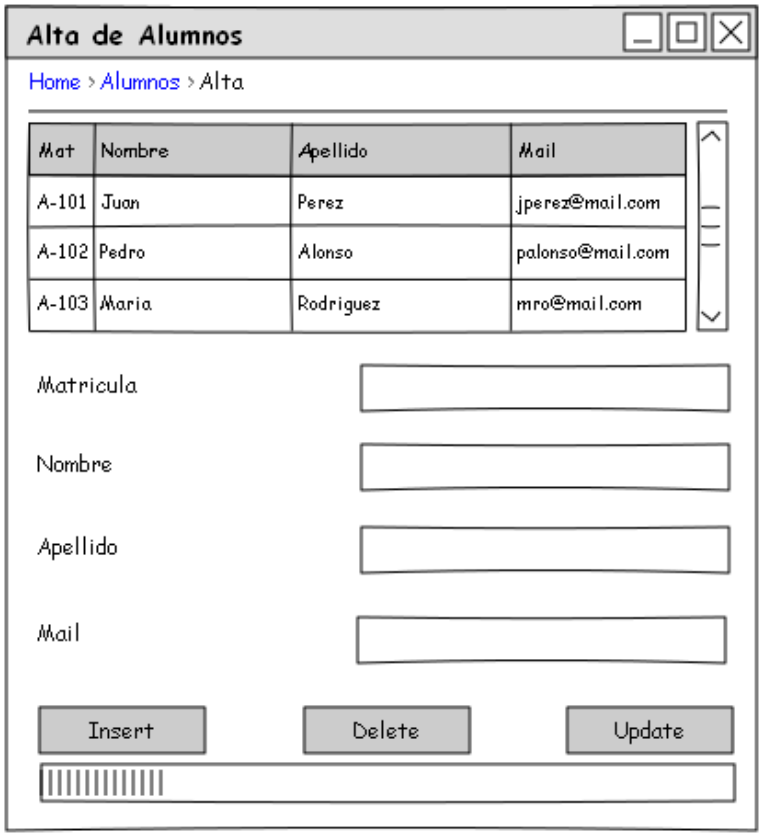


Figura. Fragmento de la aplicación: Interfaz de Usuario.

"… El modelo de iteración de la aplicación es el siguiente, si se desea insertar un registro el mismo debe ser cargado en los campos de texto que se ven en la parte inferior de la pantalla, una vez terminado esto se hace click en el botón "Insert", y con esta acción queda el registro agregado a la base, la única validación que hace la aplicación es el número de matricula, éste no puede estar repetido. Si uno desea borrar un registro de la tabla, lo selecciona haciendo click sobre el número de matricula, y este es cargado sobre los campos de texto editables en la parte inferior de la pantalla; luego de esto se hace click sobre el botón "Delete" y el registro es eliminado de la base. Si es necesario modificar un registro se selecciona el registro, se carga en los campos de texto, se realiza la modificación en los campos y se hace click sobre el botón "Update", esta acción actualiza el contenido del legajo en la base..."

"…La infraestructura seleccionada es de uso común en el desarrollo de soluciones para el dominio Web. Esta utiliza el lenguaje de programación PHP, que se puede encontrar en la página www.php.net, esta selección se debe a su amplia difusión como lenguaje de programación para ambientes Web, además de ser de uso libre y gratuito; como puerta de enlace entre los usuarios y el sistemas se usa un servidor Web, para el mismo se ha seleccionado el servidor Nginx, este se localiza en http://nginx.org/, el motivo de selección de este servidor de páginas estáticas para ambientes Web es por ser gratuito y de código

Cuadro 5.1.b. partes del documento de relevamiento de las necesidades de la organización pertinentes al desarrollo de la aplicación web.

abierto seguido a esto, este servidor se encuentra en el estado del arte con respecto a los avances de los servidores Web, dentro de los avances que contiene, es de destacar que soluciona el problema de 10K, es decir soporta más de 10.000 conexiones concurrentes. Como base de datos se ha seleccionado MySQL, que se puede encontrar en la página www.mysql.com, la razón de esta selección es al igual que los demás componentes antes descriptos es de uso gratuito y se alinea a las políticas de software libre, sumado a esto su amplia aceptación en la comunidad informática ha servido para inclinar la balanza a su favor. En la interfaz de usuario se ha utilizado HTML5 y Java Script con Sencha como Framework, que se puede encontrar en la página http://www.sencha.com/, con Ajax y la mensajería JSon la responsable de comunicar la interfaz del usuario con el servidor. Como marco de trabajo se ha seleccionado a Codeigniter, que se puede encontrar en la página http://ellislab.com/codeigniter/, que enmascara un patrón Model-View-Controler y el Framework Active Record que permite acceder a una base de datos relacional, en este caso MySQL. La selección de este marco de trabajo se debe a su amplia difusión en el ambiente de desarrolladores de PHP por proveer un conjunto de facilidades como ser vistas pre-configuradas, manejo de mail, manejo de sesión, configuración de seguridad, etc. junto a una sencilla forma de adaptar el entorno de trabajo a las necesidades de cada proyecto…"

"… Es necesario para la concreción de las diversas tareas se requerirá de la participación de diferentes perfiles; este equipo está compuesto por un usuario del sistema en la actualidad llamado Juan Pérez, un experto en usabilidad Pedro Sánchez miembro del equipo que está encargado de incluir la funcionalidad de Undo/Redo en la aplicación, Juan Fernández es el arquitecto de sistemas encargado de liderar al equipo encargado de incluir la funcionalidad de Undo/Redo. Reinaldo Álvarez programador con conocimientos en PHP, HTML, Ajax, MySQL y Python…"

"… Las computadoras para cada uno de los integrantes han sido asignadas y todos cuentan con los permisos de acceso necesarios así como con el software para realizar su trabajo según el perfil definido…"

"… Como todo proceso que involucra diversas tareas y pasos a realizar, se hace necesario contar con determinadas herramientas para que éstas sean de fácil concreción. En este sentido el programador Reinaldo Álvarez realizo un programa específico para el análisis de los archivos de sucesos del servidor MySQL, la razón de este requerimientos es permitir a un usuario no experto en el análisis de archivos de esta base de datos poder analizar el mismo..."

"…Este programa escrito en Python funciona por línea de comandos, genera un archivo en formato de texto que representa el conjunto de transacciones realizadas por los diversos usuarios del sistema.

Los parámetros de entrada al mismo son 3, los primeros dos son el rango de fechas, desde y hasta, donde se desea consultar el log, el tercer parámetro el nombre del archivo donde será dejada la información. El formato de salida del mismo es: fecha y hora que se realizó la operación, nombre del usuario que la realizo, tabla sobre la cual se realizó la operación y detalle de la operación que se realizó. A continuación se especifica un resumen del mismo que será considerado para el proceso de inclusión..."

"…

Jlr, 20130103-101224, alumnos,insert (mat=A-101, Nombre=Juan, Apellido=Perz, Mail=jerez@mail.com)
Jlr, 20130103-101433, alumnos, update (mat=A-101, Apellido=perez)
Jlr, 20130103-101501, alumnos, update (mat=A-101, Apellido=Perez)
Jlr, 20130103-101624, alumnos, insert (mat=A-102, Nombre=Pedro, Apellido=alonso, Mail=plonso@mail.com)
Jlr, 20130103-101758, alumnos, update (mat=A-102, Apellido=Alonso)
Jlr, 20130103-102052, alumnos, insert (mat=A-103, Nombre=Maria, Apellido=Rodriguez, Mail=mrod@mail.com)
Jlr, 20130103-102505, alumnos, update (mat = A-103, Mail=mro@mail.com)
Jlr, 20130103-102825, alumnos, update (mat = A-102, Mail=pelonso@mail.com)
Jlr, 20130103-103301, alumnos, update (mat = A-102, Mail=palonso@mail.com)
Jlr, 20130103-104055, alumnos, delete (mat = A-101)
Jlr, 20130103-104224, alumnos, insert (mat=A-101, Nombre=Juan, Apellido=Perz, Mail=jperez@mail.com)
…"

Cuadro 5.1.c. partes del documento de relevamiento de las necesidades de la organización pertinentes al desarrollo de la aplicación web.

5.1.2. Proceso de Inclusión del Servicio en la Aplicación

En esta sección se detallan los resultados de la ejecución del proceso propuesto a la aplicación descripta en el siguiente orden: Resultado de la Fase Modelado de Servicio (Sección 5.1.2.1), Resultado de la Fase Pruebas de la Aplicación (Sección 5.1.2.2), Fase Creación del Servicio (Sección 5.1.2.3), Resultado de la Fase Pruebas del Servicio (Sección 5.1.2.4).

5.1.2.1. Fase Modelado de Servicio (F1)

En esta sección se presentan los resultado de las tareas para completar esta fase del proceso de inclusión, estas son: Análisis de Usabilidad (Sección 5.1.2.1.1), Detección de Unidades Lógicas de Cambio (Sección 5.1.2.1.2), Detección de Puntos de No Retorno (Sección 5.1.2.1.3) y Evaluación de Viabilidad de Servidor de Proximidad (Sección 5.1.2.1.4).

5.1.2.1.1. Tarea Análisis de Usabilidad (F1-T1-T)

Como guía para cumplimentar esta tarea se seguirán los pasos definidos en la tabla 4.6 (Técnica de Detección y Evaluación de Requisitos de Usabilidad); para esta tarea se detallan las entradas, técnicas y salidas en la tabla 4.5 (Fases del Proceso) y el flujo del proceso se detalla en la figura 4.1 (Flujo de Proceso).

Entrada 1: Requisitos de Sistema.

> Del documento de relevamiento (cuadro 5.1) se extrae el siguiente texto *"…Surge la necesidad por parte del usuario que la aplicación tenga una interfaz amigable y sencilla orientada a usuarios con diferentes niveles de experticia en el manejo de computadoras…"*, esto se asume que es un requisito de usabilidad, esto no puede ser utilizado como información relevante para el próximo paso, inspección heurística.

Entrada 2: Sistema.

> Del extracto del documento de relevamiento (cuadro 5.1) se detalla la interfaz a evaluar "*…Figura. Fragmento de la aplicación: Interfaz de Usuario…*"

Desarrollo de Pasos:

Paso 1: Definir documento de evaluación para cada interfaz.

1.1. Nombre de Interfaz a evaluar asignados por el equipo de desarrollo.

Se fijan los siguientes términos:

Nombre de Sistema: Sistema de Inscripción.

Nombre de Interfaz: Alta de Alumnos.

Nombre Clave: SI_01_AA.

1.2. Descripción general de la interfaz.

Del documento de relevamiento (cuadro 5.1) se extrae el siguiente texto "*…donde los alumnos cargan el número de matrícula, nombres, apellido y mail…"*

Paso 2: Preparación del material y selección del equipo evaluador.

2.1. Selección de usuarios y/o expertos en usabilidad que evaluarán el sistema.

Del documento de relevamiento (cuadro 5.1) se extrae el siguiente texto "*…un usuario del sistema en la actualidad llamado Juan Pérez, un experto en usabilidad Pedro Sánchez…".* Se selecciona a Juan Pérez y Pedro Sánchez para la evaluación.

2.2. Disponer de los equipos necesarios para realizar la prueba.

Del documento de relevamiento (cuadro 5.1) se extrae el siguiente texto "*…Las computadoras para cada uno de los integrantes han sido asignadas y todos cuentan con los permisos de acceso necesarios…".* Como se ha definido se cuenta con los accesos necesarios.

Paso 3: Desarrollo del método de Inspección Heurística de Molich y Nielsen.

Se detalla el proceso de evaluación de la aplicación mediante el método de inspección heurística, para esto se siguen los pasos detallados en la tabla 4.6 (Técnica de Detección y Evaluación de Requisitos de Usabilidad).

Finalizada la primera etapa del proceso de evaluación, el equipo ha entregado las siguientes tablas (Figura 5.1, Figura 5.2).

Paso 4: Unificar opiniones de los evaluadores y generar deliberable de la tarea.

Se procederá a generar un documento unificado con el resumen de los distintos evaluadores. Tomando los documentos generados por los auditores (Figura 5.1, Figura 5.2) se calcula el promedio para cada punto a evaluar y se generar un documento final.

Documento de Evaluación de Interfaz			
Fecha	2013-02-10	Responsable	Juan Pérez
Nombre de Sistema	Sistema de Inscripción		
Nombre de Interfaz	Alta de Alumnos	Nombre Clave	SI_01_AA
Descripción General			
Esta interfaz es utilizada para la carga, baja o modificaciones de alumnos.			
Puntos a Evaluar		Valor (Rango de 1-5) 1=Malo - 5=Excelente	
1.	Los cuadros de diálogos son simples y naturales.	3	
2.	La aplicación habla el lenguaje del usuario.	1	
3.	Se minimiza la carga de memoria en el usuario.	4	
4.	Hay consistencia en los datos pedido por la interfaz.	4	
5.	La interfaz proporciona información de proceso.	2	
6.	La interfaz proporciona métodos de salida.	2	
7.	La interfaz proporciona caminos abreviados.	3	
8.	La interfaz proporciona mensajes de error.	4	
9.	La aplicación previene errores.	2	
Firma			
Documento Versión	0.0.1		

Figura 5.1. Evaluación Final del Proceso de Inspección Heurística de Juan Pérez.

Documento de Evaluación de Interfaz			
Fecha	2013-02-12	Responsable	Pedro Sánchez
Nombre de Sistema	Sistema de Inscripción		
Nombre de Interfaz	Alta de Alumnos	Nombre Clave	SI_01_AA
Descripción General			
Esta interfaz es utilizada para la carga, baja o modificaciones de alumnos.			
Puntos a Evaluar		Valor (Rango de 1-5) 1=Malo - 5=Excelente	
1.	Los cuadros de diálogos son simples y naturales.	4	
2.	La aplicación habla el lenguaje del usuario.	1	
3.	Se minimiza la carga de memoria en el usuario.	5	
4.	Hay consistencia en los datos pedido por la interfaz.	3	
5.	La interfaz proporciona información de proceso.	2	
6.	La interfaz proporciona métodos de salida.	2	
7.	La interfaz proporciona caminos abreviados.	3	
8.	La interfaz proporciona mensajes de error.	3	
9.	La aplicación previene errores.	3	
Firma			
Documento Versión	0.0.1		

Figura 5.2. Evaluación Final del Proceso de Inspección Heurística de Pedro Sánchez.

Salida: Evaluación de Usabilidad (**F1-T1-R**).

En la figura 5.3 (Evaluación Final del Proceso de Inspección Heurística) se presenta el documento final, que representa la evaluación de usabilidad para la interfaz "Alta de Alumnos" del "Sistema de Inscripción".

Resumen de Evaluación de Interfaz			
Fecha	2013-02-12	Auditor Líder	Juan Fernández
Nombre de Sistema	Sistema de Inscripción		
Nombre de Interfaz	Alta de Alumnos	Nombre Clave	SI_01_AA
Descripción General			
Esta interfaz es utilizada para la carga, baja o modificaciones de alumnos.			
Puntos a Evaluar		Valor (Rango de 1-5) 1=Malo - 5=Excelente	
1.	Los cuadros de diálogos son simples y naturales.	3.5	
2.	La aplicación habla el lenguaje del usuario.	1	
3.	Se minimiza la carga de memoria en el usuario.	4.5	
4.	Hay consistencia en los datos pedido por la interfaz.	3.5	
5.	La interfaz proporciona información de proceso.	2	
6.	La interfaz proporciona métodos de salida.	2	
7.	La interfaz proporciona caminos abreviados.	3	
8.	La interfaz proporciona mensajes de error.	3.5	
9.	La aplicación previene errores.	2.5	
Firma			
Documento Versión	0.0.2		

Figura 5.3. Evaluación Final del Proceso de Inspección Heurística.

5.1.2.1.2. Detección de Unidades Lógicas de Cambio (F1-T2-T)

Como guía para cumplimentar esta tarea se seguirán los pasos definidos en la tabla 4.7 (Técnica de Detección de ULC); para esta tarea se detallan las entradas técnicas y salidas en la tabla 4.5 (Fases del Proceso) y el flujo del proceso se detalla en la figura 4.1 (Flujo de Proceso).

Entrada 1: Requerimientos de Sistema.

Del documento de relevamiento (cuadro 5.1) se extrae el siguiente texto *"…sumado a esto se cuenta con un conjunto de casos de uso de las principales acciones del mismo..."*. Se accede al caso de uso *"Alta de Alumnos"*, figura 5.4 donde se detalla el caso de uso.

Entrada 2: Sistema.

Del extracto del documento de relevamiento (cuadro 5.1) se detalla la interfaz a evaluar *"…Figura. Fragmento de la aplicación: Interfaz de Usuario…"*.

Entrada 3: Evaluación de Usabilidad (**F1-T1-R**).

El deliberable del proceso de evaluación de usabilidad se detalla en figura 5.3 (Evaluación Final del Proceso de Inspección Heurística).

Especificación de Casos de Uso – Alta de Alumnos

Descripción Breve

Esta interfaz es utilizada para mantener el padrón de alumnos de la universidad.

Actores

1. Actor Principal – Alumnos

Precondición

No estar registrado o querer hacer una cuenta nueva en el sistema.

Pos condición

Registrar un nuevo alumno en el sistema.

Flujo de Eventos

1. Flujo Principal - Alta de Alumno:
   - 1.1 Acceso:
     Se accede a la interfaz de "Alta de Alumnos", a través de la opción de menú "Alumnos".
   - 1.2 Alta:
     El alumno carga su matrícula, nombre apellido y mail.
   - 1.3 Grabar Datos:
     Con los datos cargado se inserta los valores con el boto de agregar.

2. Flujo Alternativo 1 – Modificación de Alumnos
   - 2.1 Acceso:
     Se accede a la interfaz de "Alta de Alumnos", a través de la opción de menú "Alumnos".
   - 2.2 Modificación:
     Se selecciona el registra a modificar y se cambian los datos que sean necesarios.
   - 2.3 Modificar Datos:
     Con los datos actualizados se hacen permanentes los cambios.

3. Flujo Alternativo 2 - Borrar Alumno
   - 3.1 Acceso:
     Se accede a la interfaz de "Alta de Alumnos", a través de la opción de menú "Alumnos".
   - 3.2 Modificación:
     Se selecciona el registra a borrar.
   - 3.3 Modificar Datos:
     Con los datos seleccionados se hace click en el botón borrar.

Figura 5.4. Caso de Uso Alta de Alumnos.

Desarrollo de Pasos:

Paso 1: Reutilizar la información generada en el paso 1 de la etapa 1.

Se toman los términos definidos en 5.1.2.1.1 (Tarea Análisis de Usabilidad) y se agrega:

Nombre ULC: ULC01-AL-PT.

Paso 2: Preparación del material y selección del equipo evaluador.

2.1. Selección del arquitecto de sistemas que evaluarán el sistema.

Del documento de relevamiento (cuadro 5.1) se extrae el siguiente texto *"…Juan Fernández es el arquitecto de sistemas…".* El arquitecto Juan Fernández será el evaluador responsable de la inclusión de la funcionalidad de UNDO/REDO en la aplicación.

2.2. Disponer de los equipos necesarios para realizar la evaluación.

Del documento de relevamiento (cuadro 5.1) se extrae el siguiente texto *"…Las computadoras para cada uno de los integrantes han sido asignadas y todos cuentan con los permisos de acceso necesarios…".* Como se ha definido se cuenta con los accesos necesarios.

Paso 3: Análisis de interfaces.

Para cada interfaz se debe evaluar:

3.1. Clasificar los datos según el tipo campo de ingreso.

Ejemplo: campo de texto, combo, selección múltiple, etc.

3.2. Evaluar cada campo según la siguiente escala.

3.2.1. El campo es atómico.

3.2.2. Está relacionado con otro campo.

3.2.3. El campo puede ser modificado.

En la figura 5.5 (Evaluación Final del Arquitecto sobre la interfaz "Alta de Alumnos") se detalla el resultado de aplicar el paso 3 sobre la interfaz de "Alta de Alumnos".

Documento de Evaluación de ULC			
Fecha	2013-02-15	Responsable	Juan Fernández
Nombre de Sistema	Sistema de Inscripción		
Nombre de Interfaz	Alta de Alumnos	Nombre Clave	SI_01_AA
Nombre de ULC	ULC01-AL-PT		
Descripción General			
Esta interfaz es utilizada para la carga, baja o modificaciones de alumnos.			
Cantidad de Campos de la interfaz		4	
Para cada Campo de la Interfaz Evaluar			
Campo Matrícula			
1.1	Tipo de Campo	Alfanumérico	
1.2	Identificador de Campo	Matrícula	
1.3	El campo puede ser modificado	Si	
1.4	El campo es atómico.	No	
Campo Nombre			
2.1	Tipo de Campo	Alfanumérico	
2.2	Identificador de Campo	Nombre	
2.3	El campo puede ser modificado	Si	
2.4	El campo es atómico.	No	
Campo Apellido			
3.1	Tipo de Campo	Alfanumérico	
3.2	Identificador de Campo	Apellido	
3.3	El campo puede ser modificado	Si	
3.4	El campo es atómico.	No	
Campo Mail			
4.1	Tipo de Campo	Alfanumérico	
4.2	Identificador de Campo	Mail	
4.3	El campo puede ser modificado	Si	
4.4	El campo es atómico.	No	
Agregar tantos campos como sea necesario			
Firma			
Documento Versión	0.0.1		

Figura 5.5. Evaluación Final del Arquitecto sobre la interfaz “Alta de Alumnos”.

Paso 4: Generar deliberable de la tarea.

4.1. Descartar los campos que no pueden ser modificados.

4.2. Denominar a cada campo relacionado con un identificador de interfaz y de ULC.

4.3. Para el resto de los campos atómicos también darle una identificación de interfaz y ULC.

A continuación se detalla el proceso llevado a cabo para la construcción de la unidad lógica de cambio ULC01-AL-PT. En la figura 5.6 (Detección de ULC) se representa el concepto de ULC, en esta interfaz de carga de profesores, existen tres campos en la

interfaz: Matrícula, Nombre, Apellido y Mail, estos aunque se cargan en forma separada, para un proceso de UNDO/REDO no pueden ser tomados por separado, pues un número de matrícula está asociado unívocamente a un nombre, apellido y mail, es por esta razón que deben ser considerados como una unidad indivisible de información, este mismo proceso de razonamiento se debe llevar a cabo para todos los demás campos de cada interfaz, en la interfaz en cuestión no se encuentran más campos y en consecuencia el análisis de la misma ha finalizado.

Mat	Nombre	Apellido	Mail
A-101	Juan	Perez	jperez@mail.com
A-102	Pedro	Alonso	palonso@mail.com
A-103	Maria	Rodriguez	mro@mail.com

Figura 5.6. Detección de ULC.

Salidas: Definición de ULC (**F1-T2-R**).

En la figura 5.7 (Evaluación Final de la ULC para la interfaz “Alta de Alumnos”) se detalla el documento final como deliberable de la presente tarea.

Documento de Evaluación Final de ULC			
Fecha	2013-02-16	Responsable	Juan Fernández
Nombre de Sistema	Sistema de Inscripción		
Nombre de Interfaz	Alta de Alumnos	Nombre Clave	SI_01_AA
Nombre de ULC	ULC01-AL-PT		
Descripción General			
Esta interfaz es utilizada para la carga, baja o modificaciones de alumnos.			
Campo	Tipo		
Matrícula	Alfanumérico		
Nombre	Alfanumérico		
Apellido	Alfanumérico		
Mail	Alfanumérico		
Firma			
Documento Versión	0.0.1		

Figura 5.7. Evaluación Final de la ULC para la interfaz “Alta de Alumnos”.

5.1.2.1.3. Técnica de Detección de Puntos de No Retorno (F1-T3-T)

Como guía para cumplimentar esta tarea se seguirán los pasos definidos en la tabla 4.8 (Técnica de Detección de PNR); para esta tarea se detallan las entradas, técnicas y salidas en la tabla 4.5 (Fases del Proceso) y el flujo del proceso se detalla en la figura 4.1 (Flujo de Proceso).

Entrada 1: Requerimientos de Sistema.

En el cuadro 5.1 no se detecta ninguna especificación referente a situaciones donde no puedan presentarse situaciones donde no se pueda volver hacia atrás al sistema.

Entrada 2: Sistema.

Del extracto del documento de relevamiento (cuadro 5.1) se detalla la interfaz a evaluar *"…Figura. Fragmento de la aplicación: Interfaz de Usuario…"*.

Entrada 3: Definición de ULC (**F1-T2-R**).

Se toma el documento final representado en la figura 5.7 (Evaluación Final de la ULC para la interfaz "Alta de Alumnos".).

Desarrollo de Pasos:

Paso 1: Reutilizar la información generada en el paso 1 de la etapa 1

Se toman los términos definidos en 5.1.2.1.2 (Detección de Unidades Lógicas de Cambio) paso 1 y se agregan los siguiente:

Nombre PNR: PNR01-AL-PT.

Paso 2: Preparación del material y selección del equipo evaluador.

2.1. Selección del arquitecto de sistemas que evaluarán el sistema.

Del documento de relevamiento (cuadro 5.1) se extrae el siguiente texto *"…Juan Fernández es el arquitecto de sistemas…".* El arquitecto Juan Fernández será el evaluador responsable de la inclusión de la funcionalidad de UNDO/REDO en la aplicación.

2.2. Disponer de los equipos necesarios para realizar la evaluación.

Del documento de relevamiento (cuadro 5.1) se extrae el siguiente texto *"…Las computadoras para cada uno de los integrantes han sido asignadas y todos cuentan con los permisos de acceso necesarios…".* Como se ha definido se cuenta con los accesos necesarios.

Paso 3: Análisis de PNR.

Para cada interfaz se debe evaluar:

3.1. Existe algún requisito de sistema que impida volver a un estado anterior.

No se detecta como se definió en Entrada 1. Tampoco se observa en el sistema ninguna condición, por lo que se define que no existe condición externa para no poder volver hacia atrás.

3.2. Clasificar la probabilidad de modificaciones de la ULC.

Del documento de relevamiento (cuadro 5.1) se extrae el siguiente texto *"…específico para el análisis de los archivos de sucesos del servidor MySQL... Figura: Archivo de eventos con formato…".* En este fragmento se puede observar que la modificación de los datos de la tabla alumnos que interactúa con la interfaz "Alta de Alumnos", representa el 63% de todas operaciones realizadas.

3.3. Clasificar la profundidad de la pila de cada ULC.

Del documento de relevamiento (cuadro 5.1) se extrae el siguiente texto *"…específico para el análisis de los archivos de sucesos del servidor MySQL... Figura: Archivo de eventos con formato…".* Se puede observar que la cantidad de modificaciones realizadas en promedio es de 2 cambios.

Paso 4: Generar deliberable de la tarea.

4.1. Si se detectaron requisitos definir un PNR.

No se han detectado requisitos.

4.2. Si la probabilidad de modificación es alta y la profundidad de pila es alta definir un PNR.

La probabilidad de modificación es alta (63%) pero la profundidad es baja solo 2 cambios, con lo cual no se define un PNR.

4.3. Para todos los demás casos no definir un PNR.

Salidas: Definición de PNR (**F1-T3-R**).

En la figura 5.8 (Deliberable de la etapa de Detección de PNR) se detalla el documento deliberable para la etapa.

Documento de Evaluación de PNR				
Fecha	2013-03-05		Responsable	Juan Fernández
Nombre de Sistema	Sistema de Inscripción			
Nombre de Interfaz	Alta de Alumnos		Nombre Clave	SI_01_AA
Nombre de ULC	ULC01-AL-PT			
Nombre de PNR	PNR01-AL-PT			
Tipo de PNR				
1	Condición Externa	No		
2	Profundidad de Cola	Si	Profundidad	2
3	Tiempo de Persistencia	No		
Descripción del a PNR				
No se han detectado ninguna restricción para la ULC= ULCC01-AL-PT, en consecuencia se le asigno una profundidad de cola máxima por razones de performance.				
Firma				
Documento Versión	0.0.1			

Figura 5.8. Deliberable de la etapa de Detección de PNR.

5.1.2.1.4. Estudio de Viabilidad de Servidor de Proximidad (F1-T4-T)

Como guía para cumplimentar esta tarea se seguirán los pasos definidos en la tabla 4.9 (Técnica de Evaluación de Viabilidad de Servidor de Proximidad); para esta tarea se detallan las entradas, técnicas y salidas en la tabla 4.5 (Fases del Proceso) y el flujo del proceso se detalla en la figura 4.1 (Flujo de Proceso).

Entrada 1: Requerimiento de Sistema.

Se toma el texto redactado en el cuadro 5.1 para su análisis.

Entrada 2: Sistema.

Del extracto del documento de relevamiento (cuadro 5.1) se detalla la interfaz a evaluar "*…Figura. Fragmento de la aplicación: Interfaz de Usuario…*".

Desarrollo de Pasos:

Paso 1: Cumplimentar el proceso definido como Evaluación de Viabilidad

Del documento de relevamiento (cuadro 5.1) se extrae el siguiente texto *"…Juan Fernández es el arquitecto de sistemas…".* El arquitecto Juan Fernández será el encargado será el encargado de llevar adelante el estudio de viabilidad. En la tabla 5.1 se presenta el resultado a las preguntas sugeridas.

ID	Pregunta asociada	Respuesta
1	Existen restricciones de seguridad	Si
2	Deben ser integrados varios servicios	No
3	Existe la posibilidad de cambio en los proveedores de servicios	No
4	Se accederá a la aplicación desde múltiples dispositivos	No
5	Existen cambios frecuentes en la aplicación	No
6	Se desea contar con mejoras pos implementación	No

Tabla 5.1. Respuestas obtenidas.

Conversión a intervalos difusos:

En la tabla 5.2 se detallan la conversión a intervalos difusos, se ha tomado como referencia a la tabla 4.3.

Si	**10**
No	**0**

Tabla 5.2. Resumen de Intervalos Difusos.

Definición de Pesos

En la tabla 5.3 se muestran los pesos luego del cálculo de los intervalos difusos.

ID	Pregunta asociada	Peso
1	Existen restricciones de seguridad	10
2	Deben ser integrados varios servicios	0
3	Existe la posibilidad de cambio en los proveedores de servicios	0
4	Se accederá a la aplicación desde múltiples dispositivos	0
5	Existen cambios frecuentes en la aplicación	0
6	Se desea contar con mejoras pos implementación	0

Tabla 5.3. Cálculo de pesos.

Detalle de cada respuesta

1. Existen restricciones de seguridad: Del documento de relevamiento (cuadro 5.1) se extrae el siguiente texto *"…Existen tres tipos de perfiles de usuarios para la aplicación, una es el administrador con permiso general para acceder a todo el sistema, el perfil profesor que puede definir el curso que dictará, el día y el horario del mismo y si este necesita algún requerimiento especial para su dictado. Con esto se asigna automáticamente las aulas para los cursos. Con la grilla de cursos y horarios el tercer perfil, alumnos, que puede darse de alta e inscribe hasta el tope máximo que permite cada aula..."* En este extracto se detallan tres perfiles de usuarios distintos y permisos diferentes para cada uno, con lo cual se define que existen restricciones de seguridad.
2. Deben ser integrados varios servicios: En el análisis realizado a la documentación (cuadro 5.1) no se hace referencia a ningún otro servicio que será utilizado, con lo cual se define que no deben integrarse otros servicios.
3. Existe la posibilidad de cambio en los proveedores de servicios: Por lo detallado en el punto 2, se deduce que no habrá cambio de proveedores de servicios.
4. Se accederá a la aplicación desde múltiples dispositivos: En el análisis realizado a la documentación (cuadro 5.1) no se hace referencia al tipo de dispositivo y por encontrarse este sistema en un instituto educativo solo cuenta con equipos de escritorio, con lo cual no se accederá a la aplicación desde otros dispositivos.
5. Existen cambios frecuentes en la aplicación. En el análisis realizado a la documentación (cuadro 5.1) no se detecta la existencia de cambios frecuentes en la aplicación.

6. Se desea contar con mejoras pos implementación: En el análisis realizado a la documentación (cuadro 5.1) no se detecta la posibilidad de implementar mejoras pos implementación.

Cálculo de Promedio

La suma de los pesos es 10 esto se divide por 6, la cantidad de preguntas asociadas, como resultado se obtiene 1.66 se redondea a 2.

Como el valor obtenido es menor a 5 no es necesaria la inclusión de un servidor de proximidad.

Salida: Estudio de Viabilidad (**F2-T4-R**).

En la figura 5.9 (Deliberable de la etapa Estudio de Viabilidad) se presenta el documento final, que representa la evaluación del estudio de viabilidad.

Documento Estudio de Viabilidad			
Fecha	2013-04-05	Responsable	Juan Fernández
Nombre de Sistema	Sistema Administración Curricular		
Recomendación Final	Aceptado		
Observaciones			
Como el valor obtenido es menor a 5 no es necesaria la inclusión de un servidor de proximidad.			
Firma			
Documento Versión	0.0.1		

Figura 5.9. Deliberable de la etapa Estudio de Viabilidad.

5.1.2.2. Fase Pruebas de la Aplicación (F2)

En esta sección se presentan los resultados de las tareas para completar esta fase de pruebas de la aplicación, estas son: Prueba de Usabilidad de la Aplicación (Sección 5.1.2.2.1) y Prueba de Carga de la Aplicación (Sección 5.1.2.2.2).

5.1.2.2.1. Prueba de Usabilidad de la Aplicación (F2-T5-T)

Como guía para cumplimentar esta tarea se seguirán los pasos definidos en la tabla 4.10 (Prueba de Usabilidad de Aplicación); para esta tarea se detallan las entradas, técnicas y salidas en la tabla 4.5 (Fases del Proceso) y el flujo del proceso se detalla en la figura 4.1 (Flujo de Proceso).

Entrada 1: Sistema.

Del extracto del documento de relevamiento (cuadro 5.1) se detalla la interfaz a evaluar "*…Figura. Fragmento de la aplicación: Interfaz de Usuario…*".

Desarrollo de Pasos:

Paso 1: Tomar el documento F1-T1-R

1.1 Si la interfaz ha sufrido cambios desde la generación del reporte rehacer F1-T1-T Paso 3.

No se han producido modificaciones.

Salida: Generar deliberable de la tarea (**F2-T5-R**).

En la figura 5.3 (Evaluación Final del Proceso de Inspección Heurística) se presenta el documento final, que representa la evaluación de usabilidad para la interfaz "Alta de Alumnos".

5.1.2.2.2. Prueba de Carga de la Aplicación (F2-T6-T)

Como guía para cumplimentar esta tarea se seguirán los pasos definidos en la tabla 4.11 (Prueba de Carga de Aplicación); para esta tarea se detallan las entradas, técnicas y salidas en la tabla 4.5 (Fases del Proceso) y el flujo del proceso se detalla en la figura 4.1 (Flujo de Proceso).

Entrada 1: Requerimientos de Sistema.

Se toma el texto redactado en el cuadro 5.1 para su análisis.

Entrada 2: Sistema.

Del extracto del documento de relevamiento (cuadro 5.1) se detalla la interfaz a evaluar "*…Figura. Fragmento de la aplicación: Interfaz de Usuario…*".

Desarrollo de Pasos:

Paso 1: Detectar en los requisitos de sistemas.

1.1. Concurrencia máxima.

Del documento de relevamiento (cuadro 5.1) se extrae el siguiente texto *"…100 usuarios como máximo…".* Aquí se detecta que la concurrencia máxima es 100 usuarios.

1.2. Tiempo máximo de respuesta.

Del documento de relevamiento (cuadro 5.1) se extrae el siguiente texto *"…el tiempo de respuesta estipulado para todo el sistema se de menos de 5 segundos…".* El tiempo de respuesta máximo de la aplicación debe ser menos de 5 segundos.

Paso 2: Construcción del programa para la prueba de stress

2.1. De los requerimientos detectar el mecanismo de comunicación entre la aplicación y la interfaz.

Del documento de relevamiento (cuadro 5.1) se extrae el siguiente texto *"…la interfaz de usuario se ha utilizado HTML5 y Java Script, siendo Ajax con mensajería Json…"* El mecanismo de comunicación es HTML con AJAX y JSon.

2.2. Construir un programa que simule la concurrencia máxima de usuarios.

Del documento de relevamiento (cuadro 5.1) se extrae el siguiente texto *"…Reinaldo Álvarez programador con conocimientos en PHP, HTML, Ajax, MySQL y Python…".* El programador Reinaldo Álvarez será el encargado de construir la aplicación para la simulación de carga máxima.

2.3. Obtener los datos que serán necesarios para la prueba.

Del documento de relevamiento (cuadro 5.1) se extrae el siguiente texto *"…cuentan con los permisos de acceso necesarios…".* El arquitecto Juan Fernández será el encargado de proveer los usuarios y claves para la prueba de carga.

Paso 3: Preparación del material y selección del equipo evaluador

3.1. Selección del experto que evaluará el sistema.

Del documento de relevamiento (cuadro 5.1) se extrae el siguiente texto *"…un experto en usabilidad Pedro Sánchez miembro del equipo que está encargado de incluir la funcionalidad…".* Se selecciona al programador Pedro Sánchez para ejecutar la prueba de carga.

3.2. Disponer de los equipos necesarios para realizar la prueba.

Del documento de relevamiento (cuadro 5.1) se extrae el siguiente texto *"…Las computadoras para cada uno de los integrantes han sido asignadas y todos cuentan*

con los permisos de acceso necesarios...". Como se ha definido se cuenta con los accesos necesarios.

Paso 4: Ejecución de la prueba

4.1. Repetir el proceso de pruebas al menos 3 veces en distintos días y horarios.

Pedro Sánchez ejecuta la prueba 3 veces en distinto días y horarios. En la figuras 5.10, 5.11 y 5.12 se muestra el detalle de la ejecución de las diversas pruebas de carga.

Documento de Prueba de Carga					
Fecha	2013-03-25	Hora	09:30	Responsable	Pérez Sánchez
Nombre de Sistema		Sistema de Inscripción			
Nombre de Interfaz		Alta de Alumnos		Nombre Clave	SI_01_AA
Concurrencia Máxima		100			
Tiempo Esperado		Menos de 5 segundos		Tiempo Obtenido	3.9
Observaciones					
Esta interfaz es utilizada para la carga, baja o modificaciones de alumnos.					
Repetir para todas las interfaces					
Firma					
Documento Versión		0.0.1			

Figura 5.10. Deliberable de la primera ejecución.

Documento de Prueba de Carga					
Fecha	2013-03-26	Hora	12:30	Responsable	Pedro Sánchez
Nombre de Sistema		Sistema de Inscripción			
Nombre de Interfaz		Alta de Alumnos		Nombre Clave	SI_01_AA
Concurrencia Máxima		100			
Tiempo Esperado		Menos de 5 segundos		Tiempo Obtenido	3.4
Observaciones					
Esta interfaz es utilizada para la carga, baja o modificaciones de alumnos.					
Repetir para todas las interfaces					
Firma					
Documento Versión		0.0.1			

Figura 5.11. Deliberable de la segunda ejecución.

Documento de Prueba de Carga					
Fecha	2013-03-27	Hora	17:30	Responsable	Pedro Sánchez
Nombre de Sistema		Sistema de Inscripción			
Nombre de Interfaz		Alta de Alumnos		Nombre Clave	SI_01_AA
Concurrencia Máxima		100			
Tiempo Esperado		Menos de 5 segundos		Tiempo Obtenido	3.9
Observaciones					
Esta interfaz es utilizada para la carga, baja o modificaciones de alumnos.					
Repetir para todas las interfaces					
Firma					
Documento Versión		0.0.1			

Figura 5.12. Deliberable de la tercera ejecución.

Paso 5: Generación deliberable de la tarea

5.1. Se procederá a generar un documento unificado con el resumen de la prueba.

Pedro Sánchez genera el reporte final de la prueba de ejecución.

Salida: Reporte de Carga (**F2-T6-R**).

En la figura 5.13 (Deliberable de la etapa Prueba de Carga de la Aplicación) se detalla el documento final como deliberable de la presente tarea.

Documento de Prueba de Carga Final					
Fecha	2013-03-27	Hora	18:00	Responsable	Pedro Sánchez
Nombre de Sistema		Sistema de Inscripción			
Nombre de Interfaz		Alta de Alumnos		Nombre Clave	SI_01_AA
Concurrencia Máxima		100			
Tiempo Esperado		Menos de 5 segundos		Tiempo Obtenido	3.7
Observaciones					
Esta interfaz es utilizada para la carga, baja o modificaciones de alumnos.					
Repetir para todas las interfaces					
Firma					
Documento Versión		0.0.1			

Figura 5.13. Deliberable de la etapa Prueba de Carga de la Aplicación

5.1.2.3. Fase Creación del Servicio (F3)

En esta sección se presentan los resultados de las tareas para completar esta fase de pruebas de la aplicación, estas son: Especificación de la Configuración de Servicio (Sección 5.1.2.3.1), Evaluación del Método de Inclusión (Sección 5.1.2.3.2) e Inclusión del Servicio en la Aplicación (Sección 5.1.2.3.3).

5.1.2.3.1. Especificación de la configuración de Servicio (F3-T7-T)

Como guía para cumplimentar esta tarea se seguirán los pasos definidos en la tabla 4.12 (Especificación de la Configuración del Servicio); para esta tarea se detallan las entradas, técnicas y salidas en la tabla 4.5 (Fases del Proceso) y el flujo del proceso se detalla en la figura 4.1 (Flujo de Proceso).

Entrada 1: Sistema.

Del extracto del documento de relevamiento (cuadro 5.1) se detalla la interfaz a evaluar *"…Figura. Fragmento de la aplicación: Interfaz de Usuario…"*.

Entrada 2: Diseño de ULC.

Se toma el documento F1-T2-R, figura 5.7 Evaluación Final de la ULC para la interfaz "Alta de Alumnos".

Entrada 3: Definición de PNR.

Se toma el documento F1-T3-R, figura 5.8 Deliberable de la etapa de Detección de PNR.

Entrada 4: Permisos de Acceso.

Del documento de relevamiento (cuadro 5.1) se extrae el siguiente texto *"…Existen tres tipos de perfiles de usuarios para la aplicación, una es el administrador con permiso general para acceder a todo el sistema, el perfil profesor que puede definir el curso que dictará, el día y el horario del mismo y si este necesita algún requerimiento especial para su dictado. Con esto se asigna automáticamente las aulas para los cursos. Con la grilla de cursos y horarios el tercer perfil, alumnos, que puede darse de alta e inscribe hasta el tope máximo que permite cada aula…"*.

Desarrollo de Pasos:

Paso 1: Definir accesos al servicio.

1.1. Con los permisos del sistema anfitrión se debe definir la estructura de accesos.

En la entrada 4 se detectan los siguientes perfiles de usuarios y los siguientes tipos de acceso:

1-Administrador: Acceso general a la aplicación en modo escritura y lectura.

2-Profesores: Acceso a la interfaz "Alta de Alumnos" Lectura.

3-Alumnos: Acceso a la interfaz "Alta de Alumnos" Lectura y Escritura.

En la figura 5.14 (Estructura de Permisos de Acceso) se detalla el documento con la estructura de permisos.

Documento de Permisos de Servicio				
Fecha	2013-04-02		Responsable	Pedro Sánchez
Nombre de Sistema	Sistema de Inscripción			
Nombre de Interfaz	Alta de Alumnos		Nombre Clave	SI_01_AA
Definir para cada Perfil				
Perfil	Tipo de Acceso.			
Administrador	Lectura	Si	Escritura	Si
Profesor	Lectura	Si	Escritura	No
Alumno	Lectura	Si	Escritura	Si
Observaciones				
Firma				
Documento Versión	0.0.1			

Figura 5.14. Estructura de Permisos de Acceso.

Paso 2: Cargar Información para el Servicio.

2.1 A través de las pantallas de configuración del sistema cargar las ULC y PNR.

El arquitecto Juan Fernández será el encargado de dar de alta, se toma como base el documento obtenido en la F1-T2-R, F1-T3-R, F1-T4-R y la estructura detallada en la figura 5.14 (Estructura de Permisos de Acceso).

Paso 3: Generación deliberable de la tarea

Juan Fernández genera el reporte final de configuración de servicio.

Salidas: Reporte de Configuración + Servicio Configurado (**F3-T7-R**)

En la figura 5.15 (Deliberable de la etapa Configuración del Servicio) se detalla el deliberable final de esta Fase.

Documento de Configuración de Servicio			
Fecha	2013-04-02	Responsable	Pedro Sánchez
Nombre de Sistema	Sistema de Inscripción		
Nombre de Interfaz	Alta de Alumnos	Nombre Clave	SI_01_AA
Seguridad	Hereda la estructura del sistema anfitrión.		
ULC	Especificada en el documento F1-T2-R.		
PNR	Especificada en el documento F1-T3-R.		
Proxy	Especificada en el documento F1-T4-R. – No Utiliza.		
Observaciones			
El servicio ha sido configurado satisfactoriamente.			
Firma			
Documento Versión	0.0.1		

Figura 5.15. Deliberable de la etapa Configuración del Servicio.

5.1.2.3.2. Evaluación del Método de Inclusión (F3-T8-T)

Como guía para cumplimentar esta tarea se seguirán los pasos definidos en la tabla 4.13 (Evaluación de Métodos de Inclusión); para esta tarea se detallan las entradas, técnicas y salidas en la tabla 4.5 (Fases del Proceso) y el flujo del proceso se detalla en la figura 4.1 (Flujo de Proceso).

Entrada 1: Sistema.

Del extracto del documento de relevamiento (cuadro 5.1) se detalla la interfaz a evaluar *"…Figura. Fragmento de la aplicación: Interfaz de Usuario…"*.

Entrada 2: Diseño de ULC.

Se toma el documento F1-T2-R.

Entrada 3: Documentación de Sistema.

Se toma el texto redactado en el cuadro 5.1 para su análisis.

Desarrollo de Pasos:

Paso 1: Analizar la documentación del sistema.

El programador Pedro Sánchez es el encargado de realizar el análisis de la documentación y validar la aplicación anfitriona.

1.1 En este paso se debe evaluar cual fue el modelo seleccionado para la construcción del sistema anfitrión.

Del documento de relevamiento (cuadro 5.1) se extrae el siguiente texto *"…se ha utilizado HTML5 y Java Script, siendo Ajax con mensajería Json…".* Se detecta que el método de comunicación de la aplicación es HTML con AJAX.

1.2 Verificar que la documentación este alineada con la versión de producción del sistema anfitrión.

Del extracto del documento de relevamiento (cuadro 5.1) se detalla la interfaz a evaluar *"…Figura. Fragmento de la aplicación: Interfaz de Usuario…*".Del análisis realizado de la aplicación por el programador Pedro Sánchez se confirma que la aplicación utiliza HTML y AJAX.

Paso 2: Definir alternativas para la inclusión

2.1 Definir el conjunto de alternativas para la inclusión del servicio en función de la tecnología utilizada en la aplicación anfitriona.

Se definen dos alternativas en fucnión del estilo de comunicación que tiene la aplicación anfitriona:

1. Utilizar un modelo de mensajería basado en XML.
2. Utilizar un modelo de mensajería basado en JSon.

Como decisión final el programador Pedro Sánchez decide que la segunda opción es más apropiada por utilizar una menor longitud de mensaje para

enviar un mismo conjunto de información en comparación al la solución de XML.

Paso 3: Generación deliberable de la tarea

3.1 Generar Reporte con la mensajería del Servicio.

Pedro Sánchez genera el reporte final de la Evaluación del Método de Inclusión.

Salidas: Método de Inclusión (**F3-T8-R**)

En la figura 5.16 (Deliberable de la etapa Método de Inclusión) se detalla el deliberable final de esta Fase.

Documento metodológica de inclusión			
Fecha	2012-04-05	Responsable	Pedro Sánchez
Nombre de Sistema	Sistema de Inscripción		
Opción Seleccionada	Ajax – Json – Proxy.		
Observaciones			
El modelo de comunicación seleccionado entre la aplicación el servicio es Json por ser un cuasi-estándar de las comunicaciones de internet y por contar con desarrolladores que lo conocen en profundad. Para agregar las llamadas dentro de la aplicación anfitriona se utilizara Ajax por ser el modelo natural de trabajo junto a Json y puesto que la aplicación anfitriona ya trabaja con la Ajax no se vislumbran problemas en el proceso de integración.			
Firma			
Documento Versión	0.0.1		

Figura 5.16. Deliberable de la etapa Método de Inclusión.

5.1.2.3.2. Inclusión del Servicio en la Aplicación (F3-T9-T)

Como guía para cumplimentar esta tarea se seguirán los pasos definidos en la tabla 4.14 (Inclusión del Servicio en la Aplicación); para esta tarea se detallan las entradas, técnicas y salidas en la tabla 4.5 (Fases del Proceso) y el flujo del proceso se detalla en la figura 4.1 (Flujo de Proceso).

Entrada 1: Sistema.

Del extracto del documento de relevamiento (cuadro 5.1) se detalla la interfaz a evaluar *"…Figura. Fragmento de la aplicación: Interfaz de Usuario…"*.

Entrada 2: Deliberable de la etapa Método de Inclusión.

Se toma el documento F3-T8-R, figura 5.16 Deliberable de la etapa Método de Inclusión.

Desarrollo de Pasos:

Paso 1: Definición de la documentación requerida.

1.1 Definir el modelo de mensajería a utilizar.

Se asigna al programador Reinaldo Álvarez para la definición de la mensajería que se manejará.

Este provee el siguiente conjunto de mensajes:

Mensaje inicio de sesión:

De la Aplicación al Servicio:

Mensaje: alfanumérico de 5.

Token de Aplicación: alfanumérico de 20.

Usuario: alfanumérico de 10.

Del Servicio a la Aplicación:

Aceptación: booleano

Mensaje envío de datos:

De la Aplicación al Servicio:

Mensaje: alfanumérico de 5.

Token de Aplicación: alfanumérico de 20.

Usuario: alfanumérico de 10.

Interfaz: alfanumérico de 10.

Dato: alfanumérico variable (max. 512)

Del Servicio a la Aplicación:

Aceptación: booleano

Mensaje pedido de datos:

De la Aplicación al Servicio:

Mensaje: alfanumérico de 5.

Token de Aplicación: alfanumérico de 20.

Usuario: alfanumérico de 10.

Interfaz: alfanumérico de 10.

Del Servicio a la Aplicación:

Profundidad de Cola: numérico

Cantidad de Datos Enviados: numérico

Página: numérico

Dato: alfanumérico variable (max. 512)

Mensaje más datos:

De la Aplicación al Servicio:

Mensaje: alfanumérico de 5.

Token de Aplicación: alfanumérico de 20.

Usuario: alfanumérico de 10.

Interfaz: alfanumérico de 10.

Página: numérico

Del Servicio a la Aplicación:

Profundidad de Cola: numérico

Cantidad de Datos Enviados: numérico

Página: numérico

Dato: alfanumérico variable (max. 512)

Mensaje cierre de sesión:

De la Aplicación al Servicio:

Mensaje: alfanumérico de 5.

Token de Aplicación: alfanumérico de 20.

Usuario: alfanumérico de 10.

Del Servicio a la Aplicación:

Aceptación: booleano

1.2 Generación del plan de tiempos.

Se asigna al arquitecto Juan Fernández para la definición de los tiempos que insumirá la inclusión de la mensajería en la aplicación anfitriona.

El arquitecto estima que el tiempo para la inserción de la mensajería es de 5 días, con una dedicación completa por parte del programador encargado del trabajo.

Paso 2: Proceder a la inclusión según el plan definido anteriormente.

El arquitecto Juan Fernández designa a Reinaldo Álvarez para la inclusión de la mensajería en la aplicación.

2.1 Proceder al proceso de programación.

El programador Reinaldo Álvarez procede a la inclusión de la mensajería, esta es finalizada según el cronograma antes expuesto.

Paso 3: Generación deliberable de la tarea

3.1. Esta Tarea genera dos Deliberables el reporte y el servicio configurado.

El programador Reinaldo Álvarez procede a generar el deliberable de la etapa.

Salidas: Inclusión del Servicio

En la figura 5.17 (Deliberable de la etapa Inclusión del Servicio) se detalla el deliberable final de esta Fase.

5.1.2.4. Fase Prueba de Servicio (F4)

En esta sección se presentan los resultados de las tareas para completar esta fase de pruebas de la aplicación, estas son: Prueba de Usabilidad del Servicio (sección 5.1.2.4.1), Prueba de Carga del Servicio (sección 5.1.2.4.2) y Evaluación (sección 5.1.2.4.3).

5.1.2.4.1. Prueba de Usabilidad del Servicio (F4-T10-T)

Como guía para cumplimentar esta tarea se seguirán los pasos definidos en la tabla 4.15 (Prueba de Usabilidad del Servicio); para esta tarea se detallan las entradas, técnicas y salidas en la tabla 4.5 (Fases del Proceso) y el flujo del proceso se detalla en la figura 4.1 (Flujo de Proceso).

Entrada 1: Sistema + Invocación al Servicio.

Del extracto del documento de relevamiento (cuadro 5.1) se detalla la interfaz a evaluar *"…Figura. Fragmento de la aplcaición: Interfaz de Usuario…"*. A esta interfaz se le ha incluido la invocación al servicio.

Desarrollo de Pasos:

Paso 1: Repetir el procedimiento definido en F1-T1-T.

El arquitecto Juan Fernández procede con la recolección de los documentos generados en F1-T1-T.

1.1. Evaluar la aplicación interfaz por interfaz.

El arquitecto Juan Fernández asigna la tarea al mismo equipo de trabajo definido en la F1-T1, en las figuras 5.18 y 5.19 se detalla el resultado obtenido por los evaluadores.

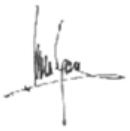
Documento de Inclusión del Servicio			
Fecha	2013-04-20	Responsable	Reinaldo Álvarez
Nombre de Sistema	Sistema Administración Curricular		
Nombre de Interfaz	Alta de profesores tutores	Nombre Clave	I01-AL-PT
Definir para cada Mensaje			
Mensaje	APL. ->SRV.	SRV. ->APL.	
Inicio de sesión	Mensaje: alfanumérico de 5. Token: alfanumérico de 20. Usuario: alfanumérico de 10.	Aceptación: booleano	
Envío de datos	Mensaje: alfanumérico de 5. Token: alfanumérico de 20. Usuario: alfanumérico de 10. Interfaz: alfanumérico de 10. Dato: alfanumérico variable (max. 512)	Aceptación: booleano	
Pedido de datos	Mensaje: alfanumérico de 5. Token: alfanumérico de 20. Usuario: alfanumérico de 10. Interfaz: alfanumérico de 10.	Profundidad: numérico Cantidad de Datos: numérico Página: numérico Dato: alfanumérico variable (max. 512)	
Más datos	Mensaje: alfanumérico de 5. Token: alfanumérico de 20. Usuario: alfanumérico de 10. Interfaz: alfanumérico de 10. Página: numérico	Profundidad: numérico Cantidad de Datos: numérico Página: numérico Dato: alfanumérico variable (max. 512)	
Cierre de sesión	Mensaje: alfanumérico de 5. Token: alfanumérico de 20. Usuario: alfanumérico de 10.	Aceptación: booleano	
Observaciones			
Mensajería Json.			
Firma			
Documento Versión	0.0.1		

Figura 5.17. Deliberable de la etapa Inclusión del Servicio.

Paso 2: Unificar opiniones de los evaluadores.

2.1. Se procederá a generar un documento unificado con el resumen de los distintos evaluadores.

El arquitecto Juan Fernández procede a unificar los documentos.

Salida: Evaluación de Usabilidad (**F4-T10-R**).

En la figura 5.20 (Evaluación Final del Proceso de Inspección Heurística) se presenta el documento final, que representa la evaluación de usabilidad para la interfaz "Alta de Alumnos".

Documento de Evaluación de Interfaz			
Fecha	2013-07-05	Responsable	Juan Pérez
Nombre de Sistema	Sistema de Inscripción		
Nombre de Interfaz	Alta de Alumnos	Nombre Clave	SI_01_AA
Descripción General			
Esta interfaz es utilizada para la carga, baja o modificaciones de alumnos.			
Puntos a Evaluar		Valor (Rango de 1-5) 1=Malo - 5=Excelente	
1.	Los cuadros de diálogos son simples y naturales.	3	
2.	La aplicación habla el lenguaje del usuario.	1	
3.	Se minimiza la carga de memoria en el usuario.	4	
4.	Hay consistencia en los datos pedido por la interfaz.	4	
5.	La interfaz proporciona información de proceso.	2	
6.	La interfaz proporciona métodos de salida.	2	
7.	La interfaz proporciona caminos abreviados.	3	
8.	La interfaz proporciona mensajes de error.	4	
9.	La aplicación previene errores.	2	
Firma			
Documento Versión	0.0.1		

Figura 5.18. Evaluación del Proceso de Inspección Heurística Juan Pérez.

Documento de Evaluación de Interfaz			
Fecha	2013-07-05	Responsable	Pedro Sánchez
Nombre de Sistema	Sistema de Inscripción		
Nombre de Interfaz	Alta de Alumnos	Nombre Clave	SI_01_AA
Descripción General			
Esta interfaz es utilizada para la carga, baja o modificaciones de alumnos.			
Puntos a Evaluar		Valor (Rango de 1-5) 1=Malo - 5=Excelente	
1.	Los cuadros de diálogos son simples y naturales.	4	
2.	La aplicación habla el lenguaje del usuario.	1	
3.	Se minimiza la carga de memoria en el usuario.	5	
4.	Hay consistencia en los datos pedido por la interfaz.	3	
5.	La interfaz proporciona información de proceso.	2	
6.	La interfaz proporciona métodos de salida.	2	
7.	La interfaz proporciona caminos abreviados.	3	
8.	La interfaz proporciona mensajes de error.	3	
9.	La aplicación previene errores.	3	
Firma			
Documento Versión	0.0.1		

Figura 5.19. Evaluación del Proceso de Inspección Heurística Pedro Sánchez.

5.1.2.4.2. Prueba de Carga del Servicio (F4-T11-T)

Como guía para cumplimentar esta tarea se seguirán los pasos definidos en la tabla 4.16 (Prueba de Carga del Servicio); para esta tarea se detallan las entradas, técnicas y salidas en la tabla 4.5 y el flujo del proceso se detalla en la figura 4.1.

Entrada 1: Sistema + Invocación al Servicio.

Del extracto del documento de relevamiento (cuadro 5.1) se detalla la interfaz a evaluar *"…Figura. Fragmento de la aplicación: Interfaz de Usuario…"*. A esta interfaz se le ha incluido la invocación al servicio.

Desarrollo de Pasos:

Paso 1: Tomar el programa construido para la ejecución de la prueba en la F2-T6-T Paso 2.

El arquitecto Juan Fernández procede a recolectar la documentación generada en la fase y tarea especificada. Además asigna al mismo equipo encargado de las pruebas para realizarlas. Pedro Sánchez procese a recuperar el programa generado para la prueba de carga.

Resumen de Evaluación de Interfaz				
Fecha	2013-07-05		Auditor Líder	Juan Fernández
Nombre de Sistema	Sistema de Inscripción			
Nombre de Interfaz	Alta de Alumnos		Nombre Clave	SI_01_AA
Descripción General				
Esta interfaz es utilizada para la carga, baja o modificaciones de alumnos.				
Puntos a Evaluar		Valor (Rango de 1-5) 1=Malo - 5=Excelente		
1.	Los cuadros de diálogos son simples y naturales.	3.5		
2.	La aplicación habla el lenguaje del usuario.	1		
3.	Se minimiza la carga de memoria en el usuario.	4.5		
4.	Hay consistencia en los datos pedido por la interfaz.	3.5		
5.	La interfaz proporciona información de proceso.	2		
6.	La interfaz proporciona métodos de salida.	2		
7.	La interfaz proporciona caminos abreviados.	3		
8.	La interfaz proporciona mensajes de error.	3.5		
9.	La aplicación previene errores.	2.5		
Firma				
Documento Versión	0.0.2			

Figura 5.20. Evaluación Final del Proceso de Inspección Heurística.

Paso 2: Preparación del material y selección del equipo evaluador

2.1. Selección del experto que evaluará el sistema.

Se selecciona al programador Pedro Sánchez para ejecutar la prueba de carga.

2.2. Disponer de los equipos necesarios para realizar la prueba.

Del documento de relevamiento (cuadro 5.1) se extrae el siguiente texto *"…Las computadoras para cada uno de los integrantes han sido asignadas y todos cuentan con los permisos de acceso necesarios..."*. Como se ha definido se cuenta con los accesos necesarios.

Paso 3: Ejecución de la prueba

3.1. Realizar una prueba de validación.

Pedro Sánchez ejecuta la prueba. En la figuras 5.21 se muestra el detalle de la ejecución.

Paso 4: Generación del reporte de prueba de carga

4.1. Se procederá a generar un documento unificado con el resumen de la prueba, el cual será el documento deliberable de la etapa.

Pedro Sánchez genera el reporte final de la prueba de ejecución.

Salida: Reporte de Carga (**F4-T11-R**).

En la figura 5.22 (Deliberable de la etapa Prueba de Carga de la Aplicación con Servicio) se detalla el documento final como deliberable de la presente tarea.

Documento de Prueba de Carga					
Fecha	2013-07-27	Hora	17:30	Responsable	Pedro Sánchez
Nombre de Sistema		Sistema de Inscripción			
Nombre de Interfaz		Alta de Alumnos		Nombre Clave	SI_01_AA
Concurrencia Máxima		100			
Tiempo Esperado		Menos de 5 segundos		Tiempo Obtenido	3.9
Observaciones					
Esta interfaz es utilizada para la carga, baja o modificaciones de alumnos.					
Repetir para todas las interfaces					
Firma					
Documento Versión		0.0.1			

Figura 5.21. Prueba de Carga del Sistema con la Invocación al Servicio.

Documento de Prueba de Carga Final					
Fecha	2013-07-27	Hora	18:00	Responsable	Pedro Sánchez
Nombre de Sistema		Sistema de Inscripción			
Nombre de Interfaz		Alta de Alumnos		Nombre Clave	SI_01_AA
Concurrencia Máxima		100			
Tiempo Esperado		Menos de 5 segundos		Tiempo Obtenido	3.9
Observaciones					
Esta interfaz es utilizada para la carga, baja o modificaciones de alumnos.					
Repetir para todas las interfaces					
Firma					
Documento Versión		0.0.1			

Figura 5.22. Deliberable de la etapa Prueba de Carga de la Aplicación con Servicio.

5.1.2.4.3. Evaluación (F4-T12-T)

Como guía para cumplimentar esta tarea se seguirán los pasos definidos en la tabla 4.14 (Técnica de Evaluación); para esta tarea se detallan las entradas, técnicas y salidas en la tabla 4.5 (Fases del Proceso) y el flujo del proceso se detalla en la figura 4.1 (Flujo de Proceso).

Entrada 1: F2-T5-R, figura 5.3 Evaluación Final del Proceso de Inspección Heurística.
Entrada 2: F2-T6-R, figura 5.13 Deliberable de la etapa Prueba de Carga de la Aplicación.
Entrada 3: F4-T10-R, figura 5.20 Evaluación Final del Proceso de Inspección Heurística.
Entrada 4: F4-T11-R, figura 5.22 Deliberable de la etapa Prueba de Carga de la Aplicación con Servicio.

Desarrollo de Pasos:

Paso 1: Comparar los Deliberables F2-T5-R y F4-T10-R.

El arquitecto Juan Fernández procede a realizar la comparación de los diversos deliberables y concluye el ambos son similares.

Paso 2: Comparar los Deliberables F2-T6-R y F4-T11-R.

El arquitecto Juan Fernández procede a realizar la comparación de los diversos deliberables y concluye el ambos tiene tiempos similares.

Paso 3: Evaluación final

El arquitecto Juan Fernández procede a realizar la comparación final, al ser satisfactoria las comparaciones hechas en los pasos uno y dos se da por concluida en forma satisfactoria la inclusión del servicio.

Paso 4: Generación de los deliberables de la tarea.

4.1. Este deliberable es la aceptación del proceso de inclusión.

El arquitecto Juan Fernández genera el reporte final.

Salida: Reporte de Carga (**F4-T12-R**).

En la figura 5.23 (Deliberable final de la inclusión del Servicio) se detalla el documento final como deliberable del proceso.

Documento de evaluación final			
Fecha	2013-07-22	Responsable	Juan Fernández
Nombre de Sistema	Sistema de Inscripción		
Recomendación Final	Aceptado		
Observaciones			
De la comparación de las pruebas realizadas no se observan cambios significantes antes y después de la inclusión de la funcionalidad de Undo/Redo. En consecuencia se ha logrado el objetivo de agregar la funcionalidad sin modificar la performance y la usabilidad existente del sistema anfitrión.			
Firma			
Documento Versión	0.0.1		

Figura 5.23. Deliberable final de la inclusión del Servicio.

5.2. CASO 2: APLICACIÓN MÓVIL

El motivo de la elección de este caso es representar la situación habitual que se presenta en las empresas donde se desea extender un software ya existente con una nueva funcionalidad o que las funcionalidades ya existentes puedan ser accedidas desde otros dispositivos electrónicos, en este caso teléfonos celulares inteligentes y tabletas electrónicas. En esta sección se detalla: descripción de la aplicación que se desea extender (Sección 5.2.1.) y la descripción y justificación de los pasos a seguir para la extensión a plataformas móviles (Sección 5.2.2).

5.2.1. Aplicación Móvil

En esta sección se presentan en el Cuadro 5.2 como parte de los nuevos requerimientos que se han producido en la aplicación "Sistema de Inscripción" (cuadro 5.1), se detalla partes del documento de relevamiento con las necesidades de la organización pertinentes al nuevo desarrollo.

"… Del uso de la aplicación administración universitaria ha surgido la necesidad que la misma pueda ser accedida desde diversos dispositivos electrónicos, se desea que la aplicación pueda ser accedida desde teléfonos celulares inteligentes con capacidades para navegar en internet y tabletas electrónicas…"
"… En función de estos requerimientos el equipo de desarrollo define que la aplicación contará con dos tipos de interfaz de usuarios desarrolladas en paralelo, la actual para, equipos de escritorio, y una nueva que será utilizada al reconocer que el punto de acceso es un dispositivo móvil, para la nueva interfaz se utilizará HTML5 y Java Script, Ajax con mensajería JSon…"
"… En función del éxito que ha sido la inclusión de un servicio en la aplicación en cuestión, como agregado de valor a la modificación que se está realizando se sugiere que se realicen los cambios necesarios para que la aplicación pueda interactuar con otros servicios existentes en el futuro próximo…"

Cuadro 5.2.a. Partes del documento con los nuevos requisitos para extender la aplicación.

"… Se encarga al mismo equipo que ha agregado la funcionalidad de UNDO/REDO la extensión de esta a la nueva interfaz, el equipo queda conformado de la siguiente forma, usuario del sistema en la actualidad llamado Juan Pérez, un experto en usabilidad Pedro Sánchez, Juan Fernández arquitecto de sistemas encargado de liderar al equipo encargado de incluir la funcionalidad de Undo/Redo y Reinaldo Álvarez programador…"
"… Las computadoras para cada uno de los integrantes han sido asignadas y todos cuentan con los permisos de acceso necesarios, así como con el software para realizar su trabajo según el perfil definido…"
"… En la figura que se detalla a continuación se observa el boceto de la interfaz para dispositivos móviles que se ha construido…"

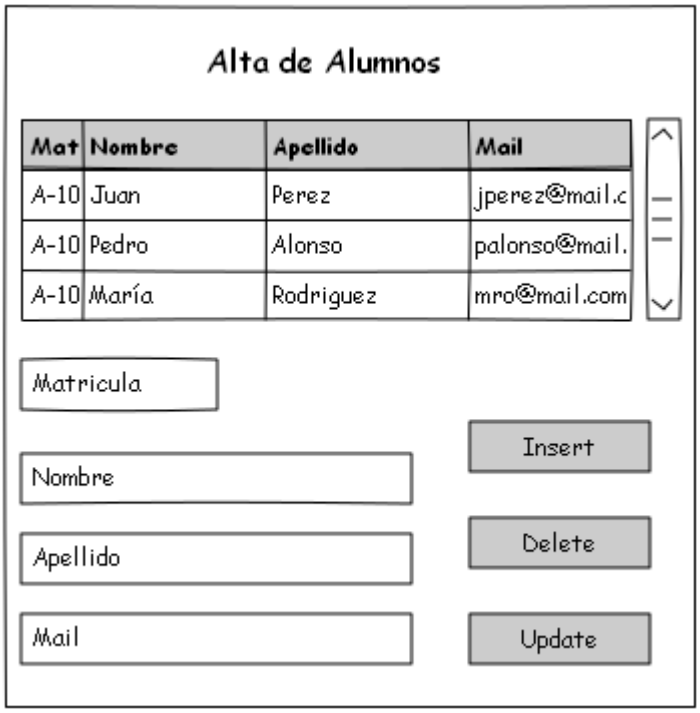


Figura. Boceto de la Interfaz Móvil.

Cuadro 5.2.b. Partes del documento con los nuevos requisitos para extender la aplicación.

5.2.2. Descripción y justificación de los pasos a seguir

Consideraciones sobre las faces del proceso a seguir para la actualización de la aplicación donde ya fue implementado el proceso de inclusión de la funcionalidad de UNDO/REDO. En esta sección, con base en la fases y tareas del propceso propuesto (tabla 4.4), se detalla para cada una de las fases y tareas si deben ser realizadas o no y la justificación de esta decisión.

5.2.2.1. Fase Modelado del Servicio (F1):

Tarea Análisis de Usabilidad (F1-T1):

Del extracto del documento de nuevos requerimientos (cuadro 5.2) se extrae el siguiente texto "…*la aplicación deba pueda ser accedida desde teléfonos celulares inteligentes con capacidades para navegar en internet y tabletas electrónicas…la*

aplicación contará con dos tipos de interfaz de usuarios…se detalla a continuación se observa el boceto de la interfaz…".

Sobre el extracto del cuadro 5.2 para la figura: Boceto de Interfaz Móvil, se puede observar que la misma es solo una adaptación al modelo definido para la página Web con lo cual se toma el documento Evaluación Final del Proceso de Inspección Heurística (Figura 5.3.) como deliberable de esta etapa.

Tarea Detección de Unidades Lógicas de Cambio (F1-T2):

Por lo expuesto en F1-T1 (Tarea Análisis de Usabilidad) se dispone utilizar el documento Evaluación Final del Arquitecto sobre la interfaz "Alta de Alumnos" (Figura 5.5.).

Tarea Detección de Puntos de No Retorno (F1-T3):

Por lo expuesto en F1-T1 (Tarea Análisis de Usabilidad) y F1-T2 (Tarea Detección de Unidades Lógicas de Cambio), se dispone utilizar el documento Deliberable de la etapa de Detección de PNR (Figura 5.8.).

Tarea Estudio de Viabilidad de Servidor de Proximidad (F1-T4):

Del extracto del documento de nuevos requerimientos (cuadro 5.2) se extrae el siguiente texto "…*la aplicación deba pueda ser accedida desde teléfonos celulares inteligentes con capacidades para navegar en internet y tabletas electrónicas…del éxito que ha sido la inclusión de un servicio en la aplicación en cuestión como agregado de valor a la modificación que se está realizando se sugiere que se hagan los cambios necesarios para que la aplicación pueda interactuar con otros servicios existentes… ".*

El texto extraído implica evaluar si es necesario la utilización de un servidor de proximidad, por esto esta tarea debe ser realizada nuevamente.

5.2.2.2. Fase Pruebas de la Aplicación (F2):

Tarea Prueba de Usabilidad de la Aplicación (F2-T5):

Por lo expuesto en F1-T1 (Tarea Análisis de Usabilidad) se toma el documento Evaluación Final del Proceso de Inspección Heurística (Figura 5.3.) como deliberable de esta etapa.

Tarea Prueba de Carga de la Aplicación (F2-T6):

La prueba de carga de aplicación antes de la inclusión de la funcionalidad de UNDO/REDO, no es necesario pues esta es una funcionalidad nueva.

5.2.2.3. Fase Creación del Servicio (F3):

Tarea Especificación de la Configuración de Servicio Prueba (F3-T7):

En función del documento Deliberable de la etapa Configuración del Servicio (Figura 5.15) junto al extracto de los nuevos requerimientos (cuadro 5.2) no es necesario realizar ninguna modificación al servicio que se encuentra en producción.

Tarea Implementación de la Invocación del Servicio (F3-T8):

Esta tarea debe ser realizada nuevamente, para evaluar las implicancias de la nueva interfaz en la aplicación.

Tarea Implementación de la Inclusión del Servicio (F3-T9):

Esta tarea debe ser realizada nuevamente para definir de ser necesario un nuevo conjunto de mensajes.

5.2.2.4. Fase Pruebas del Servicio (F4):

Tarea Prueba de Usabilidad del Servicio (F4-T10):

Esta tarea debe ser realizada nuevamente para evaluar la usabilidad de la nueva interfaz luego de la inclusión del servicio.

Tarea Prueba de Carga del Servicio (F4-T11):

Esta tarea debe ser realizada nuevamente para evaluar si luego de la inclusión del servicio se siguen con los parámetros de tiempo de respuesta y concurrencia definidos en el cuadro 5.1.

Tarea Evaluación (F4-T12):

Esta tarea debe ser realizada para obtener el cierre del proceso.

A modo de conclusión se presenta la tabla 5.4 donde se resume por fase y tarea si debe ser aplicada.

Fase	Tareas	Aplica	Observación
Modelado del Servicio (F1)	Análisis de Usabilidad (F1-T1)	Si	Se utiliza el documento Evaluación Final del Proceso de Inspección Heurística (Figura 5.3.)
	Detección de Unidades Lógicas de Cambio (F1-T2)	Si	Se utiliza el documento Evaluación Final del Arquitecto sobre la interfaz "Alta de Alumnos" (Figura 5.5.)
	Detección de Puntos de No Retorno (F1-T3)	Si	Se utiliza el utilizar el documento Deliberable de la etapa de Detección de PNR (Figura 5.8.).
	Estudio de Viabilidad de Servidor de Proximidad (F1-T4)	Si	
Pruebas de la Aplicación (F2)	Prueba de Usabilidad de la Aplicación (F2-T5)	Si	Se utiliza el documento Evaluación Final del Proceso de Inspección Heurística (Figura 5.3.)
	Prueba de Carga de la Aplicación (F2-T6)	No	
Creación del Servicio (F3)	Especificación de la Configuración de Servicio (F3-T7)	No	
	Implementación de la Invocación del Servicio (F3-T8)	Si	
	Implementación de la Inclusión del Servicio (F3-T9)	Si	
Pruebas del Servicio (F4)	Implementación de la Inclusión del Servicio (F3)	Si	
	Prueba de Carga del Servicio (F4-T11)	Si	
	Evaluación (F4-T12)	Si	

Tabla 5.4. Fases y Tareas del Proceso de Inclusión para la aplicación móvil.

5.2.3. Proceso de Actualización del Servicio en la Aplicación

En esta sección una vez hecha la evaluación de los pasos a seguir se procede a cumplimentar los pasos para adaptar el modelo de servicio a la nueva interfaz, donde se detallan: Estudio de Viabilidad de Servidor de Proximidad (5.2.3.1), Inclusión del Servicio en la Aplicación (5.2.3.2), Inclusión del Servicio en la Aplicación (5.2.3.3), Prueba de Usabilidad del Servicio (5.2.3.4) Prueba de Carga del Servicio (5.2.3.5) y Evaluación (5.2.3.6).

5.2.3.1. Estudio de Viabilidad de Servidor de Proximidad (F1-T4-T)

Del documento de relevamiento (cuadro 5.2) se extrae el siguiente texto "*…Juan Fernández es el arquitecto de sistemas...".* El arquitecto Juan Fernández será el evaluador responsable de la inclusión de la funcionalidad de UNDO/REDO en la aplicación.

El arquitecto define, por lo detallado en 5.2.2.1, que es necesario realizar un nuevo cálculo de viabilidad del servidor de proximidad, se detallan los pasos realizados a continuación.

Entrada 1: Requerimiento de Sistema.

> Del extracto del documento de nuevos requerimientos (cuadro 5.2) se extrae el siguiente texto "*…la aplicación pueda ser accedida desde teléfonos celulares inteligentes con capacidades para navegar en internet y tabletas electrónicas…del éxito que ha sido la inclusión de un servicio en la aplicación en cuestión como agregado de valor a la modificación que se está realizando se sugiere que se realicen los cambios necesarios para que la aplicación pueda interactuar con otros servicios existentes…"*

Entrada 2: Sistema.

> Del extracto del documento de relevamiento (cuadro 5.2) se detalla la interfaz a evaluar "*…Figura. Bosquejo de la Interfaz Móvil…*".

Desarrollo de Pasos:

Paso 1: Cumplimentar el proceso definido como Evaluación de Viabilidad

> Como se mencionó anteriormente el arquitecto Juan Fernández será el encargado de llevar adelante el estudio de viabilidad. En la tabla 5.5 se presenta el resulto a las preguntas sugeridas.

ID	Pregunta asociada	Respuesta
1	Existen restricciones de seguridad	Si
2	Deben ser integrados varios servicios	Si
3	Existe la posibilidad de cambio en los proveedores de servicios	Existe
4	Se accederá a la aplicación desde múltiples dispositivos	Si
5	Existen cambios frecuentes en la aplicación	Algunos
6	Se desea contar con mejoras pos implementación	No

Tabla 5.5. Respuestas obtenidas.

Conversión a intervalos difusos:

En la tabla 5.6 se detallan la conversión a intervalos difusos, se ha tomado como referencia a la tabla 4.3.

Si	**10**
No	**0**

Tabla 5.6. Intervalos Difusos.

Definición de Pesos

En la tabla 5.7 se muestran los pesos luego del calculo de los intervalos difusos.

ID	Pregunta asociada	Peso
1	Existen restricciones de seguridad	10
2	Deben ser integrados varios servicios	10
3	Existe la posibilidad de cambio en los proveedores de servicios	8
4	Se accederá a la aplicación desde múltiples dispositivos	10
5	Existen cambios frecuentes en la aplicación	6
6	Se desea contar con mejoras pos implementación	0

Tabla 5.7. Cálculo de pesos.

Detalle de cada respuesta

1. Existen restricciones de seguridad: Del documento de relevamiento (cuadro 5.1) se extrae el siguiente texto *"…Existen tres tipos de perfiles de usuarios para la aplicación, una es el administrador con permiso general para acceder a todo el sistema, el perfil profesor que puede definir el curso que dictará, el día y el horario del mismo y si este necesita algún requerimiento especial para su dictado. Con esto se asigna automáticamente las aulas para los cursos. Con la grilla de cursos y horarios el tercer perfil, alumnos, que puede darse de alta e inscribe hasta el tope máximo que permite cada aula…"* En este extracto se detallan tres perfiles de usuarios distintos y permisos diferentes para cada uno, con lo cual se define que existen restricciones de seguridad.
2. Deben ser integrados varios servicios: En el análisis realizado a la documentación (cuadro 5.2) se extrae "*…del éxito que ha sido la inclusión de un servicio en la aplicación en cuestión como agregado de valor a la modificación que se está realizando se sugiere que se*

realicen los cambios necesarios para que la aplicación pueda interactuar con otros servicios existentes..", esto define la necesidad de contemplara la integración de otros servicios

3. Existe la posibilidad de cambio en los proveedores de servicios: Por lo detallado en el punto 2, se deduce que podrá haber cambio de proveedores de servicios.
4. Se accederá a la aplicación desde múltiples dispositivos: En el análisis realizado a la documentación (cuadro 5.2) "…*que la aplicación pueda ser accedida desde teléfonos celulares inteligentes con capacidades para navegar en internet y tabletas electrónicas...*" se hace referencia al tipo de dispositivo con lo cual concluye que se accederá a la aplicación desde otros dispositivos.
5. Existen cambios frecuentes en la aplicación: En el análisis realizado a la documentación (cuadro 5.2) no se detecta la existencia de cambios frecuentes en la aplicación pero el hecho de estar realizando este cambio implica que existe alguna posibilidad de cambio.
6. Se desea contar con mejoras pos implementación: En el análisis realizado a la documentación (cuadro 5.2) no se detecta la posibilidad de implementar mejoras pos implementación.

Cálculo de Promedio

La suma de los pesos es 44 esto se divide por 6, la cantidad de preguntas asociadas, como resultado se obtiene 7.33 se redondea a 7.

Como el valor obtenido es mayor a 5 es recomendable la inclusión de un servidor de proximidad.

Salida: Estudio de Viabilidad (**F2-T4-R**).

En la figura 5.24 (Deliberable de la etapa Estudio de Viabilidad) se presenta el documento final, que representa la evaluación del estudio de viabilidad.

Documento Estudio de Viabilidad			
Fecha	2013-08-06	Responsable	Juan Fernández
Nombre de Sistema	Sistema Administración Curricular		
Recomendación Final	Aceptado		
Observaciones			
Como el valor obtenido es mayor a 5 se recomienda la inclusión de un servidor de proximidad.			
Firma			
Documento Versión	0.0.1		

Figura 5.24. Deliberable de la etapa Estudio de Viabilidad.

5.2.3.2. Implementación de la Invocación del Servicio (F3-T8-T)

Como guía para cumplimentar esta tarea se seguirán los pasos definidos en la tabla 4.14 (Inclusión del Servicio en la Aplicación); para esta tarea se detallan las entradas, técnicas y salidas en la tabla 4.5 (Fases del Proceso) y el flujo del proceso se detalla en la figura 4.1 (Flujo de Proceso).

Del extracto del documento de nuevos requerimientos (cuadro 5.2) se extrae el siguiente texto "*…un experto en usabilidad Pedro Sánchez…".* El programador Pedro Sánchez es el encargado de realizar el análisis de la documentación y validar la aplicación anfitriona.

Éste considera, en función de lo expuesto en el cuadro 5.2 "*…se desea que la aplicación pueda ser accedida desde teléfonos celulares inteligentes con capacidades para navegar en internet y tabletas electrónicas…del éxito que ha sido la inclusión de un servicio en la aplicación en cuestión como agregado de valor a la modificación que se está realizando se sugiere que se realicen los cambios necesarios para que la aplicación pueda interactuar con otros servicios existentes…"* y del resultado obtenido en Tarea Análisis de uso de Servidor de Proximidad (F2-T4) se define rehacer esta tarea nuevamente actualizando a la nueva situación.

Entrada 1: Sistema.

Del extracto del documento de relevamiento (cuadro 5.1) se detalla la interfaz a evaluar *"…Figura. Fragmento de la aplicación: Interfaz de Usuario…*" y del extracto del cuadro 5.2 *"…Figura; Bosquejo de la Interfaz Móvil…"*.

Entrada 2: Diseño de ULC.

Se toma el documento F1-T2-R.

Entrada 3: Documentación de Sistema.

Se toma el texto redactado en el cuadro 5.1 y cuadro 5.2 para su análisis.

Desarrollo de Pasos:

Paso 1: Analizar la documentación del sistema.

1.1 En este paso se debe evaluar cual fue el modelo seleccionado para la construcción del sistema anfitrión.

Del documento de relevamiento (cuadro 5.1) se extrae el siguiente texto *"…se ha utilizado HTML5 y Java Script, siendo Ajax con mensajería Json…".* Se detecta que el método de comunicación de la aplicación es HTML con AJAX, en el cuadro 5.2 no se hace referencia a la tecnología utilizada con lo que se deduce que continua la definida en el cuadro 5.1.

1.2 Verificar que la documentación este alineada con la versión de producción del sistema anfitrión.

Del extracto del documento de relevamiento (cuadro 5.1) se detalla la interfaz a evaluar *"…Figura. Fragmento de la aplicación: Interfaz de Usuario…*". Del análisis realizado de la aplicación por el programador Pedro Sánchez se confirma que la aplicación utiliza HTML y AJAX, sobre la información detallada en el cuadro 5.2 no hay proceso de verificación pues se está desarrollando la misma.

Paso 2: Definir alternativas para la inclusión

2.1 Definir el conjunto de alternativas para la inclusión del servicio en función de la tecnología utilizada en la aplicación anfitriona.

Se toman las alternativas definidas para la inclusión de la aplicación por no detectarse cambios en la comunicación:

1. Utilizar un modelo de mensajería basado en XML.
2. Utilizar un modelo de mensajería basado en JSon.

Como decisión final el programador Pedro Sánchez toma las conclusiones definidas para la etapa de inclusión del servicio y utiliza Json como medio de comunicación.

Como agregado a esto el programador Pedro Sánchez define en función de lo detallado en el cuadro 5.2, que el servidor de proximidad será utilizado en esta etapa como un puente entre la aplicación el servicio, por no requerirse de conexión a otro servicio, se deja el modelo de comunicación ya definido. Por otra parte el servidor de proximidad reconocerá el dispositivo desde el cual se está comunicando el usuario y proveerá el conjunto de interfaces acorde a este dispositivo.

Paso 3: Generación deliberable de la tarea

3.1 Generar Reporte con la mensajería del Servicio.

Pedro Sánchez genera el reporte final de la Evaluación del Método de Inclusión.

Salidas: Método de Inclusión (**F3-T8-R**)

En la figura 5.25 (Deliberable de la etapa Método de Inclusión) se detalla el deliberable final de esta Fase.

5.2.3.3. Inclusión del Servicio en la Aplicación (F3-T9-T)

El arquitecto Juan Fernández designa a Reinaldo Álvarez para la inclusión de la mensajería en la aplicación. Este último define en función de lo detallado en la tarea Implementación de la Invocación del Servicio (F3-T8) que se debe utilizar el mismo conjunto de mensajes por no haberse realizado cambio alguno, en consecuencia el deliberable de esta etal es el documento Deliberable de la etapa Inclusión del Servicio (Figura 5.17).

5.2.3.4. Prueba de Usabilidad del Servicio (F4-T10-T)

Como guía para cumplimentar esta tarea se seguirán los pasos definidos en la tabla 4.15 (Prueba de Usabilidad del Servicio); para esta tarea se detallan las entradas, técnicas y salidas en la tabla 4.5 (Fases del Proceso) y el flujo del proceso se detalla en la figura 4.1 (Flujo de Proceso).

Documento metodológica de inclusión			
Fecha	2012-08-15	Responsable	Pedro Sánchez
Nombre de Sistema	Sistema de Inscripción		
Opción Seleccionada	Ajax – Json – Proxy.		
Observaciones			
El modelo de comunicación seleccionado entre la aplicación el servicio es Json por ser un cuasi-estándar de las comunicaciones de internet y por contar con desarrolladores que lo conocen en profundad. Para agregar las llamadas dentro de la aplicación anfitriona se utilizara Ajax por ser el modelo natural de trabajo junto a Json y puesto que la aplicación anfitriona ya trabaja con la Ajax no se vislumbran problemas en el proceso de integración. Este tipo de comunicación es apto tanto sea para aplicaciones Web como móviles. Se utilizará un servidor de proximidad para gestionar los dos tipos de interfaz que son necesarios administrar, este proxy en esta primera etapa solo será utilizado como un puente entre la interfaz y el servicio, pues al hacer solo un servicio no es necesario realizar cambios al mismo.			
Firma			
Documento Versión	0.0.1		

Figura 5.25. Deliberable de la etapa Método de Inclusión.

Entrada 1: Sistema + Invocación al Servicio.

Del extracto del documento de relevamiento (cuadro 5.2) se detalla la interfaz a evaluar "*…Figura. Bosquejo de la Interfaz Móvil…*". A esta interfaz se le ha incluido la invocación al servicio.

Desarrollo de Pasos:

Paso 1: Repetir el procedimiento definido en F1-T1-T.

El arquitecto Juan Fernandez procede con la recolección de los documentos generados en F1-T1-T.

1.1. Evaluar la aplicación interfaz por interfaz.

El arquitecto Juan Fernández asigna la tarea al mismo equipo de trabajo definido en la F1-T1, en las figuras 5.26 y 5.27 se detalla el resultado obtenido por los evaluadores.

Documento de Evaluación de Interfaz			
Fecha	2013-09-01	Responsable	Juan Pérez
Nombre de Sistema	Sistema de Inscripción		
Nombre de Interfaz	Alta de Alumnos	Nombre Clave	SI_01_AA
Descripción General			
Esta interfaz es utilizada para la carga, baja o modificaciones de alumnos.			

	Puntos a Evaluar	Valor (Rango de 1-5) 1=Malo - 5=Excelente
1.	Los cuadros de diálogos son simples y naturales.	3
2.	La aplicación habla el lenguaje del usuario.	1
3.	Se minimiza la carga de memoria en el usuario.	4
4.	Hay consistencia en los datos pedido por la interfaz.	4
5.	La interfaz proporciona información de proceso.	2
6.	La interfaz proporciona métodos de salida.	2
7.	La interfaz proporciona caminos abreviados.	3
8.	La interfaz proporciona mensajes de error.	4
9.	La aplicación previene errores.	2

Firma	
Documento Versión	0.0.1

Figura 5.26. Evaluación del Proceso de Inspección Heurística Juan Pérez.

Documento de Evaluación de Interfaz			
Fecha	2013-09-01	Responsable	Pedro Sánchez
Nombre de Sistema	Sistema de Inscripción		
Nombre de Interfaz	Alta de Alumnos	Nombre Clave	SI_01_AA
Descripción General			
Esta interfaz es utilizada para la carga, baja o modificaciones de alumnos.			

	Puntos a Evaluar	Valor (Rango de 1-5) 1=Malo - 5=Excelente
1.	Los cuadros de diálogos son simples y naturales.	4
2.	La aplicación habla el lenguaje del usuario.	1
3.	Se minimiza la carga de memoria en el usuario.	5
4.	Hay consistencia en los datos pedido por la interfaz.	3
5.	La interfaz proporciona información de proceso.	2
6.	La interfaz proporciona métodos de salida.	2
7.	La interfaz proporciona caminos abreviados.	3
8.	La interfaz proporciona mensajes de error.	3
9.	La aplicación previene errores.	3

Firma	
Documento Versión	0.0.1

Figura 5.27. Evaluación del Proceso de Inspección Heurística Pedro Sánchez.

Paso 2: Unificar opiniones de los evaluadores.

2.1. Se procederá a generar un documento unificado con el resumen de los distintos evaluadores.

El arquitecto Juan Fernández procede a unificar los documentos.

Salida: Evaluación de Usabilidad (**F4-T10-R**).

En la figura 5.28 (Evaluación Final del Proceso de Inspección Heurística) se presenta el documento final, que representa la evaluación de usabilidad para la interfaz “Alta de Alumnos Móvil”.

5.2.3.5. Prueba de Carga del Servicio (F4-T11-T)

Como guía para cumplimentar esta tarea se seguirán los pasos definidos en la tabla 4.16 (Prueba de Carga del Servicio); para esta tarea se detallan las entradas, técnicas y salidas en la tabla 4.5 y el flujo del proceso se detalla en la figura 4.1.

Entrada 1: Sistema + Invocación al Servicio.

Del extracto del documento de relevamiento (cuadro 5.2) se detalla la interfaz a evaluar *“…Figura. Bosquejo de la Interfaz Móvil…”*. A esta interfaz se le ha incluido la invocación al servicio.

Resumen de Evaluación de Interfaz				
Fecha	2013-09-01		Auditor Líder	Juan Fernández
Nombre de Sistema	Sistema de Inscripción			
Nombre de Interfaz	Alta de Alumnos		Nombre Clave	SI_01_AA
Descripción General				
Esta interfaz es utilizada para la carga, baja o modificaciones de alumnos.				
Puntos a Evaluar		Valor (Rango de 1-5) 1=Malo - 5=Excelente		
1.	Los cuadros de diálogos son simples y naturales.	3.5		
2.	La aplicación habla el lenguaje del usuario.	1		
3.	Se minimiza la carga de memoria en el usuario.	4.5		
4.	Hay consistencia en los datos pedido por la interfaz.	3.5		
5.	La interfaz proporciona información de proceso.	2		
6.	La interfaz proporciona métodos de salida.	2		
7.	La interfaz proporciona caminos abreviados.	3		
8.	La interfaz proporciona mensajes de error.	3.5		
9.	La aplicación previene errores.	2.5		
Firma				
Documento Versión	0.0.2			

Figura 5.28. Evaluación Final del Proceso de Inspección Heurística.

Desarrollo de Pasos:

Paso 1: Tomar el programa construido para la ejecución de la prueba en la F2-T6-T Paso 2.

El arquitecto Juan Fernández procede a recolectar la documentación generada en la fase y tarea especificada. Además asigna al mismo equipo encargado de las pruebas para realizarlas. Pedro Sánchez procede a recuperar el programa generado para la prueba de carga.

Paso 2: Preparación del material y selección del equipo evaluador

2.1. Selección del experto que evaluará el sistema.

Se selecciona al programador Pedro Sánchez para ejecutar la prueba de carga.

2.2. Disponer de los equipos necesarios para realizar la prueba.

Del documento de relevamiento (cuadro 5.1) se extrae el siguiente texto *"…Las computadoras para cada uno de los integrantes han sido asignadas y todos cuentan con los permisos de acceso necesarios…".* Como se ha definido se cuenta con los accesos necesarios.

Paso 3: Ejecución de la prueba

3.1. Realizar una prueba de validación.

Pedro Sánchez ejecuta la prueba.

Paso 4: Generación del reporte de prueba de stress

4.1. Se procederá a generar un documento unificado con el resumen de la prueba, el cual será el documento deliberable de la etapa.

Pedro Sánchez genera el reporte final de la prueba de ejecución.

Salida: Reporte de Carga (**F4-T11-R**).

En la figura 5.29 (Deliberable de la etapa Prueba de Carga de la Aplicación con Servicio) se detalla el documento final como deliberable de la presente tarea.

Documento de Prueba de Carga Final					
Fecha	2013-09-18	Hora	09:00	Responsable	Pedro Sánchez
Nombre de Sistema		Sistema de Inscripción			
Nombre de Interfaz		Alta de Alumnos		Nombre Clave	SI_01_AA
Concurrencia Máxima		100			
Tiempo Esperado		Menos de 5 segundos		Tiempo Obtenido	3.8
Observaciones					
Esta interfaz es utilizada para la carga, baja o modificaciones de alumnos.					
Repetir para todas las interfaces					
Firma					
Documento Versión		0.0.1			

Figura 5.29. Deliberable de la etapa Prueba de Carga de la Aplicación con Servicio.

5.2.3.6. Evaluación (F4-T12-T)

Como guía para cumplimentar esta tarea se seguirán los pasos definidos en la tabla 4.14 (Técnica de Evaluación); para esta tarea se detallan las entradas, técnicas y salidas en la tabla 4.5 (Fases del Proceso) y el flujo del proceso se detalla en la figura 4.1 (Flujo de Proceso).

Entrada 1: F1-T1-R, figura 5.3 Evaluación Final del Proceso de Inspección Heurística, por lo expuesto por Pedro Sánchez.

Entrada 2: F4-T10-R, figura 5.28 Evaluación Final del Proceso de Inspección Heurística.

Entrada 3: F4-T11-R, figura 5.29 Deliberable de la etapa Prueba de Carga de la Aplicación con Servicio.

Desarrollo de Pasos:

Paso 1: Comparar los Deliberables F2-T5-R y F4-T10-R.

El arquitecto Juan Fernández procede a realizar la comparación de los diversos deliberables y concluye el ambos son similares.

Paso 2: Comparar los Deliberables F2-T6-R y F4-T11-R.

El arquitecto Juan Fernández procede a realizar la comparación, aquí no se cuenta con el deliberable F2-T6-R, por se una aplicación nueva y no de una existente, se cuenta con el documento F4-T11.R el cual cumple con los tiempos y la concurrencia especificados en el cuadro 5.1.

Paso 3: Evaluación final

El arquitecto Juan Fernández procede a realizar la comparación final, al ser satisfactoria las comparaciones hechas en los pasos 1 y dos se da por finalizada en forma satisfactoria la inclusión del servicio.

Paso 4: Generación deliberables de la tarea.

4.1. Este deliberable es la aceptación del proceso de inclusión.

El arquitecto Juan Fernández genera el reporte final.

Salida: Reporte de Carga (**F4-T12-R**).

En la figura 5.30 (Deliberable final de la inclusión del Servicio) se detalla el documento final como deliberable del proceso.

Documento de evaluación final			
Fecha	2013-09-20	Responsable	Juan Fernández
Nombre de Sistema	Sistema de Inscripción		
Recomendación Final	Aceptado		
Observaciones			
De la comparación de las pruebas realizadas no se observan cambios significantes antes y después de la inclusión de la funcionalidad de UNDO/REDO. En consecuencia se ha logrado el objetivo de agregar la funcionalidad sin modificar la performance y la usabilidad existente del sistema anfitrión.			
Firma			
Documento Versión	0.0.1		

Figura 5.30. Deliberable final de la inclusión del Servicio.

5.3. CASO 3: APLICACIÓN EN CLOUD COMPUTING

El motivo de la selección de este caso es presentar la situación a migrar de plataforma la aplicación "Sistema de Inscripción", esto se debe a un avance en la tecnología y abaratamiento de los costos del denominado modelo de "Cloud Computing" o computación en la nube. Aquí se intenta migrar a un ambiente donde el proveedor implementa una arquitectura de plataforma como servicio, donde pueden ser ejecutadas aplicaciones software realizadas según un conjunto de especificaciones dadas por el proveedor. En esta sección se detalla: descripción de la aplicación que se desea migrar (Sección 5.3.1.) y la descripción y justificación de los pasos a seguir para la migración (Sección 5.3.2).

5.3.1. Aplicación en Cloud Computing

En esta sección se presentan en el Cuadro 5.3 como parte de los nuevos requerimientos que se han producido en la aplicación *"Sistema de Inscripción"* (cuadro 5.1 y cuadro 5.2), se detalla partes del documento de relevamiento con las necesidades y requisitos del nuevo desarrollo.

"… La evolución tecnológica hace que sea necesario la adaptación de los sistemas existentes; en tal sentido la masificación en uso de la denominada ´Cloud Computing´, permite a las organizaciones, en ciertas situaciones, reducir costos de mantenimiento de infraestructura y personal dedicado a estas, es por esto que la universidad que administra la aplicación "Sistema de Inscripción" ha decidido evaluar las diversas alternativas existentes en el mercado para tomar una decisión sobre la conveniencia o no de migrar su sistema a la nube…"

"…Para realizar la tarea se asigna al ingeniero Alberto García que será el encargado de evaluar las diversas alternativas existentes en el mercado…"

"…Como primer paso el ingeniero García contacta al arquitecto Juan Fernández a sabiendas que este recientemente ha incluido la funcionalidad de UNDO/REDO dentro de la aplicación a migrar, para solicitar su ayuda en el proceso de evaluación, además para que comparta su experiencia e intercambiar opiniones sobre la aplicación…"

"…Sobre el conjunto de posibilidades existentes en el mercado se ha resuelto evaluar las alternativas que proveen Amazon y Google, esta decisión ha sido tomada por la solidez y trayectoria de las empresas, a diferencia de otras empresas que han surgido en el último tiempo, de las cuales no se cuenta con un historial o referencias de las mismas. En el proceso de evaluación se define, dentro de las diversas alternativas que maneja cada empresa, comparará: (a) "Amazon Elastic Computer Cloud" (EC2) y (b) "Google App Engine" (GAE).

"…El arquitecto García da una breve introducción a cada uno de los servicios:

´ EC2 es un servicio web que proporciona capacidad informática con tamaño modificable en la nube. Está diseñado para facilitar a los desarrolladores recursos informáticos escalables basados en Web.´

´ GAE permite a los desarrolladores crear aplicaciones web escalable, GEA expone un entorno de desarrollo totalmente escalable ´..."

".. A su vez el ingeniero García da su opinión sobre los servicios evaluados, basado en la siguiente escala: (a) Tipo de Servicio, (b) Costos, (c) Soporte y (d) Mantenimiento.

Tipo de Servicio: EC2 debe ser considerado como IaaS (Infraestructura como Servicio), al contratar el servicio de Amazon se recibe una maquina virtual la cual puede operar como si fuera una maquina propia, se debe hacer notar que la misma está configurada con todas las herramientas necesarias para poder implementar la aplicación y que rápidamente este la misma en producción.

GAE debe ser considerada como PaaS (Plataforma como Servicio) pues aquí la puesta en producción de la aplicación se realiza a través de un mecanismo de trasferencia a los servidores y Google se encarga de la infraestructura subyacente, se provee de todas las herramientas para trabajar y poner en producción una aplicación, no es necesario saber nada sobre qué sistema operativo o donde esta ejecutándose el proceso.

Costos: Basado en el programa realizado por Reinaldo Álvarez para el análisis de los archivos de suceso (cuadro 5.1), se cuenta con el volumen de información transaccional que soporta el sistema al mes. En la evaluación de esta variable para ambas empresas se ha detectado la misma situación, la dificultad para predecir con exactitud los costos finales; por las simulaciones y contactos que se han mantenido con ambas empresas este sería alrededor de 100 dólares estadounidenses de forma mensual, existen planes de pago adelantado que reducen este costo en aproximadamente 10%.

Soporte: Ambas empresas proveen soporte mediante mail y línea telefónica, vale mencionar que este soporte es en inglés, la evaluación que se ha podido realizar de foros y blogs especializados es que el nivel de atención de ambos es aceptable para los casos habituales, no así para excepciones o casos poco comunes .

Mantenimiento: EC2 merece dos tipos de mantenimiento por un lado el propio de la aplicación y por otro el de la maquina virtual, este último aunque reducido es necesario que sea realizado. En GAE solo debe ser considerado el mantenimiento de la aplicación y no de la infraestructura…".

"…Se han realizado una serie de reuniones entre el ingeniero García y el arquitecto Fernández para evaluar las distintas alternativas y se ha generado el siguiente reporte:

´Del análisis realizado sobre las soluciones provistas por Amazon y Google se ha decidido utilizar la solución provista por Google el Google App Engine, esta decisión se basa en que solo se necesita trabajar con la aplicación y no preocuparse por la infraestructura subyacente.

Cuadro 5.3.a. Partes del documento con los nuevos requisitos para la migración.

Es de mencionar que esta alternativa implica la recodificación del núcleo de la aplicación pues este no es compatible con el que se cuenta en la actualidad, esta decisión aunque trabajosa de ser llevada adelante, permite realizar una reingeniería de la aplicación en forma completa.
Se ha seleccionado como lenguaje para la rescritura del núcleo de la aplicación a Python, pues este cuenta con gran soporte por parte de Google, es la plataforma más antigua que posee y con más experiencia; además la universidad dueña de la aplicación cuenta con un equipo de programadores con experticia en este lenguaje, entre los cuales se encuentra Reinaldo Álvarez, miembro del equipo original encargado de la inclusión de la funcionalidad de UNDO/REDO dentro de la aplicación.
Sobre costos y soportes ambas empresas no presentan diferencias sustanciales por lo cual estas variables no han sido consideradas al momento de la evaluación final.
Como conclusión final se ha decidido utilizar a Google App Engine con Python como lenguaje de desarrollo´…".
"…La universidad ha tomado la decisión en función de lo antes descripto de capacitar a Reinaldo Álvarez y Pedro Sánchez para que ambos puedan realizar la migración de la aplicación, como jefe de proyecto se a designado a Alberto García y a Juan Fernández como líder para la inclusión de la funcionalidad de UNDO/REDO en el nuevo núcleo del sistema…".

Cuadro 5.3.b. Partes del documento con los nuevos requisitos para la migración.

5.3.2. Descripción y justificación de los pasos a seguir

Consideraciones sobre las fases del proceso a seguir para la migración de la aplicación "*Sistema de Inscripción*". En esta sección, con base en la fases y tareas del proceso propuesto (tabla 4.4), se detalla para cada una de las fases y tareas, si estas deberían ser realizadas o no y la justificación del porqué de cada decisión tomada.

5.3.2.1. Fase Modelado del Servicio (F1):

Tarea Análisis de Usabilidad (F1-T1):

Del extracto del documento de nuevos requerimientos (cuadro 5.3) se extrae el siguiente texto "*…La evolución tecnología hace que sea necesario la adaptación de los sistemas existentes; en tal sentido la masificación en uso de la denominada ´Cloud Computing´, permite a las organizaciones, en ciertas situaciones, reducir costos de mantenimiento de infraestructura y personal dedicado a estas, es por esto que la universidad que administra la aplicación "Sistema de Inscripción" ha decidido evaluar las diversas alternativas existentes en el mercado para tomar una decisión sobre la conveniencia o no de migrar su sistema a la nube …*".

Sobre el extracto del cuadro 5.3 el jefe de proyecto Alberto García, luego de consultar a Pedro Sánchez experto en usabilidad, decide que aunque no es necesario migrar la interfaz de usuario pues la tecnología utilizada, HTML5, JavaScript y AJAX, es soportada por GAE es útil realizar una reingeniería sobre la misma para solucionar los problemas detectados al aplicar el proceso de inspección heurística, en la figura 5.31 (Interfaz Web) y figura 5.32

(Interfaz Móvil) se puede observar el resultado final de aplicar las recomendaciones del experto en usabilidad para la nueva interfaz en cuestión; ante esta situación es menester realizar nuevamente el proceso de inspección heurística

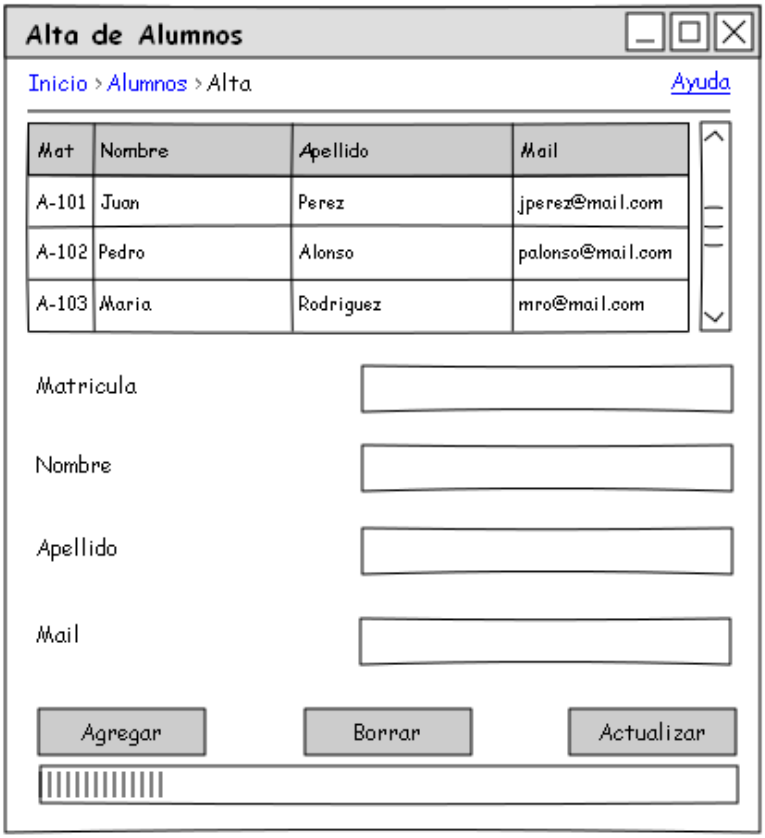


Figura 5.31. Interfaz Web.

Figura 5.32. Interfaz Móvil.

Tarea Detección de Unidades Lógicas de Cambio (F1-T2):

Sobre el extracto de los cuadros 5.1, 5.2 y 5.3 el jefe de proyecto Alberto García, luego de llevar adelante una reunión para unificar conceptos junto al arquitecto Juan Fernández, se concluye que no se ha detectado en la arquitectura cambio alguno que modifique las ULC, en consecuencia se utilizará el documento Evaluación Final del Arquitecto sobre la interfaz "Alta de Alumnos" (Figura 5.5.).

Tarea Detección de Puntos de No Retorno (F1-T3):

Por lo expuesto en F1-T2 (Tarea Detección de Unidades Lógicas de Cambio) el jefe de proyecto Alberto García se dispone a utilizar el documento Deliberable de la etapa de Detección de PNR (Figura 5.8.).

Tarea Estudio de Viabilidad de Servidor de Proximidad (F1-T4):

El jefe de proyecto Alberto García, luego del análisis del cuadro 5.3 no detecta indicio para el uso de un servidor de proximidad, pero al profundizar su análisis detecta en el extracto del documento de nuevos requerimientos (cuadro 5.2) extrae el siguiente texto "*…la aplicación pueda ser accedida desde teléfonos celulares inteligentes con capacidades para navegar en internet y tabletas electrónicas…del éxito que ha sido la inclusión de un servicio en la aplicación en cuestión como agregado de valor a la modificación que se está realizando se sugiere que se realicen los cambios necesarios para que la aplicación pueda interactuar con otros servicios existentes… ".*

El texto extraído implica evaluar si es necesaria la utilización de un servidor de proximidad, es por esta razón que esta tarea debe ser realizada nuevamente.

5.3.2.2. Fase Pruebas de la Aplicación (F2):

Tarea Prueba de Usabilidad de la Aplicación (F2-T5):

Por lo expuesto en F1-T1 (Tarea Análisis de Usabilidad) se toma el documento Evaluación Final del Proceso de Inspección Heurística (Figura 5.3.) como deliberable de esta etapa.

Tarea Prueba de Carga de la Aplicación (F2-T6):

Como base para realizar ésta tarea ell jefe de proyecto Alberto García define que esta tarea no debe ser realizada pues no se encuentra en esta instancia construida la aplicación.

5.3.2.3. Fase Creación del Servicio (F3):

Tarea Especificación de la Configuración de Servicio (F3-T7):

Alberto García, jefe de proyecto, luego de un análisis realizado sobre los cuadros 5.1, 5.2 y 5.3 sumado a las opiniones vertidas por Reinaldo Álvarez y Pedro Sánchez por sus conocimientos en la plataforma de desarrollo y del servicio que implementa la funcionalidad UNDO/REDO, concluye que no es necesario realizar ninguna modificación al servicio que se encuentra en producción y toma como documento deliberable de esta etapa al detallado en la figura 5.15 (Deliberable de la etapa Configuración del Servicio).

Tarea Implementación de la Invocación del Servicio (F3-T8):

Alberto García, jefe de proyecto, en función de la evaluación realizada sobre los cuadros 5.1, 5.2 y 5.3, sumada a las consultas realizadas a Reinaldo Álvarez y Pedro Sánchez, que se han capacitado en el ambiente de programación GAE, se define que se utilizará la documentación generada en la figura 5.25. (Deliberable de la etapa Método de Inclusión), por soportar la misma tecnología para la comunicación entre el sistema y el servicio.

Tarea Implementación de la Inclusión del Servicio (F3-T9):

Alberto García, jefe de proyecto, por lo definido en F3-T8 (Tarea Implementación de la Invocación del Servicio) define que esta tarea debe ser realizada nuevamente para definir de ser necesario, un nuevo conjunto de mensajes al detallado en figura 5.17 (Deliberable de la etapa Inclusión del Servicio).

.

5.3.2.4. Fase Pruebas del Servicio (F4):

Tarea Prueba de Usabilidad del Servicio (F4-T10):

Esta tarea debe ser realizada nuevamente para evaluar la usabilidad de la nueva interfaz luego de la inclusión del servicio según la especificación dada en el cuadro 5.1.

Tarea Prueba de Carga del Servicio (F4-T11):

Esta tarea debe ser realizada nuevamente, evaluar si luego de la inclusión del servicio se siguen con los parámetros de tiempo de respuesta y concurrencia definidos en el cuadro 5.1.

Tarea Evaluación (F4-T12):

Esta tarea debe ser realizada para obtener el cierre del proceso.

A modo de conclusión se presenta la tabla 5.8 donde por fase y etapa si se debe aplicar cada una de ellas.

5.3.3. Pasos a seguir para la migración

En esta sección una vez hecha la evaluación de los pasos a seguir se procede a su ejecución, donde se cumplimentan: Análisis de Usabilidad (5.3.3.1**),** Estudio de Viabilidad de Servidor de Proximidad (5.3.3.2), Prueba de Usabilidad del Servicio (5.3.3.3), Prueba de Carga del Servicio (5.3.3.4) y Evaluación (5.3.3.5).

5.3.3.1. Tarea Análisis de Usabilidad (F1-T1-T)

Como guía para cumplimentar esta tarea se seguirán los pasos definidos en la tabla 4.6 (Técnica de Detección y Evaluación de Requisitos de Usabilidad); para esta tarea se detallan las entradas, técnicas y salidas en la tabla 4.5 (Fases del Proceso) y el flujo del proceso se detalla en la figura 4.1 (Flujo de Proceso).

Entrada 1: Requisitos de Sistema.

Del extracto del documento de relevamiento (cuadro 5.1) se detalla la interfaz a evaluar *"…Figura. Fragmento de la aplicación: Interfaz de Usuario…"*. Tambien se cuenta con el extracto del documento de relevamiento (cuadro 5.2) donde se detalla la interfaz móvil *"…Figura. Boceto de la Interfaz Móvil…"*.

Como se detalló anteriormente el jefe de proyecto Alberto García ordena la realización de una reingeniería sobre las intefaces, como resultado de las mismas se deberá hacer la evaluación sobre las figura 5.31 (Interfaz Web) y figura 5.32. (Interfaz Móvil).

Fase	Tareas	Aplica	Observación
Modelado del Servicio (F1)	Análisis de Usabilidad (F1-T1)	Si	
	Detección de Unidades Lógicas de Cambio (F1-T2)	Si	Se utiliza el documento Evaluación Final del Arquitecto sobre la interfaz "Alta de Profesores" (Figura 5.5.)
	Detección de Puntos de No Retorno (F1-T3)	Si	Se utiliza el utilizar el documento Deliberable de la etapa de Detección de PNR (Figura 5.8.).
	Estudio de Viabilidad de Servidor de Proximidad (F1-T4)	Si	
Pruebas de la Aplicación (F2)	Prueba de Usabilidad de la Aplicación (F2-T5)	Si	Se utiliza el documento Evaluación Final del Proceso de Inspección Heurística (Figura 5.3.)
	Prueba de Carga de la Aplicación (F2-T6)	No	
Creación del Servicio (F3)	Especificación de la Configuración de Servicio (F3-T7)	No	
	Implementación de la Invocación del Servicio (F3-T8)	Si	Se utiliza el documento Deliberable de la etapa Método de Inclusión (Figura 5.25.)
	Implementación de la Inclusión del Servicio (F3-T9)	Si	Se utiliza el documento Deliberable de la etapa Inclusión del Servicio **(**Figura 5.17**)**
Pruebas del Servicio (F4)	Prueba de Usabilidad del Servicio (F4-T10)	Si	
	Prueba de Carga del Servicio (F4-T11)	Si	
	Evaluación (F4-T12)	Si	

Tabla 5.8. Fases y Tareas del Proceso de Inclusión para la aplicación móvil.

Desarrollo de Pasos:

Paso 1: Definir documento de evaluación para cada interfaz.

1.1. Nombre de Interfaz a evaluar asignados por el equipo de desarrollo.

Se fijan los siguientes términos:

Nombre de Sistema: Sistema de Inscripción.

Nombre de Interfaz: Alta de Alumnos.

Nombre Clave: SI_02_AA.

1.2. Descripción general de la interfaz.

Del documento de relevamiento (cuadro 5.1) se extrae el siguiente texto "*…donde los alumnos cargan el número de matrícula, nombres, apellido y mail…"*

Paso 2: Preparación del material y selección del equipo evaluador.

2.1. Selección de usuarios y/o expertos en usabilidad que evaluarán el sistema.

Del documento de relevamiento (cuadro 5.3) se extrae el siguiente texto *"…Reinaldo Álvarez y Pedro Sánchez para que ambos puedan realizar la migración de la aplicación…".* Se selecciona a Reinaldo Álvarez y Pedro Sánchez para la evaluación.

2.2. Disponer de los equipos necesarios para realizar la prueba.

Se cuenta con el documento de relevamiento (cuadro 5.1) de donde se extrae el siguiente texto *"…Las computadoras para cada uno de los integrantes han sido asignadas y todos cuentan con los permisos de acceso necesarios…".* Como se ha definido se cuenta con los accesos necesarios.

Paso 3: Desarrollo del método de Inspección Heurística de Molich y Nielsen.

Se detalla el proceso de evaluación de la aplicación mediante el método de inspección heurística, para esto se siguen los pasos detallados en la tabla 4.6 (Técnica de Detección y Evaluación de Requisitos de Usabilidad).

Finalizada la primera etapa del proceso de evaluación, el equipo ha entregado las siguientes tablas (Figura 5.33, Figura 5.34).

Paso 4: Unificar opiniones de los evaluadores y generar deliberable de la tarea.

Se procederá a generar un documento unificado con el resumen de los distintos evaluadores. Tomando los documentos generados por los auditores (Figura 5.33, Figura 5.34) se calcula el promedio para cada punto a evaluar y se generar un documento final (Figura 35).

Salida: Evaluación de Usabilidad (**F1-T1-R**).

En la figura 5.36 (Evaluación Final del Proceso de Inspección Heurística) se presenta el documento final, que representa la evaluación de usabilidad para la interfaz “Alta de Alumnos” del “Sistema de Inscripción”.

Documento de Evaluación de Interfaz			
Fecha	2013-10-05	Responsable	Reinaldo Álvarez
Nombre de Sistema	Sistema de Inscripción		
Nombre de Interfaz	Alta de Alumnos	Nombre Clave	SI_02_AA
Descripción General			
Esta interfaz es utilizada para la carga, baja o modificaciones de alumnos.			
Puntos a Evaluar		Valor (Rango de 1-5) 1=Malo - 5=Excelente	
1.	Los cuadros de diálogos son simples y naturales.	3	
2.	La aplicación habla el lenguaje del usuario.	4	
3.	Se minimiza la carga de memoria en el usuario.	4	
4.	Hay consistencia en los datos pedido por la interfaz.	4	
5.	La interfaz proporciona información de proceso.	2	
6.	La interfaz proporciona métodos de salida.	2	
7.	La interfaz proporciona caminos abreviados.	3	
8.	La interfaz proporciona mensajes de error.	4	
9.	La aplicación previene errores.	2	
Firma			
Documento Versión	0.0.1		

Figura 5.34. Evaluación Final del Proceso de Inspección Heurística de Reinaldo Álvarez.

Documento de Evaluación de Interfaz			
Fecha	2013-10-05	Responsable	Pedro Sánchez
Nombre de Sistema	Sistema de Inscripción		
Nombre de Interfaz	Alta de Alumnos	Nombre Clave	SI_02_AA
Descripción General			
Esta interfaz es utilizada para la carga, baja o modificaciones de alumnos.			
Puntos a Evaluar		Valor (Rango de 1-5) 1=Malo - 5=Excelente	
1.	Los cuadros de diálogos son simples y naturales.	4	
2.	La aplicación habla el lenguaje del usuario.	5	
3.	Se minimiza la carga de memoria en el usuario.	5	
4.	Hay consistencia en los datos pedido por la interfaz.	3	
5.	La interfaz proporciona información de proceso.	2	
6.	La interfaz proporciona métodos de salida.	2	
7.	La interfaz proporciona caminos abreviados.	3	
8.	La interfaz proporciona mensajes de error.	3	
9.	La aplicación previene errores.	3	
Firma			
Documento Versión	0.0.1		

Figura 5.35. Evaluación Final del Proceso de Inspección Heurística de Pedro Sánchez.

Resumen de Evaluación de Interfaz			
Fecha	2013-10-06	Auditor Líder	Juan Fernández
Nombre de Sistema	Sistema de Inscripción		
Nombre de Interfaz	Alta de Alumnos	Nombre Clave	SI_02_AA
Descripción General			
Esta interfaz es utilizada para la carga, baja o modificaciones de alumnos.			
Puntos a Evaluar		Valor (Rango de 1-5) 1=Malo - 5=Excelente	
1.	Los cuadros de diálogos son simples y naturales.	3.5	
2.	La aplicación habla el lenguaje del usuario.	4.5	
3.	Se minimiza la carga de memoria en el usuario.	4.5	
4.	Hay consistencia en los datos pedido por la interfaz.	3.5	
5.	La interfaz proporciona información de proceso.	2	
6.	La interfaz proporciona métodos de salida.	2	
7.	La interfaz proporciona caminos abreviados.	3	
8.	La interfaz proporciona mensajes de error.	3.5	
9.	La aplicación previene errores.	2.5	
Firma			
Documento Versión	0.0.2		

Figura 5.36. Evaluación Final del Proceso de Inspección Heurística.

5.3.3.2. Estudio de Viabilidad de Servidor de Proximidad (F1-T4-T)

Del documento de relevamiento (cuadro 5.3) se extrae el siguiente texto "…*Juan Fernández como líder para la inclusión de la funcionalidad de UNDO/REDO en el nuevo núcleo del sistema...".* El arquitecto Juan Fernández será el evaluador responsable de la inclusión de la funcionalidad de UNDO/REDO en la aplicación.

El líder define que es necesario realizar un nuevo cálculo de viabilidad del servidor de proximidad, se detallan los pasos realizados a continuación.

Entrada 1: Requerimiento de Sistema.

> Como extensión se evalúa el extracto del documento de nuevos requerimientos (cuadro 5.2) se extrae el siguiente texto "…*la aplicación pueda ser accedida desde teléfonos celulares inteligentes con capacidades para navegar en Internet y tabletas electrónicas…del éxito que ha sido la inclusión de un servicio en la aplicación en cuestión como agregado de valor a la modificación que se está realizando se sugiere que se realicen los cambios necesarios para que la aplicación pueda interactuar con otros servicios existentes…"*

Desarrollo de Pasos:

Paso 1: Cumplimentar el proceso definido como Evaluación de Viabilidad

> Como se definió anteriormente el arquitecto Juan Fernández será el encargado de llevar adelante el estudio de viabilidad. En la tabla 5.9 se presenta el resulto a las preguntas sugeridas.

ID	Pregunta asociada	Respuesta
1	Existen restricciones de seguridad	Si
2	Deben ser integrados varios servicios	Si
3	Existe la posibilidad de cambio en los proveedores de servicios	Existe
4	Se accederá a la aplicación desde múltiples dispositivos	Si
5	Existen cambios frecuentes en la aplicación	Si
6	Se desea contar con mejoras pos implementación	No

Tabla 5.9. Respuestas obtenidas.

Conversión a intervalos difusos:

En la tabla 5.10 se detallan la conversión a intervalos difusos, se ha tomado como referencia a la tabla 4.3.

Si	**10**
No	**0**

Tabla 5.10. Intervalos Difusos.

Definición de Pesos

En la tabla 5.11 se muestran los pesos luego del calculo de los intervalos difusos.

ID	Pregunta asociada	Peso
1	Existen restricciones de seguridad	10
2	Deben ser integrados varios servicios	10
3	Existe la posibilidad de cambio en los proveedores de servicios	8
4	Se accederá a la aplicación desde múltiples dispositivos	10
5	Existen cambios frecuentes en la aplicación	9
6	Se desea contar con mejoras pos implementación	0

Tabla 5.11. Cálculo de pesos.

Detalle de cada respuesta

1. Existen restricciones de seguridad: Del documento de relevamiento (cuadro 5.1) se extrae el siguiente texto *"…Existen tres tipos de perfiles de usuarios para la aplicación, una es el administrador con permiso general para acceder a todo el sistema, el perfil profesor que puede definir el curso que dictará, el día y el horario del mismo y si este necesita algún requerimiento especial para su dictado. Con esto se asigna automáticamente las aulas para los cursos. Con la grilla de cursos y horarios el tercer perfil, alumnos, que puede darse de alta e inscribe hasta el tope máximo que permite cada aula..."* En este extracto se detallan tres perfiles de usuarios distintos y permisos diferentes para cada uno, con lo cual se define que existen restricciones de seguridad.
2. Deben ser integrados varios servicios: En el análisis realizado a la documentación (cuadro 5.2) se extrae "…*del éxito que ha sido la inclusión de un servicio en la aplicación en cuestión como agregado de valor a la modificación que se está realizando se sugiere que se*

realicen los cambios necesarios para que la aplicación pueda interactuar con otros servicios existentes..", esto define la necesidad de contemplar la integración de otros servicios

3. Existe la posibilidad de cambio en los proveedores de servicios: Por lo detallado en el punto 2, se deduce que podrá haber cambio de proveedores de servicios.
4. Se accederá a la aplicación desde múltiples dispositivos: En el análisis realizado a la documentación (cuadro 5.2) "…*que la aplicación pueda ser accedida desde teléfonos celulares inteligentes con capacidades para navegar en internet y tabletas electrónicas...*" se hace referencia al tipo de dispositivo con lo cual concluye que se accederá a la aplicación desde diversos dispositivos.
5. Existen cambios frecuentes en la aplicación: En el análisis realizado a la documentación (cuadro 5.1, cuadro 5.2 y cuadro 5.3) no se detecta la existencia de cambios frecuentes en la aplicación pero el hecho de estar realizando este cambio implica que existe alguna posibilidad de cambio.
6. Se desea contar con mejoras pos implementación: En el análisis realizado a la documentación (cuadro 5.1, cuadro 5.2 y cuadro 5.3) no se detecta la posibilidad de implementar mejoras pos implementación.

Cálculo de Promedio

La suma de los pesos es 48 esto se divide por 6, la cantidad de preguntas asociadas, como resultado se obtiene 8.

Como el valor obtenido es mayor a 5 es recomendable la inclusión de un servidor de proximidad.

Salida: Estudio de Viabilidad (**F2-T4-R**).

En la figura 5.37 (Deliberable de la etapa Estudio de Viabilidad) se presenta el documento final, se detallla la evaluación del estudio de viabilidad.

Documento Estudio de Viabilidad			
Fecha	2013-10-15	Responsable	Juan Fernández
Nombre de Sistema	Sistema Administración Curricular		
Recomendación Final	Aceptado		
Observaciones			
Como el valor obtenido es 8 (ocho) y por ser mayor a 5 se recomienda la inclusión de un servidor de proximidad.			
Firma			
Documento Versión	0.0.1		

Figura 5.37. Deliberable de la etapa Estudio de Viabilidad.

5.3.3.3. Implementación de la Invocación del Servicio (F3-T8-T)

Como guía para cumplimentar esta tarea se seguirán los pasos definidos en la tabla 4.14 (Inclusión del Servicio en la Aplicación); para esta tarea se detallan las entradas, técnicas y salidas en la tabla 4.5 (Fases del Proceso) y el flujo del proceso se detalla en la figura 4.1 (Flujo de Proceso).

Del extracto del documento de nuevos requerimientos (cuadro 5.2) se extrae el siguiente texto "*…un experto en usabilidad Pedro Sánchez…".* El programador Pedro Sánchez es el encargado de realizar el análisis de la documentación y validar la aplicación anfitriona.

Este considera en función de lo expuesto en el cuadro 5.2 "*…se desea que la aplicación pueda ser accedida desde teléfonos celulares inteligentes con capacidades para navegar en internet y tabletas electrónicas…del éxito que ha sido la inclusión de un servicio en la aplicación en cuestión como agregado de valor a la modificación que se está realizando se sugiere que se realicen los cambios necesarios para que la aplicación pueda interactuar con otros servicios existentes…"* y del resultado obtenido en Tarea Análisis de uso de Servidor de Proximidad (F2-T4) se define rehacer esta tarea nuevamente actualizando a la nueva situación.

Entrada 1: Sistema.

> Del extracto del documento de relevamiento (cuadro 5.1) se detalla la interfaz a evaluar *"…Figura. Fragmento de la aplicación: Interfaz de Usuario…*" y del extracto del cuadro 5.2 *"…Figura; Bosquejo de la Interfaz Móvil…"*.

Entrada 2: Diseño de ULC.

> Se toma el documento F1-T2-R.

Entrada 3: Documentación de Sistema.

Se toma el texto redactado en el cuadro 5.1 y cuadro 5.2 para su análisis.

Desarrollo de Pasos:

Paso 1: Analizar la documentación del sistema.

1.1 En este paso se debe evaluar cual fue el modelo seleccionado para la construcción del sistema anfitrión.

Del documento de relevamiento (cuadro 5.1) se extrae el siguiente texto *"…se ha utilizado HTML5 y Java Script, siendo Ajax con mensajería Json…".* Se detecta que el método de comunicación de la aplicación es HTML con AJAX, en el cuadro 5.2 no se hace referencia a la tecnología utilizada con lo que se deduce que continua la existente y descrita en el cuadro 5.1.

1.2 Verificar que la documentación este alineada con la versión de producción del sistema anfitrión.

Del extracto del documento de relevamiento (cuadro 5.1) se detalla la interfaz a evaluar *"…Figura. Fragmento de la aplicación: Interfaz de Usuario…*". Del análisis realizado de la aplicación por el programador Pedro Sánchez se confirma que la aplicación utiliza HTML y AJAX, sobre la información detallada en el cuadro 5.2 no se realiza el proceso de verificación pues se está desarrollando la aplicación.

Paso 2: Definir alternativas para la inclusión

2.1 Definir el conjunto de alternativas para la inclusión del servicio en función de la tecnología utilizada en la aplicación anfitriona.

Se toman las alternativas definidas para la inclusión de la aplicación por no detectarse cambios en la comunicación:

1. Utilizar un modelo de mensajería basado en XML.
2. Utilizar un modelo de mensajería basado en JSon.

Como decisión final el programador Pedro Sánchez toma las conclusiones definidas para la etapa de inclusión del servicio y utiliza Json como medio de comunicación.

Como agregado a esto el programador Pedro Sánchez define en función de lo detallado en el cuadro 5.2, que el servidor de proximidad será utilizado en esta etapa como un puente entre la aplicación el servicio, por no requerirse de conexión a otro servicio, se deja el modelo de comunicación ya definido. Por otra parte el servidor de proximidad, reconocerá el mismo el dispositivo desde el cual se está comunicando el usuario y proveerá el conjunto de interfaces acorde al dispositivo.

Paso 3: Generación deliberable de la tarea

3.1 Generar Reporte con la mensajería del Servicio.

Pedro Sánchez genera el reporte final de la Evaluación del Método de Inclusión.

Salidas: Método de Inclusión (**F3-T8-R**)

En la figura 5.25 (Deliberable de la etapa Método de Inclusión) se detalla el deliberable final de esta Fase.

5.3.3.4. Inclusión del Servicio en la Aplicación (F3-T9-T)

El arquitecto Juan Fernández designa a Reinaldo Álvarez para la inclusión de la mensajería en la aplicación.

Este último define en función de lo detallado en la tarea Implementación de la Invocación del Servicio (F3-T8) que se debe utilizar el mismo conjunto de mensajes por no haberse realizado cambio alguno, en consecuencia el deliberable de esta etapa es el documento Deliberable de la etapa Inclusión del Servicio (Figura 5.17).

5.3.3.5. Prueba de Usabilidad del Servicio (F4-T10-T)

Como guía para cumplimentar esta tarea se seguirán los pasos definidos en la tabla 4.15 (Prueba de Usabilidad del Servicio); para esta tarea se detallan las entradas, técnicas y salidas en la tabla 4.5 (Fases del Proceso) y el flujo del proceso se detalla en la figura 4.1 (Flujo de Proceso).

Entrada 1: Sistema + Invocación al Servicio.

Del extracto del documento de relevamiento (cuadro 5.1) se detalla la interfaz a evaluar "*…Figura. Fragmento de la aplicación: Interfaz de Usuario…*".

Tambien se cuenta con el extracto del documento de relevamiento (cuadro 5.2) donde se detalla la interfaz móvil *"…Figura. Boceto de la Interfaz Móvil…"*.

Como se detalló anteriormente el jefe de proyecto Alberto García sugiere la realización de una reingeniería sobre las intefaces, como resultado de las mismas se deberá hacer la evaluación sobre las figura 5.31 (Interfaz Web) y figura 5.32 (Interfaz Móvil).

Desarrollo de Pasos:

Paso 1: Repetir el procedimiento definido en F1-T1-T.

El arquitecto Juan Fernandez procede con la recolección de los documentos generados en F1-T1-T.

1.1. Evaluar la aplicación interfaz por interfaz.

El arquitecto Juan Fernández asigna la tarea al mismo equipo de trabajo definido en la F1-T1, en las figuras 5.38 y 5.39 se detalla el resultado obtenido por los evaluadores.

Documento de Evaluación de Interfaz			
Fecha	2013-10-20	Responsable	Reinaldo Álvarez
Nombre de Sistema	Sistema de Inscripción		
Nombre de Interfaz	Alta de Alumnos	Nombre Clave	SI_02_AA
Descripción General			
Esta interfaz es utilizada para la carga, baja o modificaciones de alumnos.			
Puntos a Evaluar		Valor (Rango de 1-5) 1=Malo - 5=Excelente	
1.	Los cuadros de diálogos son simples y naturales.	3	
2.	La aplicación habla el lenguaje del usuario.	4	
3.	Se minimiza la carga de memoria en el usuario.	4	
4.	Hay consistencia en los datos pedido por la interfaz.	4	
5.	La interfaz proporciona información de proceso.	2	
6.	La interfaz proporciona métodos de salida.	2	
7.	La interfaz proporciona caminos abreviados.	3	
8.	La interfaz proporciona mensajes de error.	4	
9.	La aplicación previene errores.	2	
Firma			
Documento Versión	0.0.1		

Figura 5.38. Evaluación del Proceso de Inspección Heurística Reinaldo Álvarez.

Documento de Evaluación de Interfaz			
Fecha	2013-10-20	Responsable	Pedro Sánchez
Nombre de Sistema	Sistema de Inscripción		
Nombre de Interfaz	Alta de Alumnos	Nombre Clave	SI_02_AA
Descripción General			
Esta interfaz es utilizada para la carga, baja o modificaciones de alumnos.			
Puntos a Evaluar		Valor (Rango de 1-5) 1=Malo - 5=Excelente	
1.	Los cuadros de diálogos son simples y naturales.	4	
2.	La aplicación habla el lenguaje del usuario.	5	
3.	Se minimiza la carga de memoria en el usuario.	5	
4.	Hay consistencia en los datos pedido por la interfaz.	3	
5.	La interfaz proporciona información de proceso.	2	
6.	La interfaz proporciona métodos de salida.	2	
7.	La interfaz proporciona caminos abreviados.	3	
8.	La interfaz proporciona mensajes de error.	3	
9.	La aplicación previene errores.	3	
Firma			
Documento Versión	0.0.1		

Figura 5.39. Evaluación del Proceso de Inspección Heurística Pedro Sánchez.

Paso 2: Unificar opiniones de los evaluadores.

2.1. Se procederá a generar un documento unificado con el resumen de los distintos evaluadores.

El arquitecto Juan Fernández procede a unificar los documentos.

Salida: Evaluación de Usabilidad (**F4-T10-R**).

En la figura 5.40 (Evaluación Final del Proceso de Inspección Heurística) se presenta el documento final, que representa la evaluación de usabilidad para la interfaz "Alta de Alumnos Móvil".

Resumen de Evaluación de Interfaz			
Fecha	2013-10-20	Auditor Líder	Juan Fernández
Nombre de Sistema	Sistema de Inscripción		
Nombre de Interfaz	Alta de Alumnos	Nombre Clave	SI_01_AA
Descripción General			
Esta interfaz es utilizada para la carga, baja o modificaciones de alumnos.			

Puntos a Evaluar		Valor (Rango de 1-5) 1=Malo - 5=Excelente
1.	Los cuadros de diálogos son simples y naturales.	3.5
2.	La aplicación habla el lenguaje del usuario.	1
3.	Se minimiza la carga de memoria en el usuario.	4.5
4.	Hay consistencia en los datos pedido por la interfaz.	3.5
5.	La interfaz proporciona información de proceso.	2
6.	La interfaz proporciona métodos de salida.	2
7.	La interfaz proporciona caminos abreviados.	3
8.	La interfaz proporciona mensajes de error.	3.5
9.	La aplicación previene errores.	2.5

Firma	
Documento Versión	0.0.2

Figura 5.40. Evaluación Final del Proceso de Inspección Heurística.

5.3.3.6. Prueba de Carga del Servicio (F4-T11-T)

Como guía para cumplimentar esta tarea se seguirán los pasos definidos en la tabla 4.16 (Prueba de Carga del Servicio); para esta tarea se detallan las entradas, técnicas y salidas en la tabla 4.5 y el flujo del proceso se detalla en la figura 4.1.

Entrada 1: Sistema + Invocación al Servicio.

El jefe de proyecto Alberto García sugiere la realización de un proceso de reingeniería sobre las intefaces, como resultado de las mismas se deberá hacer la evaluación sobre las figura 5.31 (Interfaz Web) y figura 5.32. (Interfaz Móvil).

Desarrollo de Pasos:

Paso 1: Tomar el programa construido para la ejecución de la prueba en: Caso 1, Aplicación Web, F2-T6-T, Paso 2.

El arquitecto Juan Fernández procede a recolectar la documentación generada en la fase y tarea especificada, aquí se mantiene el realizado originalmente en el caso 1

(Aplicación Web). Además asigna al mismo equipo encargado de las pruebas para realizarlas. Pedro Sánchez procese a recuperar el programa generado para la prueba de carga.

Paso 2: Preparación del material y selección del equipo evaluador

2.1. Selección del experto que evaluará el sistema.

Se selecciona al programador Pedro Sánchez para ejecutar la prueba de carga.

2.2. Disponer de los equipos necesarios para realizar la prueba.

Del documento de relevamiento (cuadro 5.1) se extrae el siguiente texto *"…Las computadoras para cada uno de los integrantes han sido asignadas y todos cuentan con los permisos de acceso necesarios…".* Como se ha definido se cuenta con los accesos necesarios.

Paso 3: Ejecución de la prueba

3.1. Realizar una prueba de validación.

Pedro Sánchez ejecuta la prueba.

Paso 4: Generación del reporte de prueba de carga

4.1. Se procederá a generar un documento unificado con el resumen de la prueba, el cual será el documento deliberable de la etapa. Pedro Sánchez genera el reporte final de la prueba de ejecución.

Salida: Reporte de Carga (**F4-T11-R**).

En la figura 5.41 (Deliberable de la etapa Prueba de Carga de la Aplicación con Servicio) se detalla el documento final como deliberable de la presente tarea.

Documento de Prueba de Carga					
Fecha	2013-10-27	Hora	17:30	Responsable	Pedro Sánchez
Nombre de Sistema		Sistema de Inscripción			
Nombre de Interfaz		Alta de Alumnos		Nombre Clave	SI_02_AA
Concurrencia Máxima		100			
Tiempo Esperado		Menos de 5 segundos		Tiempo Obtenido	3.5
Observaciones					
Esta interfaz es utilizada para la carga, baja o modificaciones de alumnos.					
Repetir para todas las interfaces					
Firma					
Documento Versión		0.0.1			

Figura 5.41. Deliberable de la etapa Prueba de Carga de la Aplicación con Servicio.

5.3.3.7. Evaluación (F4-T12)

Como guía para cumplimentar esta tarea se seguirán los pasos definidos en la tabla 4.14 (Técnica de Evaluación); para esta tarea se detallan las entradas, técnicas y salidas en la tabla 4.5 (Fases del Proceso) y el flujo del proceso se detalla en la figura 4.1 (Flujo de Proceso).

Entrada 1: F1-T1-R, figura 5.3 Evaluación Final del Proceso de Inspección Heurística, por lo expuesto por Pedro Sánchez.

Entrada 2: F4-T10-R, figura 5.28 Evaluación Final del Proceso de Inspección Heurística.

Entrada 3: F4-T11-R, figura 5.29 Deliberable de la etapa Prueba de Carga de la Aplicación con Servicio.

Desarrollo de Pasos:

Paso 1: Comparar los Deliberables F2-T5-R y F4-T10-R.

El arquitecto Juan Fernández procede a realizar la comparación de los diversos deliberables y concluye el ambos son similares.

Paso 2: Comparar los Deliberables F2-T6-R y F4-T11-R.

El arquitecto Juan Fernández procede a realizar la comparación, aquí no se cuenta con el deliberable F2-T6-R, por se una aplicación nueva, la interfaz móvil, con lo

cual se cuenta con el documento F4-T11.R el cual cumple con los tiempos y la concurrencia especificados en el cuadro 5.1.

Paso 3: Evaluación final

El arquitecto Juan Fernández procede a realizar la comparación final, al ser satisfactoria las comparaciones hechas en los pasos uno y dos se da por finalizada en forma satisfactoria la inclusión del servicio.

Paso 4: Generación deliberables de la tarea.

4.1. Este deliberable es la aceptación del proceso de inclusión.

El arquitecto Juan Fernández genera el reporte final.

Salida: Reporte de Cierre (**F4-T12-R**).

En la figura 5.42 (Deliberable final de la inclusión del Servicio) se detalla el documento final como deliberable del proceso.

Documento de evaluación final			
Fecha	2013-10-29	Responsable	Juan Fernández
Nombre de Sistema	Sistema de Inscripción		
Recomendación Final	Aceptado		
Observaciones			
De la comparación de las pruebas realizadas no se observan cambios significantes antes y después de la inclusión de la funcionalidad de UNDO/REDO. En consecuencia se ha logrado el objetivo de agregar la funcionalidad sin modificar la performance y mejorar la usabilidad de la aplicación.			
Firma			
Documento Versión	0.0.1		

Figura 5.42. Deliberable final de la inclusión del Servicio.

5.4. CASOS VALIDADOS

En la tabla 5.12 se presenta a modo de resumen los casos presentados y el espectro de posibilidades que se han evaluado.

Característica	**Caso 1: Web**	**Caso 2: Móvil**	**Caso 3: Nube**
Aplicación Existente	Si		
Ampliación de App. Existente		Si	
Aplicación Nueva			Si
Utiliza Proxy		Si	Si
Hereda Proceso		Si	Si
Plataforma	Web	Móvil	Nube

Tabla 5.12. Casos Validados.

6. CONCLUSIONES

En este capítulo se presentan las aportaciones de esta tesis doctoral (Sección 6.1) y se destacan las futuras líneas de investigación que se consideran de interés en base al problema que se abordó en este trabajo de Tesis (Sección 6.2).

6.1. APORTACIONES DE LA TESIS

En esta sección se presentan los aportes derivados de las preguntas de investigación de la Tesis (Sección 6.1.1) y las conclusiones generales (Sección 6.1.2).

6.1.1. Aportes derivados de las preguntas de investigación de la Tesis

A continuación se procede a responder a las preguntas que se formularon en la sección 3.3 correspondiente al Sumario de Investigación.

La primera pregunta del Sumario rezaba lo siguiente:

"¿Es posible desarrollar un proceso para la inclusión de la funcionalidad UNDO/REDO que sea simple y rápido de incluir en una aplicación nueva o existente?"

Para dar respuesta a este primer interrogante, cabe hacer mención al proceso que se desarrolla en el capítulo correspondiente a la solución del problema (sección 4); en el cuál se ha presentado un proceso de integración de la funcionalidad UNDO/REDO, poniendo de manifiesto la evolución que ha tenido este trabajo de Tesis, desde un concepto de patrón único para la funcionalidad UNDO/REDO a un modelo SOA.

Este modelo de proceso dividido en fases, donde cada una de ellas está constituida por tareas, que a su vez son implementadas por medio de técnicas desarrolladas especialmente a tal efecto, para las cuales se han definido entradas, actividades y salidas; sumado a un servicio Web que encapsula la funcionalidad UNDO/REDO, el cual permite integrar la funcionalidad tanto sea en una aplicación nueva como una ya existente. Esto se demuestra en el capítulo 5 correspondiente a Casos de Validación, de manera rápida simple y sistematizada, reduciendo el nivel de experticia del equipo encargado de la inclusión en la aplicación anfitriona.

La segunda pregunta del Sumario se expresaba de la siguiente forma:

"¿Es posible definir una abstracción para el modelo de datos a ser manejada por la funcionalidad de UNDO/REDO?"

Para dar respuesta a esta segunda pregunta, cabe hacer mención al concepto de "Unidades Lógicas de Cambio" que se introduce en el capítulo correspondiente a la solución del problema (sección 4.3.3); el cual hace referencia a los datos que deben ser tratados como un conjunto indivisible a los efectos de conservar la coherencia de la información en un sistema de la funcionalidad UNDO/REDO. En otras palabras, se puede afirmar que este concepto determina la granularidad de los datos para el manejo de la funcionalidad, dado que el mismo puede diferir del utilizado en la aplicación.

Aunque el manejo de granularidad es conocido en el dominio de las bases de datos, no lo es así en el ámbito de la usabilidad de software. La inclusión del concepto unidad lógica de cambio en el ámbito de usabilidad para la funcionalidad UNDO/REDO es central para la orquestación de la misma; habida cuenta de que este concepto permite reconocer el modelo de almacenamiento más adecuado para cada aplicación, ampliando de esta manera el grado de abstracción para el diseño de sistemas.

En la misma línea que el concepto explicado, se pasa a detallar la relación existente entre los "Puntos de No Retorno" y su relación con las unidades lógicas de cambio, desarrollada en la sección 4.2.4 del capítulo correspondiente a la solución del problema.

A la sazón los puntos de no retorno no son un concepto nuevo, dado que el mismo ha sido ampliamente tratado en el área de base de datos, en este trabajo de tesis se lo presenta desde el punto de vista de una asociación necesaria con las unidades lógicas de cambio para la construcción de un proceso de UNDO/REDO; ya que los puntos de no retorno son los que delimitan la cantidad de posibles estados hacia atrás que las unidades lógicas de cambio podrán tener. En este sentido se puede afirmar que una está anclada a la otra y que la suma de ambas genera un concepto de implicancias mayores que si se consideraran por separado.

En virtud de estas consideraciones, es importante destacar que ambos conceptos ("Unidades Lógicas de Cambio" y "Puntos de No Retorno") permiten generar un grado de abstracción suficiente para que el modelo se adapte a diversos tipos de dominio, que actuaría a modo de contenedor sin tener en cuenta el contenido, sino solo el modelo de proceso.

Como extensión a esto, el servicio puede ser utilizado en ambientes colaborativos, ya que los puntos de no retorno pueden actuar como mecanismos de mediación entre los diferentes actores que colaboran entre sí.

6.1.2. Conclusiones Generales

Como aporte adicional a los expresados en la sección 6.1.1 correspondiente a las aportaciones derivadas de las preguntas de investigación de la Tesis, es necesario citar el reconocimiento de la existencia de dos tipos de aplicaciones que utilizan la funcionalidad de UNDO/REDO.

Por una parte caben considerarse las aplicaciones para las cuales la funcionalidad UNDO/REDO es fundamental y no es posible contemplar la misma sin su existencia, siendo el ejemplo paradigmático un editor de texto.

En otro sentido, también es importante hacer mención a un conjunto de aplicaciones para las cuales la funcionalidad UNDO/REDO es recomendable pero no es el núcleo de la aplicación; debiéndose destacar, que el presente trabajo de Tesis se ha focalizado dentro de esta categoría de aplicaciones. Por su parte, es importante señalar que este hecho no impide que los lineamientos desarrollados en este trabajo puedan ser utilizados como guías para dar solución a las aplicaciones correspondientes a la primera categoría.

Siguiendo con la línea de aportes, se estima de interés hacer mención a la definición de una descripción formal de transformaciones, sección 4.2.2, la cual se considera de suma utilidad para validar un proceso de UNDO/REDO.

A modo de cierre, el proceso de construcción de conocimiento llevado a cabo en esta Tesis junto a la validación de los ejemplos desarrollados, permite inferir que la usabilidad es un concepto que evoluciona con el tiempo en función del uso de los sistemas.

Conforme a esta idea, se asume que es fundamental extender el concepto de usabilidad para que este incluya la variable tiempo; dado que la interacción del usuario con la aplicación hace que el mismo realice un proceso cognitivo que aumenta su capacidad y conocimiento del sistema. De esta manera, el usuario se halla en condiciones de reconocer atajos en el trabajo que hace con el sistema y genera así una nueva relación con la aplicación que demandará ser suplida por nuevas funcionalidades de usabilidad.

Este proceso es de carácter evolutivo; y en este sentido, una propuesta de usabilidad que intenta dar solución a un problema reconocido en el dominio de la usabilidad y no contemple los procesos cognitivos de los usuarios, tendrá una utilidad acotada en el tiempo.

6.2. FUTURAS LÍNEAS DE INVESTIGACIÓN

Del proceso de investigación llevado a cabo en este trabajo de Tesis se desprenden a juicio del autor las siguientes líneas de investigación:

I. Como se ha detallado en el desarrollo del presente trabajo, existen un conjunto de funcionalidades de uso común en el dominio de la usabilidad de software, como ser las funcionalidades Cancel, Wizard y Feedback, que deberían ser agregadas al servicio que implementa la funcionalidad de UNDO/REDO para obtener un servicio que satisfaga las principales características de usabilidad de una aplicación.

II. Como desprendimiento del punto anterior y en función de la experiencia ganada en el proceso de inclusión para la funcionalidad UNDO/REDO, se considera que es necesario adicionar procesos específicos para las funcionalidades de usabilidad Cancel, Wizard y Feedback, esto sumado al reconocimiento de las características particulares de los mismos a los ya existentes en el UNDO/REDO dará un marco en el cual a priori se generarán modificaciones en los procesos ya definidos.

III. En virtud de los conceptos mencionados, es menester continuar con la experimentación del proceso de inclusión tanto sea para la funcionalidad de UNDO/REDO como para las otras funcionalidades, en poblaciones de programadores y diseñadores segmentadas por experticia, obteniendo de esta manera información necesaria para refinar los procesos y métodos detallados para poder realizar nuevas inferencias y realizar mejoras al servicio y al proceso de UNDO/REDO.

IV. De lo expresado en la sección 5.3 se desprende que es necesario continuar con la experimentación en poblaciones con poca experiencia para poder validar la mejora en el tiempo de inclusión de la funcionalidad de UNDO/REDO en una aplicación.

V. Una de las características que permiten ser soportadas por el servicio es el trabajo compartido y distribuido dado por la combinación de los conceptos de unidades lógicas de cambio y puntos de no retorno. Estas características habilitan al servicio para el trabajo en entornos

colaborativos, es por esto que se hace menester seguir experimentando en estos ambientes para poder validar la robustez del modelo.

VI. La formalización presentada para la funcionalidad UNDO/REDO, además de un concepto novel de esta tesis, deja sentada las bases para la generación de un modelo formal para las demás funcionalidades de usabilidad permitiendo así que la comunidad de usabilidad lo valide y se generalice el uso de estas características.

VII. Como se ha definido en las conclusiones de este trabajo se han reconocido dos tipos de aplicaciones en las cuales puede ser incluida la funcionalidad de UNDO/REDO, la cual puede ser para la aplicación una funcionalidad central o deseable. El modelo se ha desarrollado para las del segundo tipo, y sería de esperar que se pueda extender este modelo para que pueda ser usado, de ser posible, en aplicaciones donde la funcionalidad de UNDO/REDO sea central.

VIII. El proceso presentado en esta tesis implica el agregado de una tarea más al proceso de construcción de sistemas; por tal razón, que una posible líneas de investigación se focalice en la integración desde los casos de uso mediante el desarrollo de un modelo de procesamiento de lenguaje natural para detectar los posibles hitos donde el sistema necesitará utilizar la funcionalidad de UNDO/REDO y así iniciar el proceso en forma automática.

IX. Como se ha podido observar el modelo implementado de un servidor de proximidad de usabilidad ha revelado ser un mecanismo viable para facilitar la implementación de una aplicación que deba soportar el acceso de usuarios de distintos dispositivos y así poder extender este modelo a toda la aplicación.

X. Por consiguiente y en línea con el punto inmediato anterior se recomienda investigar en el área de la trazabilidad de las aplicaciones, dado que las mismas, al encontrarse construidas bajo un modelo de servidor de proximidad, podrán contar con un mecanismo de mejora perfectiva. Si a este hecho se le agrega un modelo de aprendizaje automático, los cambios en los comportamientos de usuarios podrán ser reconocidos por el sistema y en cierta medida adaptar el mismo a las nuevas características de uso.

7. REFERENCIAS

Abowd, G.; Dix, A. 1992. *Giving undo attention*. Interacting with Computers, Volume 4, Issue 3, December 1992, pp. 317–342. DOI: 10.1016/0953-5438(92)90021-7.

Abrams, S.; Oppenheim, D. 2001. *Method and apparatus for combining UNDO and redo contexts in a distributed access environment*. PN: 6.192.378 US.

Alexander, C 1979. *The Timeless Way of Building*. Oxford University Press.

Apple 2012. Apple http://www.apple.com/. Página válida al 2012-12-20.

Ardagna, C.; Damiani, E.; Frati, F.; Rebeccani, D.; Ughetti, M. 2012. *Scalability Patterns for Platform-as-a-Service*. Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on, pp. 718-725. DOI: 10.1109/CLOUD.2012.41. 2012.06.24.

Armbrust, M.; Fox, A.; Griffith, R.; Joseph, A.; Katz, R.; Konwinski, A.; Lee, G; Patterson, D.; Rabkin, A; Stoica, I.; Zaharia, M. 2009. *Above the clouds: A berkeley view of cloud computing*, in Tech. Rep. UCB/EECS-2009-28, EECS Department, U.C. Berkeley.

Armbrust, M.; Fox, A.; Griffith, R.; Joseph, A.; Katz, R.; Konwinski, A.; Lee, G.; Patterson, D.; Rabkin, D.; Stoica, I.; Zaharia, A. 2010. *A View of Cloud Computing*. Communications of the ACM. April 2010 vol. 53 no. 4. DOI: 10.1145/1721654.1721672.

Azeez, A.; Perera, S. ; Gamage, D. ; Linton, R. ; Siriwardana, P. ; Leelaratne, D. ; Weerawarana, S. ; Fremantle, P. (2010). Multi-tenant SOA Middleware for Cloud Computing.Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on. Page(s): 458-465. ISBN: 978-1-4244-8207-8. DOI: 10.1109/CLOUD.2010.50.

Baker, B.; Storisteanu, A. 2001. *Texts edit system with enhanced UNDO user interface*. PN: 6.185.591 US.

Bates, C. y Ryan, M. 2000. *Method and system for UNDOing edits with selected portion of electronic documents*. PN: 6.108.668 US.

Berlage, T; Genau, A. 1993. *From Undo to Multi-User Applications*. German National Research Center for Computer Science.

Berlage. T. 1994. *A selective UNDO Mechanism for Graphical User Interfaces Based On command Objects*. German National Research Center for Computer Science.

Bevan, N. 2009. International Standards for Usability Should Be More Widely Used. Journal of Usability Studies. Vol.4 Issue 3, May 2009, pp. 106-113

Bin, W; Yuan, H; Xi, L; Min, X. 2009. *Open Identity Management Framework for SaaS Ecosystem*. ICEBE '09. IEEE International Conference on e-Business Engineering, pp. 512-517. ISBN: 978-0-7695-3842-6. DOI: 10.1109/ICEBE.2009.82.

Blythe, J.; Deelman, E; Gil, Y; Vahi, K; Mandal, A; Kennedy, K.. 2005. *Resource Allocation Strategies for Workflows in Grids*. In IEEE International Symposium on Cluster Computing and the Grid CCGrid.

Boniface, M.; Nasser, B.; Papay, J.; Phillips, S.; Servin, A.; Yang, X.; Zlatev, Z.; Gogouvitis, S.; Katsaros, G.; Konstanteli, K.; Kousiouris, G.; Menychtas, A.; Kyriazis, D. 2010. *Platform-as-a-Service Architecture for Real-Time Quality of Service Management in Clouds*. Internet and Web Applications and Services (ICIW), 2010 Fifth International Conference on, pp. 155-160. DOI: 10.1109/ICIW.2010.91.

Brown, A; Patterson, D, 2003. *Undo for Operators: Building an Undoable E-mail Store*. University of California, Berkeley. EECS Computer Science Division.

Burke, S. 2007. *UNDO infrastructure*. PN: 7.207.034 US.

Buschmann, F; Meunier, R; Rohnert, H; Sommerlad, P; Stal, M. 1996. *Pattern-Oriented Software Architecture: A System of Patterns*. John Wiley & Sons.

Byun, E. 2008. *Efficient Resource Capacity Estimate of Workflow Applications for Provisioning Resources*. University of Southern California, Technical Report 08-898.

Byun, E. 2011. *Cost optimized provisioning of elastic resources for application workflows*. Future Generation Computer Systems, 2011.

Cai, H.; Zhang, K.; Jun Zhou, M.; Gong, W.; Cai, J.; Mao, X. 2009. *An End-to-End Methodology and Toolkit for Fine Granularity SaaS-ization*. Cloud Computing, 2009. CLOUD

'09. IEEE International Conference on, pp. 101-108. ISBN: 978-1-4244-5199-9. DOI: 10.1109/CLOUD.2009.63.

Caraman, M.C. ; Moraru, S.A. ; Dan, S. ; Grama, C. 2012. *Continuous Disaster Tolerance in the IaaS Clouds.* 13th International Conference on Optimization of Electrical and Electronic Equipment OPTIM 2012, pp. 1226-1232. ISBN: 978-1-4673-1650-7. DOI: 10.1109/OPTIM.2012.6231987.

Cascading. 2012. http://www.cascading.org/. Página Válida al 2012.10.05

CCUCDG. 2010. Cloud Computing Use Cases in http://www.scribd.com/doc/18172802/Cloud-Computing-Use-Cases-Whitepaper, Página Válida al 2012-12-05.

Chen, D; Sun, C. 2001. *Undoing Any Operation in Collaborative Graphics Editing Systems*. School of Computing and Information Technology, Griffith University Australia.

Craig Wood, K. 2010. *Personal Blog*. http://www.katescomment.com/iaas-paas-saas-definition/. Página Válida al 2012-09-01.

Cusumano, M.A. 2008. *The Changing Software Business: Moving from Products to Services*. IEEE Computer, 41(1): 20-27. ISSN: 0018-9162. DOI: 10.1109/MC.2008.29.

Cysneiros, L.; Kushniruk, A. 2003. *Bringing Usability to the Early Stages of Software Development*. RE '03 Proceedings of the 11th IEEE International Conference on Requirements Engineering. ISBN: 0-7695-1980-6.

Cysneiros, L.; Werneck, V.; Kushniruk, A. 2005. *Reusable Knowledge for Satisficing Usability Requirements*. 13th Int'l Conf. Requirements Engineering.

Dix, A; Mancini, R; Levialdi, S. 1997. *The cube – extending systems for undo*. School of Computing, Staffordshire University. UK.

DMTF, 2010. *Common Information Model CIM Infrastructure Specification, Version 2.6,* Distributed Management Task Force DMTF, 2010. [Online]. http://dmtf.org/sites/default/files/standards/documents/DSP0004_2.6.0_0.pdf. Página Válida al 2012/10/05.

Dong, F; Akl, S. 2007. *PFAS: A Resource-Performance-Fluctuation-Aware Workflow Scheduling Algorithm for Grid Computing*. In Proceedings of IPDPS.

Edwards, W: Igarashi, T; La Marca, Anthony; Mynatt, E. 2000. *A Temporal Model for Multi-Level Undo and Redo.* Proceedings of the 13th annual ACM symposium on User interface software and technology, pp. 31-40. ACM Press. ISBN1-58113-212-3.

Edwards, W; Mynatt, E. 1998. *Timewarp: Techniques for Autonomous Collaboration.* Xerox Palo Alto Research Center.

Engine Yard. 2012. *Engine Yard:* http://www.engineyard.com/paas-vs-iaas. Página Válida al 2012-09-01.

Fayad, M.; Shumidt, D. 1997. *Object Oriented Application Frameworks*. Comunications of the ACM, 4010: 32-38.

Ferraiolo, D.; Barkley, J.; Kuhn, D. 1999. *A role-based access control model and reference implementation within a corporate intranet*. ACM Transactions on Information and System Security (TISSEC) - Special issue on role-based access control TISSEC Homepage archive. Volume 2 Issue 1, pp. 34-64, Feb. 1999. doi:10.1145/300830.300834.

Ferre, X., Juristo, N., Moreno, A., Sanchez, I. 2003. *A Software Architectural View of Usability Patterns*. 2nd Workshop on Software and Usability Cross-Pollination at INTERACT'03 Zurich Switzerland.

Ferre, X; Juristo, N; and Moreno, A. 2004. *Framework for Integrating Usability Practices into the Software Process*. Universidad Politécnica de Madrid.

Gamma, E., R. Helm, R. Johnson, and J. Vlissides. 1994. *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison- Wesley.

Gold, N.; Mohan, A.; Knight, C.; Munro, M. 2004. *Understanding service-oriented software*. IEEE Software, 21(2): 71-77. ISSN: 0740-7459. DOI: 10.1109/MS.2004.1270766.

Goncalves, V.; Ballon, P. 2009. *An exploratory analysis of Software as a Service and Platform as a Service models for mobile operators*. International Conference on Intelligence in Next Generation Networks - ICIN. DOI: 10.1109/ICIN.2009.5357056.

Granollers, T; Lorés Vidal, J; Cañas Delgado, J. 2005. Diseño de sistemas interactivos centrados en el usuario. Editorial UOC. ISBM 84-9788-320-9.

Guo, C.; Sun, W.; Huang, Y.; Wang, Z.; Gao, B. 2007. *A Framework for Native Multi-Tenancy Application Development and Management*. The 9th IEEE International Conference on E-Commerce Technology and the 4th IEEE International Conference on Enterprise Computing, E-Commerce, and E-Services. CEC/EEE 2007, pp. 551-558. ISBN: 0-7695-2913-5. DOI: 10.1109/CEC-EEE.2007.4.

Hansen, R.; Probst, C.; Nielson, F. 2006. *Sandboxing in myKlaim. Availability, Reliability and Security*, 2006. ARES 2006. The First International Conference on. DOI: 10.1109/ARES.2006.115.

Hibernate, 2012. *Hibernate framework*. http://www.hibernate.org/. Página Válida: 2012/12/05.

ISO 9241-210:2010. *Ergonomics of human-system interaction -- Part 210: Human-centred design for interactive systems*, 2010. [Online]. http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=52075. Página Válida al 2012/10/05.

Juristo, N.; Moreno, A.; Sanchez-Segura, M. 2007. *Guidelines for eliciting usability functionalities*, IEEE Transactions on Software Engineering, vol. 33, no. 11, pp. 744–758.

Juristo, N; Lopez, M; Moreno, A; Sanchez, M. 2003. *Improving software usability through architectural patterns*. ICSE'03 - International Conference on Software Engineering.

Juristo, N; Moreno, A; Sanchez-Segura, M; Davis, A. 2005. *Gathering Usability Information through Elicitation Patterns*. Grupo de Investigación en Ingeniería de Software Empirica. Universidad Politécnica de Madrid. http://www.grise.upm.es/docs/Gathering_Usability_Information.pdf. Pagina Valida al 2012-10-07.

Keane, P. and Mitchell, K. 1996. *Method of and system for providing application programs with an UNDO/redo function*. PN: 5.481.710 US.

Kent, S.; Lynn, C.; Seo, K. 2000. Secure Border Gateway Protocol (S-BGP). Selected Areas in Communications, IEEE Journal on. Volume: 18 , Issue: 4, pp. 582-592. DOI: 10.1109/49.839934.

Kolb, P. 2009. *EAI Patterns as Software as a Service SaaS*. University of Stuttgart, Faculty of Computer Science, Electrical Engineering, and Information Technology, Diploma Thesis No. 2851.

Korenshtein, R. 2003. *Selective UNDO*. PN: 6.523.134 US.

Laplante, P.; Zhang, J.; Voas, J. 2008. *What's in a Name? Distinguishing between SaaS and SOA*. IT Professional. Volume: 10, Issue: 3, pp. 46-50. ISSN: 1520-9202. DOI: 10.1109/MITP.2008.60.

Lauton, G. 2008. *Developing Software Online With Platform-as-a-Service Technology*. Computer Magazine. Volume: 41, Issue: 6, pp. 13-15. DOI: 10.1109/MC.2008.185. 2008-07.

Lawton, G. 2008. Developing Software Online With Platform-as-a-Service Technology. Computer. June 2008. Volume: 41 , Issue: 6, pp. 13-15. DOI: 10.1109/MC.2008.185.

Lee, B.; Yan, S.; Ma, D.; Zhao, G. . 2011. *Aggregating IaaS Service*. SRII Global Conference SRII, 2011 Annual, pp. 335-338 . ISBN: 978-1-61284-415-2. DOI: 10.1109/SRII.2011.44.

Lee, J.; Hur, S. 2011. *Level 2 SaaS platform and platform management framework*. Advanced Communication Technology ICACT, 2011 13th International Conference on, pp. 1177-1180. ISBN: 978-1-4244-8830-8.

Lee, W.; Choi, M. 2012. *A Multi-tenant Web Application Framework for SaaS*. Cloud Computing CLOUD, 2012 IEEE 5th International Conference on, pp. 970-971. ISBN: 978-1-4673-2892-0. DOI: 10.1109/CLOUD.2012.27.

Li, C. 2006. *UNDO/redo algorithm for a computer program*. PN: 7.003.695 US.

Liao, H. 2009. *Design of SaaS-Based Software Architecture*. International Conference on New Trends in Information and Service Science, 2009. NISS '09, pp. 277-281. ISBN: 978-0-7695-3687-3. DOI: 10.1109/NISS.2009.46.

Linux. 2012. *Linux*: http://www.linux.org/. Página válida al 2012-12-20

Liu, G. 2010. *Research on independent SaaS platform*. Information Management and Engineering ICIME, 2010 The 2nd IEEE International Conference on, pp. 110-113. ISBN: 978-1-4244-5263-7. DOI: 10.1109/ICIME.2010.5477498.

Liyang, T.; Zhiwei, N.; Zhangjun, W.; Li, W. 2011. *A Conceptual Framework for Business Intelligence as a Service SaaS BI*. Intelligent Computation Technology and Automation ICICTA, 2011 International Conference on. Volume: 2, pp. 1025-1028. ISBN: 978-1-61284-289-9. DOI: 10.1109/ICICTA.2011.541.

Loutas, N.; Kamateri, E.; Tarabanis, K. 2011. A Semantic Interoperability Framework for Cloud Platform as a Service. Cloud Computing Technology and Science (CloudCom), 2011 IEEE Third International Conference on, pp. 280-287. DOI: 10.1109/CloudCom.2011.45.

Ludscher, B.; Altintas, I.; Berkley, C.; Higgins, D.; Jaeger, E.; Jones, M.; Lee, E.; Tao, J.; Zhao, Y. 2006. *Scientific Workflow Management and the Kepler System*, In: Concurrency and Computation: Practice and Experience, vol. 18, pp. 1039–1065.

Majithia, S., Shields, M., Taylor, I., Wang, I.: Triana, A. 2004. *Graphical Web Service Composition and Execution Toolkit* In: Proc. IEEE Intl. Conf. Web Services ICWS, pp. 514–524.

Mancini, R., Dix, A., Levialdi, S. 1996. *Reflections on UNDO*. University of Rome. Technical Report RR9611, 1996.

Mandal, A.; Kennedy, K; Koelbel, C; Marin, G.; Mellor-Crummey, J; Liu, B.; Johnsson, L. 2005. *Scheduling Strategies for Mapping Application Workflows onto the Grid*. In IEEE International Symposium on High Performance Distributed Computing HPDC 2005.

Martinez, A.; Rhan, M. 2000. *Graphical UNDO/redo manager and method*. PN: 6.111.575 US.

Mehta, S; Mulik, S. 2009. CLOUD'09. *IEEE International Conference on Cloud Computing*. ISBN: 978-1-4244-5199-9.

Meier, J. 2010. *Personal Blog*. http://blogs.msdn.com/b/jmeier/archive/2010/02/11/software-as-a-service-saas-platform-as-a-service-paas-and-infrastructure-as-a-service-iaas.aspx. Página Válida al 2012-09-01.

Melendy, B. 2012. *Blog*. http://www.tech-faq.com/saas.html. Pagina Valida al 2012-09-01.

Meshorer, T. 1998. *Add an undo/redo function to you Java app with Swing*. JavaWord, June, IDG Communications.

Mitchell, D. 2008. Defining *Platform-As-A-Service, or PaaS*. Fromhttp://blogs.bungeeconnect.com/ 2008/02/18/defining-platform-as-aservice-or-paas/.

Montero, R. 2012. *Building IaaS Clouds and the Art of Virtual Machine Management*. High Performance Computing and Simulation HPCS, 2012 International Conference on. Pages: 573. ISBN: 978-1-4673-2359-8. DOI: 10.1109/HPCSim.2012.6266975.

MS-DOS. 2012. *MS-DOS*: http://www.microsoft.com/resources/documentation/windows/xp/all/proddocs/en-us/windows_dos_overview.mspx?mfr=true. Página válida al 2012-12-20.

Mutavdzic, R. 2012. *Decision framework for building platform as a service (PaaS) based government services*. MIPRO, 2012 Proceedings of the 35th International Convention, pp. 1655-1660. ISBN: 978-1-4673-2577-6. 2012.05.21.

Nakajima, S. and Wash, B. 1997. *Multiple levels UNDO/redo mechanism*. PN: 5.659.747 US.

NIST. 2011. *The NIST Definition of Cloud* Computing. Recommendations of the National Institute of Standards and Technology. Special Publication 800-145. http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf. Página Válida al 2012/12/05

Nurmi, D. 2009. *The Eucalyptus Open-Source Cloud-Computing System*. Cluster Computing and the Grid, 2009. CCGRID '09. 9th IEEE/ACM International Symposium on, pp. 124-131. DOI: 10.1109/CCGRID.2009.93.

O´Brain, J; Shapiro, M. 2004. *Undo for anyone, anywhere, anytime*. Microsoft Research.

OCCIWG, 2012. *Open Cloud Computing Interface Specification*. [Online]. Available: http://www.occi-wg.org. Página Válida al 2012/10/05.

Oinn, T., Greenwood, M., Addis, M.J., Alpdemir, M.N., Ferris, J., Glover, K., Goble, C., Goderis, A., Hull, D., Marvin, D.J., Li, P., Lord, P., Pocock, M.R., Senger, M., Stevens, R., Wipat, A., Wroe, C. 2002. *Taverna: Lessons in Creating a Work*$_7$ *ow Environment for the Life Sciences*, J. Concurrency and Computation: Practice and Experience 18, pp. 1067–1100

OpenStack. 2012. *OpenStack*. http://openstack.org. Página Válida al 2012-12.05

Pallis, G. 2010. *Cloud Computing: The New Frontier of Internet Computing*. Internet Computing, IEEE. Volume: 14, Issue: 5, pp. 70-73. ISSN: 1089-7801. DOI: 10.1109/MIC.2010.113. 2010.

Pathirage, M.; Perera, S.; Kumara, I.; Weerawarana, S. 2011. *A Multi-tenant Architecture for Business Process Executions*. IEEE International Conference on Web Services ICWS, 2011, pp. 121-128. ISBN: 978-1-4577-0842-8. DOI: 10.1109/ICWS.2011.99.

Pervez, Z.; Khattak, A.M.; Sungyoung Lee; Young-Koo Lee. 2010. *Dual Validation Framework for Multi-Tenant SaaS Architecture*. Future Information Technology FutureTech, 2010 5th International Conference on, pp. 1-5. ISBN: 978-1-4244-6948-2. DOI: 10.1109/FUTURETECH.2010.5482743.

Qin, X. y Sun, C. 2001. *Efficient Recovery algorithm in Real-Time and Fault-Tolerant Collaborative Editing Systems*. School of computing and Information Technology Griffith University Australia.

Ressel, M.; Nitsche-Ruhland, R. and Gunzenhauser, R. 1996. *An Integrating, Transformation-Oriented Approach to Concurrency Control and Undo in Group Editors*. CSCW '96 Proceedings of the 1996 ACM conference on Computer supported cooperative work, pp. 288-297. DOI: 10.1145/240080.240305.

Rimal, B.; El-Refaey, M. 2010. *A Framework of Scientific Workflow Management Systems for Multi-tenant Cloud Orchestration Environment*. 19th IEEE International Workshop on Enabling Technologies: Infrastructures for Collaborative Enterprises WETICE 2010, pp. 88-93. ISBN: 978-1-4244-7216-1. DOI: 10.1109/WETICE.2010.20.

Roder, H. 2012. *Specifying usability features with patterns and templates*. Usability and Accessibility Focused Requirements Engineering UsARE, First International Workshop on, pp. 6-11. Publicación de la conferencia.

Sakellariou, R.; Zhao, H. 2004. *A Low-Cost Rescheduling Policy for efficient mapping of Workflows on Grid Systems*. Scientific Programming, vol. 12.

SCO. 2012. *SCO*: http://www.sco.com/. Página válida al 2012-12-20

Shi, Y.; Luan, S.; Li, Q.; Wang, H. 2009. *A Flexible Business Process Customization Framework for SaaS*. Information Engineering, 2009. ICIE '09. WASE International Conference on. Volume: 2, pp. 350-353. ISBN: 978-0-7695-3679-8. DOI: 10.1109/ICIE.2009.226.

Shinnar, A; Tarditi, D; Plesko, M; Steensgaard, B. 2004. *Integrating support for undo with exception handling*. Microsoft Research.

Shixing, Y.; Lee, D.; Singhal, S. 2010. *A Model-Based Proxy for Unified IaaS Management*. Systems and Virtualization Management SVM, 2010 4th International DMTF Academic Alliance Workshop on, pp. 15–20. ISBN: 978-1-4244-9181-0. DOI: 10.1109/SVM.2010.5674747.

Signore,R.; Creamer, J.; Stegman, M. 1995. *The Odbc Solution: Open Database Connectivity in Distributed Environment*s. Mcgraw-Hill. ISBN-13: 978-0079118806.

Spring, 2012. *Spring framework*. http://www.springsource.org/. Page Valid at 2012/12/05.

Sun, C. 2000. *Undo any operation at time in group editors*. School of Computing and Information Technology, Griffith University Australia.

Sun, W.; Zhang, X.; Guo, C.; Sun, P.; Su, H. 2008. *Software as a Service: Configuration and Customization Perspectives*. Congress on Services Part II, 2008. SERVICES-2. IEEE, pp. 18-25. ISBN: 978-0-7695-3313-1. DOI: 10.1109/SERVICES-2.2008.29.

Tang, K.; Zhang, J.; Jiang, Z.. 2010. *Framework for SaaS Management Platform*. ICEBE, 2010 IEEE 7th International Conference on e-Business Engineering, pp. 345-350. ISBN: 978-1-4244-8386-0. Digital Object Identifier : 10.1109/ICEBE.2010.79.

Topcuoglu, H.; Hariri, S.; Min-You Wu. 2002. *Performance-Effective and Low-Complexity Task Scheduling for Heterogeneous Computing*. IEEE Transaction on Parallel and Distributed Systems, Volume: 13, Issue: 3, pp. 260-274. DOI: 10.1109/71.993206.

Torbacki, W. 2008. *SaaS - direction of technology development in ERP/MRP systems*. Archives of Materials Science and Engineering. Volume 32 Issue 1 July 2008, pp. 57-60.

TraceOne 2012. *TraceOne*. http://www.traceone.com/es/servicios/software-as-a-service/ventajas-del-saas.html. Página Válida al 2012-09-01.

Tsai, W.; Huang, Y.; Shao, Q. 2011. *EasySaaS: A SaaS development framework*. Service-Oriented Computing and Applications SOCA, 2011 IEEE International Conference on. Pages: 1-4. ISBN: 978-1-4673-0318-7. DOI: 10.1109/SOCA.2011.6166262

Tsai, W.; Shao, Q.; Li, W. 2010. *OIC: Ontology-based intelligent customization framework for SaaS*. Service-Oriented Computing and Applications SOCA, 2010 IEEE

International Conference on, pp. 1-8. ISBN: 978-1-4244-9802-4. DOI: 10.1109/SOCA.2010.5707139.

Voith, T. ; Oberle, K. ; Stein, M. ; Oliveros, E. ; Gallizo, G. ; Kübert, R. 2010. *A Path Supervision Framework – a key for service monitoring in Infrastructure as a Service IaaS Platforms*. Software Engineering and Advanced Applications SEAA, 2010 36th EUROMICRO Conference on, pp. 127–130. ISBN: 978-1-4244-7901-6. DOI: 10.1109/SEAA.2010.42.

Washizaki, H; Fukazawa, Y. 2002. *Dynamic Hierarchical Undo Facility in a Fine-Grained Component Environment.* Department of Information and Computer Science, Waswda University. Japan.

Wieczorek, M.; Prodan, R.; Fahringer, T. 2005. *Scheduling of Scientific Workflows in the ASKALON Grid Environment*. SIGMOD Record, volume 343, 2005.

Xiao, D.; Hui, M.; Luo, R.; Wang, Q. 2010. *The design and implementation of manual task customization in workflow system based on SaaS*. 3rd IEEE International Conference on Computer Science and Information Technology ICCSIT, 2010, pp. 493-497. ISBN: 978-1-4244-5537-9. DOI: 10.1109/ICCSIT.2010.5563953.

Xu, M.; Xie, F.; Wang, H.; Chen, Z. 2012. *A Novel Workflow Based Data Processing Platform as a Service*. Computing, Communications and Applications Conference, pp. 283–287. ISBN: 978-1-4577-1717-8. DOI: 10.1109/ComComAp.2012. 6154858.

Yang, E.; Zhang, Y.; Wu, L.; Liu, Y.; Liu, S. 2011. *A Hybrid Approach to Placement of Tenants for Service-Based Multi-tenant SaaS Application.* IEEE Asia-Pacific Services Computing Conference APSCC, 2011, pp. 124-130. ISBN 978-1-4673-0206-7. DOI: 10.1109/APSCC.2011.3.

Yang, H; Dasdan, A; Hsiao, R; Parker, D. 2007. Map-reduce-merge: simplified relational data processing on large clusters. SIGMOD '07 Proceedings of the 2007 ACM SIGMOD international conference on Management of data, pp. 1029-1040. DOI: 10.1145/1247480.1247602.

Yang, J; Gu, N; Wu, X. 2004. *A Documento mark Based Method Supporting Group Undo*. Department of Computing and Information Technology. Fudan University, China.

Yu, J.; Buyya, R. 2006. *A Budget Constraint Scheduling of Workflow Applications on Utility Grids using Genetic Algorithms* Proceedings of the Workshop on Workflows in Support of Large-Scale Science WORKS06.

Zhang, K.; Shi, Y.; Li, Q.; Bian, J. 2009. *Data Privacy Preserving Mechanism Based on Tenant Customization for SaaS*. MINES '09. International Conference on Multimedia Information Networking and Security, 2009, pp. 599-603. ISBN: 978-0-7695-3843-3. DOI: 10.1109/MINES.2009.256.

Zhang, Y.; Liu, S.; Meng, X.. 2009. *Towards high level SaaS maturity model: Methods and case study*. IEEE Asia-Pacific Services Computing Conference, 2009. APSCC 2009, pp. 273-278. ISBN 978-1-4244-5338-2. DOI: 10.1109/APSCC.2009.5394111.

Zhao, L.; Liu, J. 2011. *Component-Oriented SaaS Integration Framework Research Based on OFBiz*. International Conference on Management and Service Science MASS, 2011, pp. 1-3. ISBN 978-1-4244-6579-8. DOI: 10.1109/ICMSS.2011.5998588.

Zheng, Y.; Pang, J.; Li, J.; Cui, L. 2012. *Business Process Oriented Platform-as-a-Service Framework for Process Instances Intensive Applications*. Parallel and Distributed Processing Symposium Workshops & PhD Forum (IPDPSW), 2012 IEEE 26th International, pp. 2320-2327. DOI: 10.1109/IPDPSW.2012.284.

Zhou, Y.; Liu, X.; Wang, X.; Xue, L.; Liang, X.; Liang, S. 2010. *Business Process Centric Platform-as-a-Service Model and Technologies for Cloud Enabled Industry Solutions*. Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on, pp. 534- 537. DOI: 10.1109/CLOUD.2010.52.