UNIVERSIDAD CATÓLICA ANDRÉS BELLO
FACULTAD DE INGENIERÍA
ESCUELA DE INFORMÁTICA

Este Jurado; una vez realizado el examen del presente trabajo ha evaluado su contenido con el resultado: **Dieciocho (18).**

JURADO EXAMINADOR

Firma:	Firma:	Firma:
Nombre: Lúcia Cardoso	Nombre Honack Villanueva	Nombre: Gustavo Bonalde

REALIZADO POR	Fernandes Agrela Ana Karina. Miranda Bouzas Carlos Manuel.
TUTOR	Ing. Gustavo Bonalde
FECHA	Febrero 2009

Contenido

Índice de Figuras

Índice de Tablas

Sinopsis

Todo desarrollo de software tiene como objetivo la obtención de un programa, diseñado para cubrir un conjunto de necesidades. Para lograr cumplir exitosamente este objetivo y obtener un producto de calidad, es necesario seguir las etapas del ciclo de vida del desarrollo del software, aplicando una metodología que dictará el orden y las actividades que se deben seguir.

La ingeniería de requisitos ha sido una de las áreas de la ingeniería del software en la que más esfuerzo de investigación se ha realizado durante las últimas décadas, y con motivo porque los errores de comprensión cometidos en esta etapa inicial de los proyectos son los más costosos de resolver. Si no se detectan a tiempo, implican la realización de actividades erróneas durante todas las fases subsiguientes, hasta llegar a las pruebas.

El desarrollo de esta herramienta web como Trabajo Especial de Grado, permite una gestión de los requerimientos para desarrollos de software que se rijan bajo Metodología Open UP, y que le permite al equipo crear y compartir sus requerimientos utilizando métodos familiares basados en documentos, potenciados por la aplicación de las capacidades de una base de datos, tales como la trazabilidad. Los proyectos exitosos comienzan con una buena administración de requerimientos, cuanto más efectiva sea su ejecución, mayor será el resultado en calidad y satisfacción del cliente.

Capítulo I

Planteamiento del Problema

- Visión Preliminar

El siguiente capítulo permite relacionar al lector con los aspectos de identificación del tema, el problema, objetivos a alcanzar, justificación y limitaciones del Trabajo Especial de Grado.

1.1 Planteamiento del Problema.

1.2 Objetivo General y Objetivos Específicos.

1.3 Alcance.

1.4 Limitaciones.

1.5 Justificación.

1.1 Planteamiento del Problema:

Todo desarrollo de software tiene como objetivo la obtención de un programa, diseñado para cubrir un conjunto de necesidades que varían dependiendo del área a la cual será aplicado el mismo. Para lograr cumplir exitosamente este objetivo y obtener un producto de calidad, que satisfaga las necesidades de los clientes es necesario seguir las etapas del ciclo de vida del desarrollo del software, aplicando una metodología que dictará el orden y las actividades que se deben seguir.

Independientemente de la metodología elegida, en el ciclo de desarrollo del software una de las etapas más importantes y clave para lograr un producto exitoso, es la Definición de Requerimientos, acompañado de otras etapas que complementan la obtención de un producto de calidad.

La Definición de Requerimientos es una etapa crucial en el desarrollo de software, porque si los requerimientos no se definen correctamente, no es posible construir soluciones que posean un éxito medible. Es necesario que exista flexibilidad en la definición de requerimientos para prevenir los cambios que puedan ocurrir, si esta flexibilidad no se permite, el software desarrollado no cumplirá con las exigencias deseadas.

En muchos casos no se hace una distinción entre los requerimientos del cliente[1], y los de los usuarios[2]. Los requerimientos de los clientes son necesidades que debe

---

1. **Cliente:** Es la persona que solicita el sistema y es responsable por aprobarlo. Por lo general, los clientes pagan por el desarrollo del sistema
2. **Usuarios:** Son las personas que utilizan el sistema desarrollado.

satisfacer el sistema y los de los usuarios son servicios que debe prestar el sistema en orden de cumplir con dichas necesidades. Al omitir esta distinción se pierde la trazabilidad, que es la técnica utilizada para determinar el origen de los requerimientos, así como la flexibilidad ante requerimientos cambiantes o nuevos haciendo al sistema menos escalable y más complejo para mantener [19].

Otro factor es que los requerimientos originales son almacenados en documentos que se recopilan en un repositorio, pero a medida que cambian nunca son modificados perdiendo el control y obteniendo un producto que no cumple con las necesidades del cliente. Además muchas veces no existe un proceso de control de cambio definido para requerimientos o, si se define, nunca es utilizado, o simplemente no hay forma de verificar si el sistema cumple con todos los requerimientos. La RE[3] (Ingeniería de Requisitos) indica que para mantener madurez y calidad en los requerimientos, éstos deben ser documentados después de que se llega a un consenso con los múltiples interesados del sistema. La técnica para la obtención de requerimientos debe ser utilizada según el tipo de proyecto y los interesados en el mismo. Los repositorios deben ser actualizados a lo largo del ciclo de vida del desarrollo del proyecto y como control final debe desarrollarse y mantenerse una matriz de trazabilidad.

Muchos otros escenarios se pueden mencionar, pero la necesidad en la madurez en RE[3] es necesaria porque es lo correcto, porque nos permite obtener un producto que

---

3. **RE:** Requirement Engineering (Ingeniería de Requisitos).

cumpla con las necesidades de los clientes y porque es la base de las etapas del ciclo de desarrollo de software. Existen muchos modelos para el análisis de madurez en los requerimientos, por ejemplo, el modelo CMMI[4] cuyo principio indica que la calidad de un producto o de un sistema, se deriva en mayor parte como consecuencia de la calidad en los procesos empleados para su desarrollo y mantenimiento [15]. En consecuencia ante la importancia de una gestión eficiente de los requerimientos, en un proceso de desarrollo de software, es necesaria una herramienta que permita controlar y mantener actualizado cualquier modificación en ellos. Para el desarrollo de software que se realiza bajo la metodología Open UP no se dispone de una herramienta que ofrezca una gestión eficiente de requerimientos que además los mantenga actualizados durante todo el ciclo de vida de desarrollo de software. Por lo tanto si no se posee requerimientos bien definidos no será posible desarrollar soluciones exitosas.

4. **CMMI:** Capability Maturity Model Integration

1.2 Objetivos:

1.2.1 Objetivo General:

Desarrollar una herramienta web de código abierto, para el apoyo de la Gestión de Requerimientos en el ciclo de vida del desarrollo de software, que cumpla con las premisas establecidas para el nivel dos (2) CMMI y para proyectos basados en metodología Open UP.

1.2.2 Objetivos Específicos:

- Diseñar e implementar una herramienta web, que permita la gestión de requerimientos.
- Diseñar y construir una base de datos relacional, para centralizar los distintos tipos de requerimientos y sus atributos.
- Diseñar e implementar las interfaces web necesarias para la gestión de requerimientos.
- Desarrollar un módulo de administración para la herramienta web que permita asociar a los proyectos con los usuarios.
- Diseñar y desarrollar un mecanismo para la integración de una herramienta software libre que genere documentos con la herramienta de gestión de requerimientos, reflejando los requerimientos almacenados en la base de datos.

- Diseñar y desarrollar un mecanismo para la integración de una herramienta de código abierto para la gestión de cambios en los requerimientos, con la herramienta de gestión de requerimientos.

- Diseñar y desarrollar un mecanismo para la integración de aplicaciones de Ingeniería Social que apoyen la comunicación, colaboración y coordinación del equipo de trabajo, con la herramienta de gestión de requerimientos.

1.3 Alcance:

Como resultado del Trabajo Especial de Grado el producto final será una herramienta web de código abierto desarrollada en lenguaje PHP, utilizando AJAX, que brindará soporte a la gestión de requerimientos de proyectos de software, permitiendo la colaboración de los miembros del equipo a través del modulo de administración, que se desarrollará basado en roles. La herramienta se conectará a una base de datos relacional, para obtener los requerimientos que han sido definidos, los cuales se encuentran contenidos en la misma. Esta herramienta web de código abierto generará la documentación establecida por la etapa de definición de requerimientos de la metodología Open UP. Además permitirá dar seguimiento a las relaciones entre requerimientos funcionales y no funcionales, necesidades y características detallados en los requisitos del software, manteniendo la trazabilidad entre ellos. Se establece que esta herramienta permita gestionar requerimientos según el nivel dos (2) de madurez establecido por el modelo CMMI, ofreciendo la oportunidad de mantener los requerimientos gestionados y controlados en todo momento a lo largo del desarrollo del software. Adicionalmente se integrará con otra herramienta de software libre para la gestión de cambio sobre los requerimientos, además de integrar aplicaciones de Ingeniería Social que apoye la comunicación, colaboración y coordinación del equipo de trabajo a través de notas o comentarios que se podrán agregar y ser vistos al momento de revisar los requerimientos.

1.4 Limitaciones:

- Se incorporaran únicamente aplicaciones o herramientas que cumplan de estándares abiertos o de software libre.
- Se establece para esta herramienta la utilización de MySQL para la definición de la base de datos.
- La herramienta web no permitirá realizar modelado UML.
- El usuario debe ser capaz de indicar a la herramienta que es una necesidad y que es un requerimiento.
- Las pruebas que se le realizarán a la herramienta, serán realizadas con un caso de estudio específico definido por los desarrolladores.
- Las aplicaciones de Ingeniería Social serán notas o comentarios sobre los requerimientos definidos.

1.5 Justificación:

La ingeniería de requisitos se refiere al análisis, a la especificación, y a la validación de los requisitos del software. Está extensamente reconocido dentro de la industria del software que los proyectos de la ingeniería de software son críticamente vulnerables cuando estas actividades se realizan mal.

La ingeniería de requisitos ha sido una de las áreas de la ingeniería del software en la que más esfuerzo de investigación se ha realizado durante las últimas décadas, y con motivo, porque los errores de comprensión cometidos en esta etapa inicial de los proyectos son los más costosos de resolver. Si no se detectan a tiempo, implican la realización de actividades erróneas durante todas las fases subsiguientes, hasta llegar a las pruebas. Momento en el que, a la vista de los defectos detectados en la ejecución de los casos de prueba, se concluye que la repetición de las actividades erróneas puede ser una manera de resolver la situación.

Como es bien conocido, el desarrollo de software es una tarea de equipo, es crítico que los miembros del equipo que van a implementar la solución comprendan los objetivos y entregables apropiados. Ante esta situación surge la siguiente interrogante ¿Cómo puede un equipo entregar la solución adecuada, si sus miembros no tienen acceso a la visión del proyecto, sus metas, especificaciones y otros requerimientos que detallan lo que la solución final debe hacer?

Los proyectos exitosos comienzan con Ingeniería de Requerimientos. Cuanto

mejor sea la comunicación y administración de requerimientos, mayor será la oportunidad para que los proyectos se entreguen a tiempo. Las herramientas software para la gestión de requisitos se han convertido en la vía más eficiente para soportar la automatización de estos procesos clave de las organizaciones que desarrollan o compran tecnología, de aplicación tanto al desarrollo de software tradicional, como el software crítico. Es determinante que estas herramientas ofrezcan facilidades para adaptarse a los procesos flexibles de especificación de requisitos propios de los sistemas de información [14].

La ingeniería de requisitos es foco de metodologías, estándares y modelos de mejora, por ejemplo CMMI que vinculan las necesidades de los usuarios y partes interesadas. Se busca con esta herramienta mantener a los equipos de proyectos al día gracias a la creación, análisis y gestión de los requisitos de aplicaciones y casos de uso. Se busca que cualquier cambio en tiempo real sea almacenado.

La pieza clave para la definición de los sistemas es el correcto conocimiento de los requerimientos, lo cual se fundamenta en amplio análisis de la solución a implementar. Como se ha mencionado el desarrollo de software es una tarea de equipo, de tal forma, es crítico que todos los miembros del equipo posean un entendimiento compartido de la visión de sus proyectos, metas, especificaciones y requerimientos.

El desarrollo de esta herramienta web que se propone como trabajo especial de grado, permitirá una gestión de los requerimientos para desarrollos de software que se rijan bajo Metodología Open UP, y que le permite al equipo crear y compartir sus requerimientos utilizando métodos familiares basados en documentos, potenciados por la aplicación de las capacidades de una base de datos, tales como la trazabilidad. Los proyectos exitosos comienzan con una buena administración de requerimientos, cuanto más efectiva sea su ejecución, mayor será el resultado en calidad y satisfacción del cliente.

Capítulo II

Marco Referencial

• Visión Preliminar

2.1 *Ciclo de Vida de Desarrollo del Software.*

2.2 *Objetivos de las fases del ciclo de vida del desarrollo del Software.*

2.3 *Gestión de Requerimientos:*

- 2.3.1 *Pirámide de Requerimientos.*
- 2.3.2 *Trazabilidad entre Requerimientos.*
- 2.3.3 *Características de buenos Requerimientos.*
- 2.3.4 *Visión general del proceso de gestión de Requerimientos.*

2.4 *Modelo de Madurez de capacidades (CMM) e Integración del modelo de madurez de capacidades (CMMI).*

- 2.4.1 *CMMI – Gestión de Requerimientos (REQM).*
- 2.4.2 *Prácticas genéricas de CMMI.*

2.5 *Metodología OpenUP*

- 2.5.1 *Principios básicos de OpenUP*
- 2.5.2 *Roles de OpenUP*
- 2.5.3 *Productos de trabajo de OpenUP*

2.6 *Herramientas CASE*

- 2.6.1 *Clasificación de Herramientas CASE*

Una visión general del proceso de gestión de requerimientos, contiene los siguientes grandes pasos:

- Establecer un plan para la gestión de requerimientos
- Obtener requerimientos
- Desarrollar el documento de Visión
- Crear Casos de Uso
- Diseño del Sistema.

La gestión de Requerimientos es un proceso interactivo, ya que estando en cualquiera de los pasos podemos retroceder y repetir una actividad, para obtener un nuevo requerimiento o esclarecer cualquier duda. [19]

2.1 Ciclo de Vida de Desarrollo de Software [19]:

Cuando se menciona el término ciclo de vida del software[5], se describe el desarrollo de software, desde la fase[6] inicial hasta la fase final. El propósito es definir las distintas fases intermedias, que se requieren para validar el desarrollo de un sistema, es decir, para garantizar que el software cumpla los requisitos y verificar los procedimientos de desarrollo, cerciorando que los métodos utilizados son apropiados. El ciclo de vida permite que los errores se detecten lo antes posible y por lo tanto, permite a los desarrolladores concentrarse en la calidad[7] del software y en los plazos de implementación. Un ciclo de vida para un proyecto se compone de fases sucesivas compuestas por tareas planificables. Sin embargo, la forma de agrupar las actividades, los objetivos de cada fase, los tipos de productos intermedios que se generan, entre otros, pueden ser muy diferentes dependiendo del tipo de producto o proceso a generar y de las tecnologías empleadas.

2.2 Objetivos de las fases del ciclo de vida de desarrollo del software [19]:

Dentro de cada fase general, se pueden establecer una serie de objetivos:

Fase de definición (análisis): Se construye un modelo de los requisitos.

---

5. **Software:** Es el conjunto de los programas de cómputo, procedimientos, reglas, documentación y datos asociados que forman parte de las operaciones de un sistema de computación.

6. **Fases:** Una fase es un conjunto de actividades relacionadas con un objetivo en el desarrollo del proyecto. Se construye agrupando tareas (actividades elementales) que pueden compartir un tramo determinado del tiempo de vida de un proyecto [19].

7. **Calidad:** La norma ISO 9000:2000 la define como: La capacidad de un conjunto de características intrínsecas para satisfacer requisitos [10].

Fase de diseño: A partir del modelo de análisis se deducen las estructuras de datos, la estructura en la que se descompone el sistema y la interfaz de usuario.

Fase de construcción (codificación): Se construye el sistema. La salida de esta fase es el código ejecutable.

Fase de pruebas y mantenimiento: Se comprueba que se cumplen criterios de corrección, calidad, asegurando que el sistema siga funcionando y adaptándose a nuevos requisitos.

2.3 Gestión de Requerimientos:

Un requerimiento se define como *"una condición o capacidad que un sistema debe cumplir"* [19]. Puede ser cualquiera de las siguientes:

• Una capacidad que debe cumplirse, o que posee un sistema para satisfacer un contrato, un estándar, una especificación u otro documento formalmente impuesto.

• Una restricción impuesta por una de las partes interesadas[8].

En la etapa de Definición de Requerimientos se obtiene una descripción de los requisitos del sistema (es decir, las condiciones o capacidades que el sistema debe cumplir) suficientemente buena como para que pueda llegarse a un acuerdo entre el

---

8. **Interesados (partes interesadas):** Se define como alguien que se afectado por el sistema que se está desarrollando.

cliente (incluyendo a los usuarios) y los desarrolladores, sobre qué debe y qué no debe hacer el sistema. El flujo de trabajo de la etapa de Requisitos consta de las siguientes etapas [11]:

- Analizar el problema.
- Análisis de las necesidades de los implicados en el proceso de desarrollo.
- Definir el sistema.
- Gestionar el ámbito del sistema.
- Evaluación.
- Gestión de requisitos.

Algunos de los productos de desarrollo del software fundamentales que se producen en la etapa de Requisitos son:

- Especificación de Requisitos, que puede ser Documento de Visión y Glosario de términos.
- Modelo de Casos de Uso, que incluye Especificación de Casos de Uso, descripción de actores, diagrama e informe del modelo de casos de uso.
- Arquitectura Inicial.
- Documento de cambios.
- Informe de evaluación.

2.3.1 Pirámide de Requerimientos:

Dependiendo del formato, fuente y características comunes, los requerimientos se pueden dividir en diferentes tipos [19]:

- ***Participación de necesidad:*** la exigencia de una parte interesada.
- ***Característica:*** un servicio prestado por el sistema, un propósito de una característica es cumplir una necesidad de interesados.
- ***Caso de Uso:*** Una descripción del comportamiento del sistema en términos de secuencias de acciones.
- ***Requerimientos no Funcionales:*** otro requisito (por lo general no funcional) que no puede ser capturados en casos de uso.
- ***Casos de prueba***: una especificación de prueba de los insumos, las condiciones de ejecución y los resultados esperados.
- ***Escenario:*** una secuencia específica de acciones, con una ruta específica a través de un caso de uso.

La pirámide de requerimientos agrupa en el nivel superior son las necesidades de los interesados. En los niveles más bajos son las características, casos de uso, y los requerimientos suplementarios. La principal diferencia entre las necesidades y características está en la fuente de la obligación, es decir, las necesidades provienen de las partes interesadas y las características son formuladas por los analistas del negocio. El papel de casos de prueba consiste en comprobar si el uso de los requerimientos

(funcionales y no funcionales) se aplica correctamente. Los escenarios permiten derivar casos de uso desde casos de prueba, facilitan el diseño y la implementación de determinadas partes a través de casos de uso.

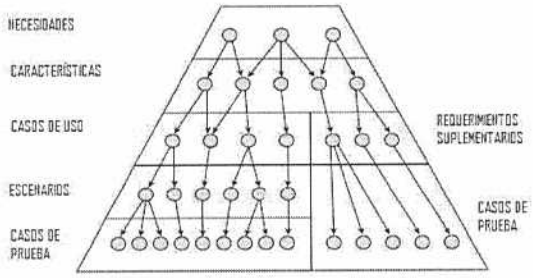


Figura 1. Pirámide de Requerimientos [19].

2.3.2 Trazabilidad entre Requerimientos:

La trazabilidad es una técnica que proporciona una relación entre los distintos niveles de requerimientos en el sistema. Esta técnica le ayuda a determinar el origen de cualquier requisito. En la Figura 1, se ilustra cómo es la trazabilidad de los requerimientos desde el nivel superior hasta el inferior. Cada necesidad normalmente es mapeada con varias características, a su vez las características son relacionadas con varios requerimientos, que también se relacionan con varios casos de uso.

La trazabilidad desempeña varias funciones importantes:

- **Comprobación de que una aplicación cumple con todos los requisitos:** Todo lo que el cliente pidió se puso en práctica.

- **Verificar que la aplicación solo hace lo que se solicitó:** No aplicar algo que el cliente nunca solicitó.

• **Ayudar con la gestión del cambio:** Cuando algunos de los requisitos fueron modificados, se quiere conocer que hubo cambios y cuales son para realizar los nuevos casos de prueba.

La trazabilidad es un elemento del proyecto que debe rastrearse a partir de otro elemento. Nos permite identificar la relación Característica – Requerimientos – Casos de Uso; cuales han sido los cambios suscitados a lo largo del proyecto.

2.3.3 Características de buenos Requerimientos:

Los buenos requerimientos deben tener las siguientes características [19]:

- **Inequívoco:** Debe existir una sola manera de interpretar el requerimiento.
- **Comprobable:** Se debe verificar que la exigencia del requerimiento se aplica correctamente. Existen algunas palabras que hacen difícil que un requerimiento sea comprobable, algunas de ellas son:
  - Adjetivos: Robusto, seguro, eficaz, eficiente, ampliable, flexible, mantenible, fiable, fácil de usar, adecuado.
  - Adverbios: Rápido, seguro, de forma oportuna.
  - Acrónimos: etc., y / o, PD.
- **Conciso:** Los requerimientos no deben tener palabras o información innecesaria.
- **Corregible:** Se deben plantear los requerimientos de forma que se puedan modificar fácilmente.

- **Compresible:** Deben ser redactados de forma coherente y gramáticamente correctos.
- **Factible:** Los requerimientos deben ser factibles (posibles) dentro de las limitaciones existentes tales como: tiempo, dinero y recursos.
- **Consistente:** No debe existir conflictos entre los requerimientos. Por ejemplo:

  Formato de fechas (dd/mm/aaaa – mm/dd/aaaa).

- **Independiente:** Un requerimiento no debe depender de otro requerimiento.
- **Atómico:** La exigencia del requerimiento debe tener un único elemento trazable.
- **Necesario:** Un requerimiento es innecesario si ninguna de las partes interesadas lo ha solicitado. Un ejemplo, de un requerimiento que no es necesario, es aquel que ha sido añadido asumiendo que los usuarios o el cliente lo quieren.
- **Libre implementación:** Los requerimientos no deben tener información innecesaria sobre su diseño o aplicación. Para el usuario debe ser transparente como se almacena la información.
- **No Redundante:** Cada requerimiento debe ser expresado una sola vez y no debería superponerse con otro. Por ejemplo:

  Req 1: Un calendario debe estar disponible para ayudar a introducir la fecha de nacimiento.

  Req 2: El sistema debe mostrar un pop – up con un calendario para registrar cualquier fecha.

- **Completo:** Un requerimiento debe especificar todas las condiciones que pueden ocurrir.

2.3.4 Visión general del proceso de gestión de Requerimientos:

Una descripción simple del proceso de gestión de requerimientos se resume en los siguientes pasos:

- Establecer un plan de gestión de requerimientos.
- Obtención de requerimientos.
- Desarrollar el documento de Visión[9].
- Crear casos de uso.
- Diseño del Sistema.

A medida que se van ejecutando los pasos se va construyendo la pirámide de requerimientos (Figura 1). En la Tabla 1 se describe los tipos de requerimientos y que documentos se deben crear en cada paso.

Pasos	*Tipo de Requerimiento*	*Documento*
Obtención de Requerimientos	Necesidades	Solicitud de los interesados
Desarrollo del doc. de Visión	Características	Visión
Crear Casos de Uso	Casos de uso, escenarios	Especificación de Casos de Uso
Diseño del Sistema	Diagrama de clases, entre otros	Diagramas UML

Tabla 1. Requerimientos y Documentos creados en cada paso [19].

2.4 Modelo de madurez de capacidades (CMM) e Integración del modelo de madurez de capacidades (CMMI):

El modelo más importante en la actualidad para la evaluación de la madurez de los procesos de desarrollo, es el modelo de madurez de capacidades (CMM, Capability

9. **Documento de Visión:** Es un artefacto y uno de los primeros objetivos de todo proceso de desarrollo del software, se debe explicar el producto deseado, el proceso en sí, y las necesidades que se van a cubrir en el proyecto.

Maturity Model) del Instituto de Ingeniería de Software (SEI). CMM tiene como objetivo evaluar los procesos en sus niveles de madurez e identificar los niveles que una organización debe formar para establecer una cultura de excelencia en la ingeniería de software. Los modelos CMM se generan gracias a la experiencia colectiva de los proyectos más exitosos de software. En particular, CMM es un marco de trabajo que especifica guías para las organizaciones de software que quieren incrementar su capacidad de procesos, considerando los siguientes puntos:

- Identificar fortalezas y debilidades en la organización.
- Ponderar los riesgos de seleccionar entre diferentes contratos y monitorear los mismos.
- Entender las actividades necesarias para planear e implementar los procesos de software.
- Ayudar a definir e implementar procesos de software en la organización a través de una guía.

Los procesos se evalúan mediante distintos niveles de madurez como se muestra en la tabla2 y figura 2. Existe una extensión del modelo básico de madurez, la integración del modelo de madurez de capacidades (CMMI, Capability Maturity Model Integration), el cual tiene como objetivo integrar los diferentes dominios donde se ha aplicado CMM, más allá de sólo el desarrollo de software [18]. CMMI proporciona a las organizaciones elementos esenciales para que posean procesos eficaces. CMMI se puede utilizar para guiar la mejora de todo un proceso, a través de un proyecto, una división o de toda una

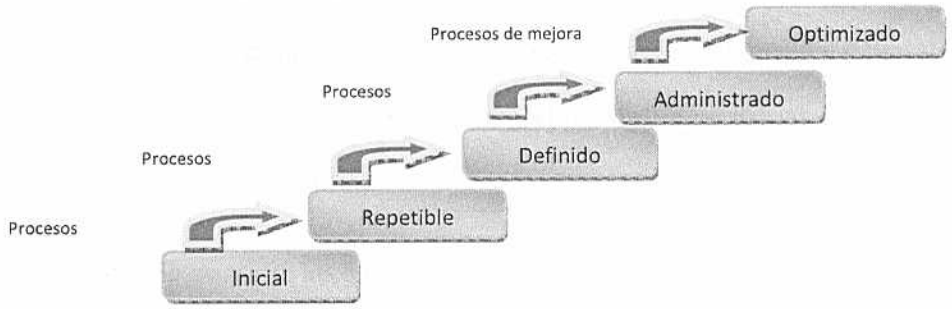


Figura 2. Los cinco niveles de madurez del proceso de software [18].

	Nivel	Características	Transición al siguiente nivel
1	Inicial	Poca formalización, herramientas aplicadas de manera informal al proceso.	Iniciar una administración rigurosa del proyecto y asegurar la calidad.
2	Repetible	Se cuenta con un proceso estable y un nivel repetible de control estadístico.	Establecer un grupo y una arquitectura de proceso de desarrollo de software. Introducir métodos y tecnologías de ingeniería de software.
3	Definido	Se cuenta con una base para un progreso mayor y continuo.	Establecer un conjunto básico de administraciones del proceso para identificar la calidad y costo de los parámetros, y una base de datos del proceso. Juntar y mantener los datos del proceso. Calcular la calidad relativa de cada producto e informar a la administración.
4	Administrado	Mejoras sustanciales en la calidad junto con medidas comprensivas del proceso.	Apoyar la recopilación automática de datos del proceso. Usar los datos para analizar y modificar el proceso.
5	Optimizado	Mejoras con base en mayor calidad y cantidad.	Continuar mejorando y optimizando el proceso.

Tabla 2. Los cinco niveles del modelo CMM [18].

organización. CMMI ayuda a integrar funciones tradicionalmente separadas de una organización, establecer objetivos de mejora en los procesos, prioridades, sirve de orientación en los procesos de calidad y establecer un punto de referencia para evaluar

los procesos actuales. CMMI busca ayudar la mejora de los procesos, recopilando las mejores prácticas, organiza y da prioridad a las actividades. Un punto a destacar es que el modelo CMMI no es un proceso; un modelo CMMI describe las características de eficacia en los procesos [17].

CMMI también mantiene los 5 niveles de capacidad y de madurez [2]:

Nivel	Nivel de Capacidad	Nivel de Madurez
0	Incompleto	N/A
1	Realizado	Inicial
2	Gestionado	Gestionado
3	Definido	Definido
4	Gestionado cuantitativamente	Gestionado cuantitativamente
5	Optimización	Optimización

Tabla 3. Comparación entre los niveles de Capacidad y Madurez. [2].

Más información del módelo CMMI puede verse en el *Apéndice D.2.*

2.4.1 CMMI – Gestión de Requerimientos (REQM) [16]:

El propósito de la gestión de requerimientos, es el manejo de todos los requisitos del proyecto, identificar las incoherencias entre ellos y el plan de trabajo. Se debe manejar todos los requerimientos generados en el proyecto tanto técnicos como no técnicos. CMMI en la gestión de requerimientos (REQM) nos indica lo siguiente:

Los requerimientos son gestionados y las inconsistencias son identificadas.

a. Comprender los requerimientos.

b. Compromiso por parte de los participantes del proyecto.

c. Gestionar los cambios de los requerimientos a lo largo del proyecto.

d. Mantener la trazabilidad bidireccional entre los requerimientos.

e. Identificar inconsistencia entre los requerimientos y el plan de trabajo.

2.4.2 Prácticas genéricas de CMMI [16]:

CMMI proporciona unas prácticas genéricas que pueden ser utilizadas en cualquier área de proceso:

- Establecer y mantener una política de planificación y ejecución de los procesos, dentro de la organización.
- Establecer y mantener un plan para la ejecución de cada proceso.
- Proporcionar los recursos adecuados para el desarrollo de proyectos, procesos y prestación de servicios.
- Entrenar el equipo de trabajo para que apoyen el desarrollo y soporte las necesidades según como sea necesario.
- Asignar niveles de control para productos de trabajo dentro de los procesos.
- Identificar e implicar las partes interesadas en los procesos.
- Monitorear y controlar los procesos de manera que se cumpla lo establecido en el plan de trabajo y tener medidas correctivas.
- Evaluar los procesos por sus objetivos, descripción, estándares y procedimientos.
- Examinar el estado, actividad, resultado de cada proceso con el mayor nivel de gestión y resolver cualquier problema.
- Establecer y mantener una descripción definida para cada proceso.

- Recoger los resultados de los procesos, los productos de trabajo, los resultados de las mediciones, información de apoyo al desarrollo de los productos, actividades de apoyo, todo para un uso futuro dentro de los procesos de la organización. Se debe tomar las mejores prácticas, para ser empleadas en procesos futuros.

2.5 Metodología Open UP [8]:

Open UP es una metodología de desarrollo de software, de código abierto diseñado para pequeños equipos organizados, tomando una *aproximación ágil*[10] del desarrollo. Además es iterativa, *Mínima, Completa, y Extensible*[11].

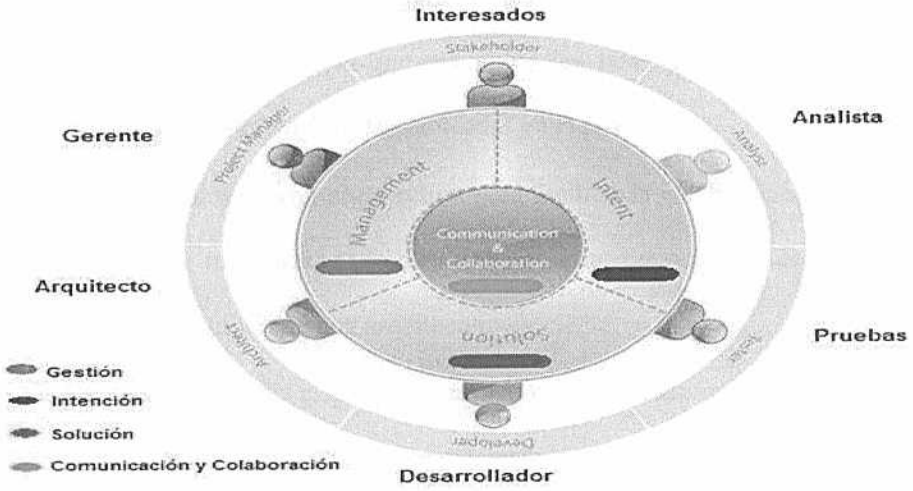


Figura 3. Áreas de Open UP [8].

Se valora la colaboración y el aporte de las partes interesadas sobre los

10. **Aproximación ágil:** En Open UP es un proceso unificado que incorpora técnicas agiles probadas. El resultado es un proceso estructurado, robusto, eficiente y liviano **[8]**.

11. **Mínima, Completa y Extensible:** Es la mínima cantidad de procesos para un equipo pequeño, puede ser usado como está y puede ser extendido, personalizado para propósitos específicos **[8]**.

entregables. Open UP está organizado dentro de cuatro áreas principales de contenido: Comunicación y Colaboración, Intención, Solución y Administración.

2.5.1 Principios básicos de Open UP [8]:

- **Colaboración**: Para alinear los intereses y un entendimiento compartido. Se debe desarrollar prácticas colaborativas que fomenten un ambiente de equipo saludable. Buenas prácticas colaborativas, encuadran los intereses de los participantes del proyecto y les ayuda a desarrollar un entendimiento compartido del proyecto.
- **Balance**: Para confrontar las prioridades (necesidades y costos técnicos). Desarrollar una solución que maximice los beneficios para los interesados y cumpla con las restricciones planteadas en el proyecto.
- **Enfoque:** Articular la arquitectura para facilitar la colaboración técnica, reducir los riesgos, minimizar excesos y trabajo extra. Enunciar las decisiones técnicas esenciales a través de una arquitectura creciente.
- **Evolución**: Dividir el proyecto en iteraciones cortas, enmarcadas en el tiempo para demostrar valor incremental, reducir riesgos[12], demostrar resultados y obtener retroalimentación temprana y continua de los clientes.

---

12. **Riesgo:** Es cualquier cosa que se pueda atravesar en el camino al éxito y es actualmente desconocido o incierto. Usualmente, un riesgo es calificado por la probabilidad de ocurrencia y el impacto en el proyecto, si este ocurre **[8]**.

2.5.2 *Roles de Open UP [8]:*

- **Analista:** Las personas en este rol representan al cliente y los usuarios finales involucrados, obteniendo información desde los interesados para entender el problema a ser resuelto; capturar y ajustar las prioridades para los requerimientos es su función.

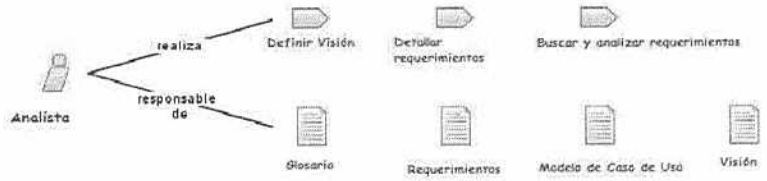


Figura 4. Analista en Open UP [8].

- **Arquitecto:** Este rol es el responsable de diseñar la arquitectura del software, la cual incluye tomar las principales decisiones técnicas que condicionan globalmente el diseño y la implementación del proyecto. Lidera o coordina el diseño técnico del sistema y tiene la responsabilidad general de manejar las principales decisiones técnicas, expresadas en la arquitectura del software. Esto típicamente incluye identificar y documentar los aspectos arquitecturalmente significativos del sistema, incluyendo vistas de requerimientos, diseño, implementación y despliegue. Este rol es también responsable de razonar sobre estas decisiones, equilibrando las preocupaciones de los interesados, reduciendo los riesgos técnicos y asegurándose que las decisiones son comunicadas eficazmente, validadas y acatadas. Además está estrechamente involucrado con el Gerente para conformar y organizar el equipo encargado de la arquitectura y planear el proyecto.

Figura 5. Arquitecto en Open UP [8].

- **Desarrollador:** Las personas en este rol son responsables por desarrollar una parte del sistema, incluyendo el diseño, para que se ajuste a la arquitectura, posiblemente prototipar la interfaz de usuario y entonces implementar, hacer pruebas unitarias e integrar los componentes que son parte de las funciones de éste rol.

Figura 6. Desarrollador en Open UP [8].

- **Gerente:** Lidera la planeación del proyecto, coordina interacciones con los interesados y conserva el equipo enfocado en alcanzar los objetivos del proyecto.

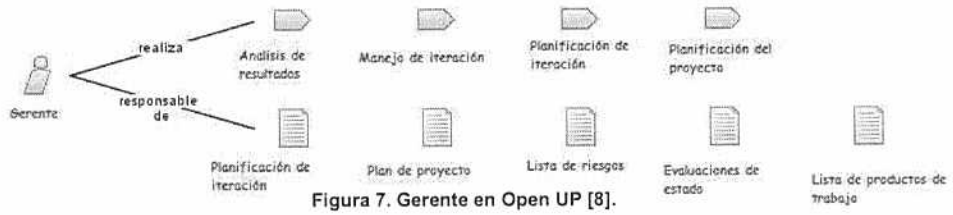


Figura 7. Gerente en Open UP [8].

Este rol:

- Aplica conocimientos de gestión, competencias, herramientas y técnicas a un amplio rango de tareas, con el propósito de alcanzar los resultados deseados para un

proyecto en un tiempo oportuno.

- Es responsable del resultado del proyecto y la aceptación del producto por parte del cliente.
- Es responsable por la evaluación de los riesgos del proyecto y por controlar estos riesgos a través de estrategias que los mitiguen.

- **Interesados:** Este rol representa grupos de interés cuyas necesidades deben ser satisfechas por el proyecto. Esto es un rol que podría ser desempeñado por cualquiera que esté (o potencialmente estará) materialmente afectado por el resultado del proyecto.

Figura 8. Interesados en Open UP [8].

- **Otros:** Cualquiera en un equipo puede cumplir este rol para llevar a cabo tareas generales.

Figura 9. Otros en Open UP [8].

- **Pruebas:** Este rol es responsable de las actividades principales del esfuerzo de las pruebas. Estas actividades incluyen identificar, definir, implementar y dirigir las pruebas necesarias, como también verificar y analizar los resultados.

Figura 10. Pruebas en Open UP [8].

Ver más información de los roles en la metodología Open Up en el *Apéndice D.1.*

2.5.3 *Productos de trabajo (artefactos) de requerimientos en Open UP [8]:*

- **Glosario:** Este artefacto define términos importantes usados por el proyecto. Estos términos son las bases para una colaboración efectiva con los interesados y otros miembros del equipo.

  - **Objetivo:** Proporcionar un vocabulario común aceptado por todos los interesados. Éste puede ayudar a las personas desde diferentes grupos funcionales a alcanzar un entendimiento mutuo del sistema. La meta no es registrar todos los términos posibles, sino únicamente aquellos que son inciertos, ambiguos o requieren elaboración.

  - **Descripción principal:** El Glosario le ayuda a evitar errores conceptuales al proporcionar dos recursos esenciales:

    - Una ubicación central para consultar los términos y abreviaciones que son nuevas para los miembros del equipo de desarrollo.
    - Las definiciones de términos que son usados en forma específica dentro del dominio del proyecto.

Las definiciones para los términos del Glosario provienen de varias fuentes, tales como documentos de requisitos, especificaciones y discusiones con interesados y expertos en el dominio del negocio. En algunos proyectos que no incluyen modelos del negocio, el Glosario es uno de los principales artefacto para capturar información sobre

el dominio de negocio del proyecto. Este artefacto puede ser actualizado en cualquier momento, por cualquier rol, cuando nuevos términos sean identificados.

Los errores conceptuales acerca del significado de datos de elementos conllevan al fracaso de muchos proyectos. Algunos de estos empiezan a ser obvios únicamente en las fases posteriores a las prueba del sistema y puede ser extremadamente costoso corregirlos.

- **Requerimientos del Sistema:** Este artefacto captura requisitos generales del sistema, incluyendo requisitos sobre atributos de calidad y requisitos funcionales globales.

  - **Descripción principal:** Las Características, Requerimientos y Casos de Uso, juntos, definen los requisitos del sistema. Los Casos de Uso describen el comportamiento de los requerimientos.

La meta de este producto de trabajo es asegurarse que todos los tipos de requisitos están cubiertos, lo cual reduce el riesgo de no considerar alguna faceta importante del sistema. Los requisitos son globales e influencian los Mecanismos de Arquitectura que se crearán, así como guiar el desarrollo de los fundamentos del sistema. Estos requisitos son frecuentemente los elementos de mayor costo, porque éstos determinan las opciones arquitectónicas. Además, si no se captura los requisitos globales del sistema en un lugar central, pero los repite a través de los Casos de Uso, el resultado será más mantenimiento y más opciones de cometer errores.

Ver plantilla de requerimientos en el Apéndice B.2

- **Especificación de Casos de Uso:** Este artefacto captura la secuencia de acciones que un sistema realiza y produce un resultado observable de valor a su interacción con el mismo.

  - **Objetivo:** El propósito principal de los Casos de Uso es capturar el comportamiento requerido del sistema desde la perspectiva del usuario final, para alcanzar una o más metas. Diferentes usuarios se benefician en diferente forma, por ejemplo:

    - **Los Clientes** los usan para describir, o al menos para aprobar, la descripción del comportamiento del sistema.
    - **Los Usuarios Potenciales** los usan para entender el comportamiento del sistema.
    - **Los Arquitectos** los usan para identificar la funcionalidad arquitectónicamente significativa.
    - **Los Desarrolladores** usan éstos para entender los comportamientos requeridos del sistema de tal manera que puedan identificar clases desde el flujo de eventos.
    - **Los de pruebas** usan éstos como una base para identificar un subconjunto de los Casos de Prueba requeridos.
    - **Los Gerentes** lo usan para planear y evaluar el trabajo en cada iteración.

- **Los Escritores Técnicos** lo usan para entender la secuencia del comportamiento del sistema, para escribir la documentación.

- **Opciones de representación:** Decidir la extensión de los Casos de Uso que se pueden elaborar:
  - ¿Describir únicamente flujos principales?
  - ¿Describir únicamente los Casos de Uso más importantes?
  - ¿Describir completamente las precondiciones y pos condiciones?
  - ¿Describir escenarios primero, y luego elevar el nivel de abstracción describiendo los flujos de los Casos de Uso?

Algunos proyectos aplican Casos de Uso informalmente para ayudar a descubrir los requisitos, documentar y gestionar estos requisitos en otra forma tal como unas historias de usuario. La forma como se presente los Casos de Uso podría depender del tamaño del proyecto, la experiencia del equipo, el conjunto de herramientas y las relaciones con el cliente.

Ver plantilla de especificación de casos de uso en el Apéndice B.3

- **Modelo de Casos de Uso:** Este artefacto captura en un modelo las funciones del sistema y su entorno, sirve como un contrato entre el cliente y los desarrolladores.

- **Objetivo:** Este artefacto presenta un panorama de la intención de comportamiento del sistema. Es la base para el acuerdo entre las partes interesadas y

el equipo del proyecto con respecto a la funcionalidad del sistema. También ayuda a orientar las diversas tareas en el ciclo de vida de desarrollo de software.

- **Opciones de representación:** Este artefacto sirve para apoyar las necesidades del equipo del proyecto.

Las opciones de representación incluye: informes y diagramas de modelado UML, las representaciones gráficas creadas con herramientas de dibujo, los dibujos en pizarras. La mayoría de la información del modelo de casos de uso es capturada a partir de la especificación de casos de uso.

- **Visión:** Este artefacto contiene la definición de la mirada de los interesados del producto a desarrollar, especificado en términos de las necesidades y características claves de los interesados. Este contiene una vista general de los requisitos principales previstos para sistema.

- **Objetivo:** Este artefacto provee a alto nivel, algunas veces contractual, la base para los requisitos técnicos más detallados que son visibles para los interesados. Captura la esencia del sistema describiendo los requisitos de alto nivel y las restricciones de diseño que dan al lector una apreciación global del sistema, desde una perspectiva de requisitos funcionales. Sirve como entrada para el proceso de aprobación del proyecto, comunica los "Qué y Por qué" fundamentales del proyecto, proporciona un plan frente al cual todas las decisiones futuras deberán ser confrontadas.

- **Descripción principal:** Este artefacto proporciona una visión completa del sistema de software en desarrollo y sirve como soporte del contrato entre los clientes

y la organización de desarrollo. Cada proyecto necesita una fuente para capturar todas las expectativas de los interesados.

Este artefacto es escrito desde la perspectiva de los clientes, enfocado en las características esenciales del sistema y niveles aceptables de calidad. El artefacto debería incluir una descripción de qué características serán incluidas, como también cuales de estas se han considerado pero no se han incluido. Si no se usa este artefacto, hay un alto riesgo de que los interesados y el equipo de desarrolladores tengan diferentes expectativas. Esto podría llevar a la cancelación del proyecto. Se debe enmarcar este artefacto como requisito para las necesidades en el proyecto. Generalmente es buena práctica conservar este artefacto, tan breve como se pueda y tan pronto como sea posible entregar este a los interesados, hacer este fácil de leer y de comprender para los interesados.

Ver plantilla de Visión en el Apéndice B.1

2.6 Herramientas CASE [6]:

Una herramienta CASE[13] se puede definir como un conjunto de programas y ayudas que dan asistencia a los analistas, ingenieros de software y desarrolladores, durante algunos pasos del Ciclo de Vida de desarrollo de un Software.

La realización de un nuevo software requiere que las tareas sean organizadas y

---

13. **CASE:** Computer Aided Software Engineering (Ayuda por Computadora a la Ingeniería de Software)

completadas en forma correcta y eficiente. Las herramientas CASE fueron desarrolladas para automatizar esos procesos y facilitar las tareas de coordinación de los eventos, que necesitan ser mejorados en el ciclo de desarrollo de software.

Estas herramientas pueden proveer muchos beneficios en todas las etapas del proceso de desarrollo de software, algunas de ellas son:

- Verificar el uso de todos los elementos en el sistema diseñado.
- Automatizar el dibujo de diagramas.
- Ayudar en la documentación del sistema.
- Ayudar en la creación de relaciones en la Base de Datos.
- Generar estructuras de código.

La principal ventaja de la utilización de una herramienta CASE, es la mejora de la calidad de los desarrollos realizados y, en segundo término, el aumento de la productividad. Para conseguir estos dos objetivos es conveniente contar con una organización y una metodología de trabajo, además de la propia herramienta.

La mejora de calidad se consigue reduciendo sustancialmente muchos de los problemas de análisis y diseño, inherentes a los proyectos de mediano y gran tamaño (lógica del diseño, coherencia, consolidación, etc.). La mejora de productividad se consigue a través de la automatización de determinadas tareas, como la generación de código y la reutilización de objetos o módulos.

2.6.1 Clasificación de herramientas CASE:

No existe una única clasificación de herramientas CASE y, en ocasiones, es difícil incluirlas en una clase determinada. Podrían clasificarse atendiendo a:

- Las plataformas que soportan.
- Las fases del ciclo de vida del desarrollo de sistemas que cubren.
- La arquitectura de las aplicaciones que producen.
- Su funcionalidad.

Las herramientas CASE, se pueden agrupar de la forma siguiente:

1. Herramientas integradas, I-CASE (Integrated CASE, CASE integrado): abarcan todas las fases del ciclo de vida del desarrollo de sistemas. Son llamadas también CASE workbench.

2. Herramientas de alto nivel, U-CASE (Upper CASE - CASE superior) o front-end: orientadas a la automatización y soporte de las actividades desarrolladas durante las primeras fases del desarrollo: análisis y diseño.

3. Herramientas de bajo nivel, L-CASE (Lower CASE - CASE inferior) o back-end: dirigidas a las últimas fases del desarrollo, construcción e implantación.

4. Juegos de herramientas o Tools-Case, son el tipo más simple de herramientas CASE. Automatizan una fase dentro del ciclo de vida. Dentro de este grupo se encontrarían las herramientas de reingeniería, orientadas a la fase de mantenimiento.

Capítulo III

Metodología

• Visión Preliminar

Actualmente el mercado cuenta con una gran cantidad de Metodologías que contribuyen a realizar de manera eficaz y sencilla todo el proceso de Desarrollo de Software, dentro de esta amplia gama de metodologías están aquellas que han sido bautizadas como *"Metodologías Ágiles"* siendo una de las más representativas y actuales *XP (Extreme Programming)*, la cual mediante un alto grado de colaboración y alta comunicación con los clientes, garantiza una maximización en la entrega de productos o soluciones finales [12].

3.1 eXtreme Programming (XP).

- *3.1.1 ¿Qué es eXtreme Programming?*
- *3.1.2 Principios de XP.*
- *3.1.3 Organización de XP.*
- *3.1.4 Prácticas de XP.*
- *3.1.5 eXtremme Programming y herramienta web (Requerimientos en Open UP)*

3.1 eXtreme Programming (XP):

3.1.1 ¿Qué es eXtreme Programming?

Extreme Programming es una disciplina de desarrollo de software basada en los valores de la simplicidad, la comunicación, la retroalimentación y coraje. Todo el equipo trabaja en presencia de simples prácticas, con suficiente información para ver dónde están y ajustar las prácticas a su situación única [13].

Tiene como objetivo reducir el riesgo en el ciclo de vida del software mediante grupos de desarrollo pequeños. Considera que la mejor manera de tratar la falta de requisitos estables en un sistema, es mediante la agilidad de un grupo pequeño de desarrollo. Aunque XP define varias prácticas a seguir, quizás la más representativa del proceso de XP es la programación en pares (*pair programming*) [18]

3.1.2 Principios de XP

- **Comunicación:** XP hace hincapié en la comunicación cara a cara, en lugar de otros tipos de comunicación, tales como documentos. XP da valor a los documentos, pero valoriza mucho más la comunicación personal. Para facilitar la comunicación, los equipos XP:
    - Utilizan un vocabulario común o metáfora del sistema.
    - Trabajan muy cerca unos de otros, en un espacio abierto.
    - Integran el código continuamente.
    - Trabajan en estrecha colaboración con los profesionales de la empresa,

preferiblemente en el mismo ambiente físico.

- Se programa en parejas.
- Poseen el código colectivamente.
- Planean e informan frecuentemente el estatus del proyecto al cliente.

- **Simplicidad:** XP supone que es mejor hacer una cosa sencilla hoy, que hacer una cosa complicada que nunca será usada. Esta es una filosofía que impregna todo en un proyecto XP. Si algo no se necesita ahora, no se hace hoy.

Por ejemplo:

- No escribir cualquier documento, hasta que haya una necesidad inmediata y significativa.
- No tomar ningún instrumento a menos que exista un beneficio tangible y verificable.

Para mantener la sencillez del software y la estructura del equipo, los equipos XP realizan:

- ¿Cuál es la cosa más fácil en la que se puede trabajar?
- Simplificación y mejora constante del diseño a través de la re-factorización.

Kent Beck ("padre de XP") ofrece las siguientes normas para un diseño simple:

- Se debe gastar en todas las pruebas.
- No debe existir código duplicado.

- Expresar todas las ideas que el autor quiere expresar.
- Reducir al mínimo las clases y métodos.

- **Retroalimentación (feedback):** La retroalimentación es de diferentes escalas en XP. En el nivel más alto, el cliente puede ver el progreso del equipo y del sistema a través de entregas ejecutables cada dos semanas. Esta información continua permite al cliente guiar el proyecto al éxito. Esto permite obtener comentarios concretos sobre el estado del sistema, que está constituido por piezas de funcionalidad que son ejecutables en repetidas ocasiones, por las pruebas de aceptación automática. Estas pruebas permiten prevenir el desarrollo del sistema de nuevo. El nivel de la programación, los desarrolladores deben escribir pruebas unitarias sobre toda la lógica del sistema, para obtener información inmediata y concreta, que les informe si el código que se está haciendo, es justo lo que se quiere escribir.

Los equipos de XP pueden:

- Desarrollar en pequeñas iteraciones.
- Reunir los requisitos y características en las historias que encajan en una iteración.
- Escribir pequeñas unidades para asegurar que el código generado, en las tareas funciona correctamente.
- Escribir pruebas para asegurarse de que las historias funcionen correctamente.

- Supervisar el progreso y mantener la comunicación frecuente con los clientes.

- **Coraje:** Tal vez un mejor nombre para este principio es la confianza al trabajo, los miembros de un equipo de XP deben tener el valor de confiar en los demás, en los clientes, en sus prácticas y en sí mismos. Los miembros del equipo de XP saben que pueden:

  - Detenerse cuando están cansados.
  - Permitir que todas las decisiones sean tomadas por el cliente.
  - Solicitar a los clientes reducir el alcance de una historia.
  - Solicitar ayuda a sus compañeros o clientes.
  - Diseñar y aplicar sólo lo necesario para hoy.
  - Hacer cambios para mejorar la funcionalidad o la estructura del código.
  - Concertar el diseño y actualización del código existente, cuando el diseño se muestra insuficiente.
  - Cambiar el código no escrito.
  - Cambiar el proceso cuando no funciona.

3.1.3 Organización de XP

La metodología XP está organizada en los siguientes apartados:

- **Las Historias de Usuario:**

Esta técnica se utiliza para especificar los requisitos del software. Se trata de tarjetas de papel en las cuales el cliente describe brevemente las características que el sistema debe poseer, sean requisitos funcionales o no funcionales. El tratamiento de las historias de usuario es muy dinámico y flexible. Cada historia de usuario es lo suficientemente comprensible y delimitada para que los programadores puedan implementarlas en unas semanas [12].

- **Roles XP:**

En *eXtreme Programming* existe una clara división entre las atribuciones del cliente y las del desarrollador, ambos están en el mismo equipo pero toman diferentes decisiones. Se pueden resumir las decisiones de cada uno de ellos de la siguiente forma:

  - *El cliente* decide alcance, prioridad, composición de las entregas para hacerlas útiles y listas para la producción, por último la fecha en que éstas entregas deben hacerse.
  - *Los programadores* en cambio deben estimar el tiempo para agregar nuevas características, deben explicar las consecuencias técnicas o posibles opciones para cumplir con algún objetivo, sin embargo el cliente tienen la decisión final. Por último es labor de los programadores establecer el esquema de trabajo del equipo y la planificación de cada iteración.

- **Proceso:**

El ciclo de desarrollo consiste en:

- El cliente define el valor de negocio a implementar.
- El programador estima el esfuerzo necesario para su implementación.
- El cliente selecciona qué construir, de acuerdo con sus prioridades y las restricciones de tiempo.
- El programador construye ese valor de negocio.
- Vuelve al paso 1.

En todas las iteraciones de este ciclo tanto el cliente como el programador aprenden. Esta metodología nos dice, que no se debe presionar al programador a realizar más trabajo que el estimado, ya que se perderá calidad en el software o no se cumplirán los plazos. De la misma forma el cliente tiene la obligación de manejar el ámbito de entrega del producto, para asegurarse que el sistema tenga el mayor valor de negocio posible con cada iteración.

Las fases de XP consiste en:

- Planificación.
- Diseño.

- Desarrollo.
- Pruebas.

3.1.4 Prácticas de XP

- ***Planificación:*** Hay una comunicación frecuente el cliente y los programadores. El equipo técnico realiza una estimación del esfuerzo requerido para la implementación de las historias de usuario y los clientes deciden sobre el ámbito y tiempo de las entregas de cada iteración.
- ***Entregas pequeñas:*** Producir rápidamente versiones del sistema que sean operativas, aunque no cuenten con toda la funcionalidad del sistema. Esta versión ya constituye un resultado de valor para el negocio. Una entrega no debería tardar más 3 meses.
- ***Metáfora:*** El sistema es definido mediante una metáfora o un conjunto de metáforas compartidas por el cliente y el equipo de desarrollo. Una metáfora es una historia compartida que describe cómo debería funcionar el sistema.
- ***Diseño simple:*** Se debe diseñar la solución más simple que pueda funcionar y ser implementada en un momento determinado del proyecto.
- ***Pruebas:*** La producción de código está dirigida por las pruebas. Éstas son establecidas por el cliente antes de escribirse el código y son ejecutadas constantemente ante cada modificación del sistema.
- ***Refactorización (Refactoring):*** Es una actividad constante de reestructuración

del código con el objetivo de remover duplicación de código, mejorar su legibilidad, simplificarlo y hacerlo más flexible para facilitar los posteriores cambios. Se mejora la estructura interna del código sin alterar su comportamiento externo.

- ***Programación en parejas***: Toda la producción de código debe realizarse con trabajo en parejas de programadores. Cualquier programador puede cambiar cualquier parte del código en cualquier momento.
- ***Integración continua:*** Cada pieza de código es integrada en el sistema una vez que esté lista. Así, el sistema puede llegar a ser integrado y construido varias veces en un mismo día.
- ***Cliente in-situ:*** El cliente tiene que estar presente y disponible todo el tiempo para el equipo. Éste es uno de los principales factores de éxito del proyecto XP. El cliente conduce constantemente el trabajo hacia lo que aportará mayor valor de negocio y los programadores pueden resolver de manera inmediata cualquier duda asociada.
- ***Estándares de programación:*** XP enfatiza que la comunicación de los programadores es a través del código, con lo cual es indispensable que se sigan ciertos estándares de programación para mantener el código legible.

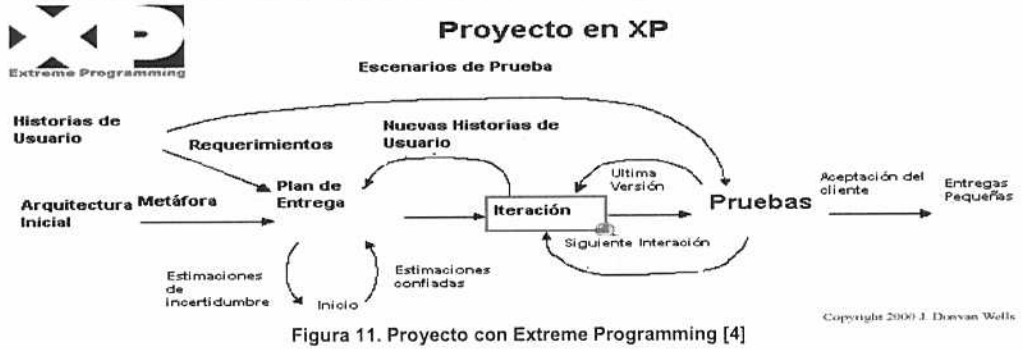


Figura 11. Proyecto con Extreme Programming [4]

3.1.5 eXtreme Programming y herramienta web (Requerimientos en Open UP):

Tomando en cuenta toda la información suministrada sobre metodología XP, es necesario justificar su uso en este trabajo especial de grado, un resumen se ve reflejado en la Tabla 4, en donde se describen las actividades y productos obtenidos:

Fase	Actividad	Producto
Planificación	• Selección de historias de usuario a desarrollar. • Asignación de prioridades de desarrollo a las historias de usuario • Patrón de arquitectura utilizar. • Herramientas de desarrollo	• Historias de Usuario • Prioridad en historias de usuario • MVC. • ODF, GCRS
Diseño	• Aplicación del patrón de arquitectura en los componentes de código (clases) que conforman la herramienta.	• Diagramas de Clases
Desarrollo	• Programación en pares	• Código fuente de la aplicación
Pruebas	• Elaboración de pruebas funcionales, para verificar el correcto funcionamiento de la herramienta.	• Pruebas funcionales de la herramienta.

Tabla 4. eXtreme Programming y la herramienta web.

Las metodologías pesadas basadas en el proceso unificado generan excesiva documentación lo cual resta tiempo para la codificación de la solución final. *XP* permite enfocarse en lo que debe hacer la herramienta e integrar de una manera rápida código funcional a la misma, adicionalmente permite visualizar los resultados más expeditamente. Otro aspecto importante es que *XP* facilita el levantamiento de información a través de la utilización de historias de usuario y permite su evolución siendo posible agregar nuevos requerimientos, refinar el diseño sin afectar los plazos de entrega. Por último la ejecución de un plan de pruebas garantiza la estabilidad de la aplicación final, reduciendo el margen de errores y la aplicación de pruebas de calidad [1].

Capítulo IV

Desarrollo

- Visión Preliminar

En el presente capítulo se da a conocer al lector las iteraciones necesarias para el desarrollo de este Trabajo especial de Grado

4.1 Introducción

El proceso de construcción de la herramienta web para la gestión de requerimientos fue dividido en seis (6) iteraciones. El proceso de ejecución para cada iteración comprende las cuatro (4) fases (Planificación, Diseño, Desarrollo y Pruebas) de la metodología aplicada para el desarrollo de éste Trabajo Especial de Grado.

4.2 Iteraciones:

4.2.1 Iteración 1 - Definición de Plataforma y Herramientas de Desarrollo:

- **Planificación:** El punto inicial del desarrollo de la herramienta web, fue la elaboración de historias de usuario (*Apéndice A*), tal como lo indica la metodología de desarrollo XP. Para cumplir con los objetivos planteados en la creación de ésta herramienta, fue necesario indagar sobre herramientas que cumplan con estándares de código abierto, como un procesador de texto y gestión de cambio. Otro factor importante fue establecer la estructura y construcción de la herramienta web siguiendo un patrón de arquitectura de software.

- **Diseño:** El punto de arranque en esta fase, fue la elaboración de historias de usuario que describen las funcionalidades de la herramienta. Las historias de usuario no son más que *la descripción del sistema desde el punto de vista del usuario*[14] y fueron construidas tomando como formato una plantilla de XP para historias de usuario (Ver

---

14. **Descripción proporcionada por Kent Beck [1].**

Apéndice B.4). La elección del procesador de texto fue otra actividad importante, varias suites ofimáticas fueron comparadas *(OpenOffice, EasyOffice, Ssuite Office)*, Open Office Writer fue el elegido para ser integrado con la herramienta web, a través del uso de librerías y mecanismos de conexión.La elección de OpenOffice Writer como procesador de texto se debe a que forma parte del conjunto de aplicaciones libres del la suite ofimática OpenOffice, está desarrollado por Sun Microsystems, además es multiplataforma, soporta el formato propietario *.doc* (Microsoft Office), posee un formato nativo XML, puede exportar a ficheros PDF sin necesidad de programas intermedios. La librería utilizada para exportar información desde una aplicación desarrollada bajo lenguaje PHP, a OpenOffice Writer es conocida como **ODF**; esta librería genera un archivo XML, permitiendo exportar la información a OpenOffice. La librería también hace uso de la extensión **ZIP** de PHP. La herramienta de gestión de cambio de requerimientos elegida fue **GCRS** (Gestión de Cambio en Requerimientos de Software); la elección de ésta herramienta para la gestión de cambio, se debe a que cumple con estándares abiertos, está desarrollada en lenguaje PHP y se acopla a otras herramientas web a las que presta servicio adaptándose al modelo de base datos de la herramienta propietaria. Toda la arquitectura de la herramienta se llevó a cabo mediante la utilización de un patrón de arquitectura como lo es **MVC** (Modelo Vista Controlador). MVC separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos. En este desarrollo la **Vista** son todas las páginas HTML[15] y

15. **HTML:** (*Lenguaje de Marcas de Hipertexto*), es el lenguaje de marcado predominante para la construcción de páginas web.

el código que provee de datos dinámicos a cada página; el **Modelo** es el Sistema de Gestión de Base de Datos y la Lógica de negocio; y el **Controlador** es el responsable de recibir los eventos de entrada desde la vista. Según lo establecido por la metodología de desarrollo, los requerimientos del sistema, deben ser expresados en historias de usuario. Otro punto a destacar fue la utilización de herramientas CASE, para elaborar el modelo entidad- relación de la base de datos que soporta la herramienta web.

- **Desarrollo:** El objetivo en esta fase fue construir las historias de usuarios, que conforman la base para el desarrollo de la herramienta (*Ver Apéndice A*). Además se elaboró el modelo entidad-relación (*Ver Apéndice F.1)*; también se realizó un diseño general de las interfaces de usuario (*Ver Apéndice F.2)*.

- **Pruebas:** Todas las pruebas realizadas en las iteraciones, se basaron en el proceso de V & V (Verificación y Validación), la verificación a través de la inspección se ocupa del análisis de representaciones estáticas del sistema para describir el problema. La validación nos ayuda a determinar si el sistema cumple con los requerimientos, es decir, intenta determinar la correspondencia entre la realidad y las especificaciones. El plan de pruebas puede ser visto en el *Apéndice E.*

El mapa de historia de usuario, permite visualizar todo el sistema, es decir, representa al producto como un todo. En el mapa se deben ubicar las grandes historias de usuario (actividades) y las historias más pequeñas que las complementan (tareas). A continuación el mapa de Historias de Usuario para el desarrollo de la herramienta web:

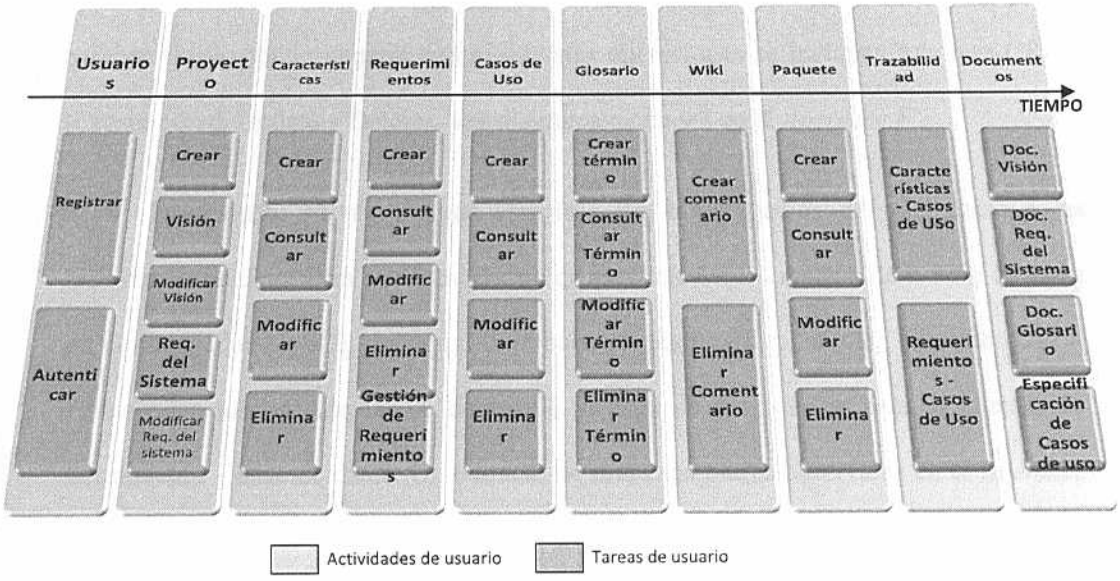


Figura 12. Mapa de historias de Usuario

4.2.2 *Iteración 2 – Diseño del Modelo de Datos y Capa de Persistencia:*

- **Planificación:** El desarrollo de la herramienta se inició por la asignación de prioridad en las historias de usuario, posteriormente se creó la Base de Datos que es la encargada de gestionar los datos y hace posible que esta se convierta en información, con la ayuda de la capa de persistencia. Se utilizó el patrón de diseño **DAO** (Data Access Object) para encapsular la interacción de la herramienta con la base de datos.

- **Diseño:** Para persistir los datos, la herramienta web debe interactuar con la base de datos. El cómo interactúa no debe ser asunto de la capa de lógica de negocio, ya que para eso existe la capa de persistencia, que es la encargada de interactuar con la base de datos. Por medio del patrón DAO (Data Access Object) es posible definir mediante un contrato las operaciones que deben ser ejecutadas en cada una de las entidades de la

base de datos, por lo general dentro de estas operaciones se incluyen los usos CRUD (Create, Read, Update y Delete). Ver diagrama de clases en el *Apéndice F.3*

- **Desarrollo:** El objetivo de esta fase fue la asignación de prioridades a las historias de usuario. Otra actividad dentro de esta fase fue la creación de la base de datos en el manejador MySQL versión 5.0; la base de datos está constituida por catorce (14) tablas relacionadas, además se desarrolló la capa de persistencia para la interacción herramienta – base de datos. Un extracto de esta capa puede observarse en el *Apéndice C.1.*

A continuación se presenta un breve resumen de la asignación de prioridad en las historias de usuario:

Historia de Usuario	Prioridad	Historia de Usuario	Prioridad	Historia de Usuario	Prioridad
Autenticar Usuario	Alta	Registrar Usuario	Alta	Crear Proyecto	Alta
Crear Características	Alta	Crear Requerimientos	Alta	Crear Casos de Uso	Alta
Crear Término	Media	Crear Comentario	Alta	Crear Paquetes	Baja
Consultar Características	Media	Modificar Características	Media	Eliminar Características	Media
Consultar Requerimientos	Media	Modificar Requerimientos	Media	Eliminar Requerimientos	Media
Consultar Casos de Uso	Media	Modificar Casos de Uso	Media	Eliminar Casos de Uso	Media
Consultar Términos	Baja	Modificar Términos	Baja	Eliminar Términos	Baja
Consultar Paquetes	Baja	Modificar Paquetes	Baja	Eliminar Paquetes	Baja
Trazabilidad Características - Requerimientos	Alta	Trazabilidad Requerimientos – Casos de Uso	Alta	Doc. Glosario	Media
Crear Visión	Alta	Doc. Visión	Media	Modificar Visión	Media
Crear doc. Requerimientos del sistema	Alta	Modificar doc. Requerimientos del sistema	Media	Requerimientos del sistema	Media
Herramienta de Gestión de Requerimientos	Alta	Doc. Especificación de casos de uso	Media	Eliminar Comentario	Baja

Tabla 5. Historias de Usuario - Prioridad

- **Pruebas:** Las pruebas ejecutadas, consta de pruebas funcionales aplicadas desde la herramienta para probar el funcionamiento de la capa de persistencia. Ver *Apéndice E*

4.2.3 Iteración 3 – Diseño y Concepción de librerías para la capa de Presentación:

- **Planificación:** El propósito de esta iteración, es producir un conjunto de componentes de código que permitan facilitar el desarrollo de una capa de presentación más usable, que pueda ser asimilada de mejor forma por el usuario. Estos componentes están basados en Javascript los cuales integran dentro de sus funcionalidades comportamiento basado en AJAX[16] permitiendo crear una interfaz de usuario más utilizable e interactiva.

- **Diseño:** Una forma posible de realizar cambios sobre las páginas web, sin necesidad de recargarlas, permitiendo interactividad y velocidad, es a través de la utilización de AJAX, que no es más que una técnica de desarrollo web para crear aplicaciones interactivas. Estas aplicaciones se ejecutan en el cliente[17], es decir, en el navegador de los usuarios mientras se mantiene la comunicación asíncrona con el servidor en segundo plano. En su mayoría la herramienta cuenta con las ventajas ofrecidas por AJAX y Javascript, para el manejo de la información. Gran parte del proceso de desarrollo se centra en el concepto de usabilidad[18], tratando de evitar

---

16. **AJAX:** Asynchronous JavaScript And XML

17. **Cliente:** Es una aplicación informática que se utiliza para acceder a los servicios que ofrece un servidor, normalmente a través de una red.

18. **Usabilidad:** Es un atributo de calidad que evalúa cuán fácil es una interfaz de usuario para su uso [5].

desordenes visuales, evitar el uso excesivo de gráficos, textos en movimiento, decoraciones excesivas que solo llevan a la distracción del usuario, la herramienta cuenta con un árbol que agrupa todos los elementos creados en un proyecto por el equipo de trabajo (usuarios), de esta manera se ofrece a los usuarios una lista detallada de todo lo creado, además permite acceder a todos estos elementos de forma sencilla y rápida. Gracias a las ventajas ofrecidas por AJAX, las páginas creadas permiten realizar modificaciones inmediatas sin necesidad de ser recargadas nuevamente. Con la utilización del componente Javascript se busca validar en el cliente la información introducida por los usuarios, antes de ser enviada al servidor, con la intención de reducir el tráfico en la red y el tiempo de respuesta al usuario *(Ver interfaces Apéndice G)*.

- **Desarrollo:** La primera actividad fue la creación de una función para crear el objeto XML HTTP Request que conforma la base para utilizar AJAX. Seguidamente se crearon funciones encargadas de manejar el cambio de estatus del objeto XML HTTP Request, así como el envío de información manera asíncrona. Ver extracto en el *Apéndice C.2*. Otra actividad fue el desarrollo de funciones en Javascript, para validar los datos enviados por los usuarios, a través de los formularios que se encuentra en las interfaces de la herramienta, porque es más rápido validar en el cliente, que ir hasta el servidor, con esto se puede reducir el tráfico de red, sin embargo estas validaciones también son aplicadas en el servidor, la principal razón de esta acción es prevenir que aún cuando el usuario desactive la ejecución de código Javascript en el navegador, la herramienta pueda procesar las acciones de forma correcta validando los datos enviados

a fin de evitar inconsistencias en los mismos. Un aspecto importante a destacar sobre la librería, es la compatibilidad Cross-Browser[19], esta compatibilidad es necesaria debido a la naturaleza intrínseca de los navegadores, los cuales son construidos por diferentes empresas y no en todos los casos respetan los estándares establecidos en lenguajes tales como: HTML, CSS (Hojas de estilo en cascada) o Javascript, algunas veces las interpretaciones pueden ser distintas por parte de estas empresas de desarrollo creando nuevas funcionalidades y decidiendo cuales soportan. Además a medida que los navegadores evolucionan, también sus especificaciones mejoran, por lo que las distintas versiones del mismo software de exploración web, también pueden mostrar las páginas de manera distinta entre sí. Es por ello que se deben crear páginas web que se vean igual en todos los navegadores, a este tipo de solución se le conoce como Cross-Browser.

- **Pruebas:** En esta fase se ejecutaron pruebas funcionales sobre los componentes de AJAX creadas. Se comprobó la compatibilidad de dichas funciones en los navegadores más conocidos (FIREFOX - iExplorer). Además se validó el correcto desempeño de las funciones de Javascript para validar los datos en los formularios de la herramienta. Ver *Apéndice E*

4.2.4 Iteración 4 – Construcción de la Capa de Negocio:

- **Planificación:** El objetivo de esta iteración, es producir un conjunto de código que permita facilitar el desarrollo de la capa de negocio. Se diseñó y desarrolló los módulos de gestión de los elementos de la herramienta, tales como proyectos, usuarios,

19. **Cross-Browser:** Es el desarrollo de páginas web que se ven exactamente iguales en cualquier navegador.

paquetes, características, requerimientos, casos de uso, términos. Estos módulos permiten funcionalidades básicas tales como creación, modificación, supresión y consulta.

- **Diseño:** Según el patrón MVC el modelo no solamente es la capa de persistencia sino también los servicios de la lógica de negocio, es decir, las clases que definen el contrato de operaciones que soportaran las historias de usuario o funcionalidades principales de la herramienta (*Ver diagrama de arquitectura Apéndice F.4)*. Durante esta iteración se definió el contrato inicial con las funcionalidades que debían ser soportadas en esta iteración.

- **Desarrollo:** El objetivo de esta fase de la iteración, fue el desarrollo inicial de la capa de negocio de la herramienta, es decir, la creación de los servicios que contienen la funcionalidades básicas requeridas, como creación, modificación, consulta, supresión, validación de usuarios, consultas por criterios específicos, entre otras; también se definieron las interfaces necesarias para hacer cumplir la funcionalidad de dichas operaciones. Estas interfaces o contratos de programación como también suelen denominarse tienen como objetivo desacoplar el diseño de la capa del modelo a fin de poder introducir cambios en las clases que implementan estas interfaces sin afectar las capas superiores como el Controlador y la Vista. Un fragmento de estas interfaces y clases desarrolladas pueden observarse en el *Apéndice C.5*.

- **Pruebas:** Para verificar el correcto funcionamiento de las clases e interfaces creadas se ejecutaron pruebas funcionales a fin de validar la vertical completa, ver *Apéndice E.*

4.2.5 Iteración 5 – WIKI, Trazabilidad y Gestión de Cambio:

- **Planificación:** En esta oportunidad el propósito de la iteración es la creación de una aplicación de Ingeniería Social que apoye la comunicación y colaboración entre los miembros del equipo de trabajo. Otro objetivo es generar en la herramienta una matriz de trazabilidad, para que el equipo de trabajo (usuarios), observe la relación entre los niveles de requerimientos de un proyecto. Otro punto importante es lograr la integración de la herramienta web con la herramienta de gestión de cambios de requerimientos **GCRS**, para permitir a los usuarios observar los cambios realizados en los elementos que conforman el proyecto.

- **Diseño:** En esta oportunidad la aplicación de Ingeniería Social, se ha denominado WIKI, que no es más que permitir a los miembros del equipo una comunicación dentro de la herramienta, basado en la generación de comentarios sobre las características, requerimientos y casos de uso. La trazabilidad es una técnica que proporciona una relación entre los distintos niveles de requerimientos en la herramienta. Esta técnica le ayuda a determinar el origen de cualquier requisito. En la herramienta web la trazabilidad, es representada a través de una matriz, donde se indica al usuario si han ocurrido cambios o modificaciones no aprobadas, en alguno de los niveles de requerimientos. Además de mostrar los cambios en la matriz de trazabilidad, éstos pueden observarse en la herramienta de gestión de cambio en requerimientos de software **GCRS**, a través de la integración con la herramienta web. **GCRS** lista todas las modificaciones realizadas en los requerimientos de un proyecto y los usuarios que

efectuaron tales cambios.

- **Desarrollo:** El objetivo de esta fase fue la creación de un conjunto de componentes de código, que permita fluir la información entre la capa de presentación, la capa de negocio y la capa de persistencia; con el fin de permitir la generación de comentarios sobre características, requerimientos, casos de uso y generar una matriz de trazabilidad. Para lograr la creación de la matriz de trazabilidad, también fue necesario desarrollar un conjunto importante de componentes de código, que lograra a través de la capa de persistencia extraer la relación entre los distintos de niveles de requerimientos, una vez extraído tales datos, otros componentes fueron construidos en la capa de negocio a fin de transformar esos datos en información lógica y entendible para el usuario. Una vez que se obtuvo la información se llevó a la capa de presentación para que el usuario pueda observar las relaciones entre requerimientos y observar cuales presentan alguna modificación no aprobada. El componente de AJAX desarrollado para la herramienta, también fue utilizado en esta iteración, para hacer la generación de comentarios más dinámica y rápida, así como también para la aprobación de cambios sobre la matriz de trazabilidad. Para lograr la integración entre la herramienta web y **GCRS** fue necesario el desarrollo de un importante conjunto de componentes de código, a fin de permitir a **GCRS** interactuar con la capa de persistencia desarrollada para la herramienta web y poder generar el historial de cambios realizados en los niveles de requerimientos creados.
- **Pruebas:** En esta fase de la iteración fueron ejecutas pruebas funcionales, para

comprobar el funcionamiento de los componentes creados y detectar algún error, que afecte el correcto funcionamiento de la herramienta. Ver *Apéndice E*

4.2.6 Iteración 6 – Generar documentos de Open UP en la herramienta:

- **Planificación:** En esta oportunidad el objetivo es la generación de documentos o artefactos de la metodología Open UP, desde la herramienta web. Tales documentos son la Visión del proyecto, Requerimientos del sistema, Glosario. Estos documentos creados a través de la herramienta deben ser "transferidos" a un procesador de texto de código abierto.

- **Diseño:** En esta oportunidad la iteración está basada en la construcción de un mecanismo de integración entre la herramienta web desarrollada y la librería ***ODF***, que en conjunto con la extensión ***ZIP*** de ***PHP***, permite a través de la creación de un archivo ***XML***, generar un documento en OpenOffice Writer desde una aplicación desarrollada en lenguaje PHP. Ver extracto de la librería ***ODF*** y del mecanismo de integración en el *Apéndice C.6 y C.7.*

- **Desarrollo:** Esta fase se basa en la construcción de componentes de código que permitan a través de la interacción con la capa de persistencia extraer los datos, y por el mecanismo de integración desarrollado, enviarlos a la librería ***ODF*** y ésta a su vez pueda generar un archivo ***XML*** que es interpretado por el procesador de texto.

- **Pruebas:** Como en otras iteraciones, esta fase se basa en la ejecución de pruebas funcionales para verificar la completa vertical de la herramienta. Ver *Apéndice E*

Capítulo V

Resultados

• Visión Preliminar

5.1 Objetivos vs Resultados.

En el presente capítulo se da a conocer al lector los resultados obtenidos en cada objetivo durante el desarrollo de este Trabajo Especial de Grado

5.1.1 *Diseñar e implementar una herramienta web, que permita la gestión de requerimientos.*

5.1.2 *Diseñar y construir una base de datos relacional, para centralizar los distintos tipos de requerimientos y sus atributos.*

5.1.3 *Diseñar e implementar las interfaces web necesarias para la gestión de requerimientos.*

5.1.4 *Desarrollar un módulo de administración para la herramienta web que permita asociar a los proyectos con los usuarios.*

5.1.5 *Diseñar y desarrollar un mecanismo para la integración de una herramienta software libre que genere documentos con la herramienta de gestión de requerimientos, reflejando los requerimientos almacenados en la base de datos.*

5.1.6 *Diseñar y desarrollar un mecanismo para la integración de una herramienta de código abierto para la gestión de cambios en los requerimientos, con la herramienta de gestión de requerimientos.*

5.1.7 *Diseñar y desarrollar un mecanismo para la integración de aplicaciones de Ingeniería Social que apoyen la comunicación, colaboración y coordinación del equipo de trabajo, con la herramienta de gestión de requerimientos.*

5.1 Objetivos vs Resultados:

5.1.1 Diseñar e implementar una herramienta web, que permita la gestión de requerimientos:

Este objetivo no es más que la herramienta web en su totalidad. La herramienta está diseñada para permitir a los usuarios la gestión de requerimientos de un proyecto basado en metodología Open UP; además posee un mecanismo de integración con el procesador de texto OpenOffice Writer, a fin de poder exportar la documentación generada (Visión del Proyecto, Requerimientos del sistema, Glosario) desde la herramienta que es necesaria para cualquier proyecto de Open UP. Igualmente está integrada con la herramienta de gestión de cambio **GCRS**. Permitiendo a los usuarios observar los cambios generados. La herramienta está soportada por una base de datos relacional, donde se almacenan los datos de los distintos niveles de requerimientos y sus respectivos atributos. Además otro punto importante que debe mantenerse en la gestión de requerimientos, es la trazabilidad, para proporcionar la relación entre los distintos niveles de requerimientos de un proyecto; la herramienta refleja la trazabilidad entre requerimientos, a través de una matriz, que permite a los usuarios comprobar que se cumplen con todos los requisitos del proyecto, verificar que se han creado los requerimientos necesarios para cumplir las necesidades del cliente y ayudar con la gestión de cambio dentro del proyecto, mostrando cuales requisitos fueron modificados y no poseen aprobación. La herramienta fue desarrollada en ***PHP*** versión ***5.2.6*** y la base de datos que la soporta esta creada bajo el manejador ***MySQL*** versión ***5.0.51b***; también

se desarrollaron componentes de AJAX y Javascript para lograr una herramienta interactiva y con mayor velocidad.

5.1.2 Diseñar y construir una base de datos relacional, para centralizar los distintos tipos de requerimientos y sus atributos:

Para cumplir con este objetivo fue creado un modelo entidad – relación con ayuda de herramientas CASE, con el fin de reflejar todas las entidades, relaciones y restricciones necesarias para soportar la gestión de requerimientos según las normas que establece la metodología Open UP. El modelo de datos diseñado para soportar la gestión de requerimientos, fue normalizado hasta la tercera forma normal, es decir, que todos los atributos son atómicos, existen claves primaria, no hay dependencias parciales, ningún atributo forma parte de ninguna clave porque dependen directamente y no transitivamente de la clave primaria. Además fueron creados índices, que nos ayuda a obtener datos de las "tablas" creadas en forma más rápida, porque sin un índice el manejador de base de datos debe leer a través de todas los registros de las tablas para localizar la información deseada; con la utilización de índices, el manejador de base de datos puede entonces dirigirse primero a los registros indexados agilizando la búsqueda. La base de datos creada que soporta la herramienta está constituida por catorce (14) tablas relacionadas, en MySQL versión 5.0.51b (Modelo *Ap. F* y script en el *Ap. C.8).*

5.1.3 Diseñar e implementar las interfaces web necesarias para la gestión de requerimientos:

Este objetivo fue alcanzado con la creación de todas las páginas web que conforman la herramienta de gestión de requerimientos, para esto se hizo uso del lenguaje HTML que permite definir la estructura y semántica del contenido y de CSS (Hojas de estilo en cascada) que define la presentación (colores, fuentes, fondos, tamaño de fuentes, etc.) . Estas interfaces web (páginas web) permiten al usuario trabajar con la herramienta para gestionar todos los niveles de requerimientos(características, requerimiento y casos de uso) de forma interactiva y rápida, mediante el componente desarrollado de AJAX, que permiten actualizar fragmento de una página sin recargarla completamente; también se utilizó el componente de Javascript que permite hacer validaciones directamente en el cliente, sin necesidad de llegar hasta el servidor, permitiendo de esta manera reducir el tráfico en la red y tiempo de ejecución. Las páginas fueron desarrolladas manteniendo un estándar de diseño a través de los estilos CSS. Además fue verificada la compatibilidad Cross-Browser para todas las páginas creadas.

5.1.4 Desarrollar un módulo de administración para la herramienta web que permita asociar a los proyectos con los usuarios:

Este objetivo fue alcanzado mediante el desarrollo de un componente de código, que permite asociar a los usuarios registrados en la herramienta web con los proyectos creados, según los roles de trabajo de la metodología Open UP. Este componente permite validar el acceso y los privilegios de los usuarios en la herramienta, según el rol ocupado en el proyecto. La asociación usuario-proyecto puede efectuarse al crear un

nuevo proyecto o al gestionar uno existente. Además se permite incluir o excluir usuarios al equipo de trabajo, como también modificar el rol que ocupan dentro del proyecto. A continuación un resumen de los roles y artefactos que puede modificar:

Rol	Artefactos	Actividades Adicionales
Analista	• Glosario. • Requerimientos. • Casos de Uso. • Modelo de Casos de Uso. • Visión.	• Analizar los requerimientos de arquitectura. • Evaluar los resultados. • Crear casos de prueba. • Diseñar soluciones. • Manejo y planificación de iteraciones. • Plan de proyecto.
Arquitecto	• Block de Notas del Arquitecto. • Diseño.	• Definir la Visión. • Diseñar soluciones y evaluar resultados. • Requerimientos. • Manejo y planificación de iteraciones. • Desarrollar la arquitectura.
Desarrollador	• Construir. • Diseño. • Pruebas desarrollador. • Implementación. • Casos de Uso.	• Analizar los requerimientos de arquitectura. • Evaluar los resultados. • Crear casos de prueba. • Detallar Requerimientos. • Desarrollar la arquitectura. • Buscar y analizar requerimientos. • Manejo y planificación de iteraciones. • Plan de proyecto.
Gerente	• Planificación de Iteración y proyecto • Lista de riesgos. • Evaluaciones de estado. • Lista de productos de trabajo.	• Analizar los requerimientos de arquitectura. • Definir Visión. • Desarrollar la arquitectura.
Interesados		• Evaluar resultados. • Crear casos de prueba. • Definir Visión. • Diseñar la solución. • Detallar los requerimientos. • Desarrollar la arquitectura. • Buscar y analizar requerimientos. • Implementar la solución. • Manejo y planificación de iteración. • Plan de proyecto.
Otros	• Lista de productos de trabajo.	• Evaluar resultados. • Gestión de iteración.
Pruebas	• Casos de prueba. • Log de pruebas. • Script de pruebas. • Implementar pruebas.	• Evaluar resultados. • Requerimientos. • Implementar soluciones. • Planificar iteración.

Tabla 6. Roles y privilegios de Open UP

5.1.5 Diseñar y desarrollar un mecanismo para la integración de una herramienta software libre que genere documentos con la herramienta de gestión de requerimientos, reflejando los requerimientos almacenados en la base de datos:

Este objetivo fue alcanzado con la creación de un componente de código, que permite integrar la herramienta web con el procesador de texto OpenOffice Writer. Este componente hace uso de la librería ODF y de unas plantillas creadas en Open Office Writer para generar los documentos. Este componente extrae la información necesaria (datos, nombre de la plantilla creada en OpenOffice Writer) del modelo de datos para crear el documento por medio de la librería ODF la cual crea un documento XML con la información necesaria para que pueda ser interpretado por OpenOffice Writer para su manipulación

5.1.6 Diseñar y desarrollar un mecanismo para la integración de una herramienta de código abierto para la gestión de cambios en los requerimientos, con la herramienta de gestión de requerimientos:

Este objetivo fue alcanzado con la creación de un componente de código, que permite integrar la herramienta web con la herramienta de gestión de cambios en requerimientos de software **GCRS**. Este componente extrae la información necesaria a través de la capa de persistencia para que **GCRS** genere una lista con todos los cambios realizados en los distintos niveles de requerimientos, los usuarios que efectuaron éstos cambios y la fecha de modificación, también brinda a los usuarios la oportunidad de

aprobar dichos cambios. La lista generada ésta organizada por los proyectos creados desde la herramienta web.

5.1.7 Diseñar y desarrollar un mecanismo para la integración de aplicaciones de Ingeniería Social que apoyen la comunicación, colaboración y coordinación del equipo de trabajo, con la herramienta de gestión de requerimientos:

Este objetivo pudo ser alcanzado mediante el desarrollo de componentes de código y páginas web. Estas páginas desarrolladas con el uso de HTML, CCS (hojas de estilo en cascada), AJAX y Javascript, permite a los usuarios crear comentarios en cualquier nivel de requerimiento. Con estos comentarios se ofrece al equipo del proyecto un mecanismo para realizar todo tipo de observaciones, además de establecer un medio de comunicación, colaboración y coordinación durante el ciclo de vida del proyecto. La metáfora de estos comentarios se basa en dos ideas principales: rapidez y colaboración. Rapidez puesto que es sencillo generar el comentario y Colaboración porque los usuarios intercambian mensajes que les permitan aclarar dudas sobre cualquier requerimiento según su nivel. El equipo del proyecto puede acceder a los comentarios a través de su nivel de requerimiento, donde son mostrados con su descripción y usuario que lo creó.

Capítulo VI

Conclusiones y Recomendaciones

- Visión Preliminar

6.1 Objetivos vs Resultados.

6.1.1 Diseñar e implementar una herramienta web, que permita la gestión de requerimientos.

6.1.2 Diseñar y construir una base de datos relacional, para centralizar los distintos tipos de requerimientos y sus atributos.

6.1.3 Diseñar e implementar las interfaces web necesarias para la gestión de requerimientos.

6.1.4 Desarrollar un módulo de administración para la herramienta web que permita asociar a los proyectos con los usuarios.

6.1.5 Diseñar y desarrollar un mecanismo para la integración de una herramienta software libre que genere documentos con la herramienta de gestión de requerimientos, reflejando los requerimientos almacenados en la base de datos.

En el presente capítulo se da a conocer al lector las conclusiones y Recomendaciones para futuros Trabajos Especiales de Grado relacionados con la Gestión de Requerimientos.

6.1.6 Diseñar y desarrollar un mecanismo para la integración de una herramienta de código abierto para la gestión de cambios en los requerimientos, con la herramienta de gestión de requerimientos.

6.1.7 Diseñar y desarrollar un mecanismo para la integración de aplicaciones de Ingeniería Social que apoyen la comunicación, colaboración y coordinación del equipo de trabajo, con la herramienta de gestión de requerimientos.

6.2 Recomendaciones.

6.1 Objetivos vs Conclusiones:

6.1.1 Diseñar e implementar una herramienta web, que permita la gestión de requerimientos:

Independientemente de la metodología que se elija para el desarrollo de un proyecto de software, la etapa más importante es la definición de requerimientos, ya que ésta determina en gran parte las necesidades que el sistema debe cumplir. Sin embargo durante el desarrollo del proyecto, estos requerimientos pueden ir cambiando para adaptarse a las necesidades de los usuarios, por esta razón fue creada una herramienta web que permite gestionar todos los requerimientos de un proyecto durante el ciclo de vida de su desarrollo bajo metodología Open UP. Además de la gestión de requerimientos, ésta herramienta permite al equipo del proyecto observar la trazabilidad entre los niveles de requerimientos, los cambios efectuados a través de una herramienta de gestión de cambios en requerimientos, generar los documentos exigidos por la metodología y permite la colaboración a través de comentarios entre los miembros del equipo, ayudando a la toma de decisiones dentro del proyecto. Debido a que es una herramienta web el equipo de trabajo posee la oportunidad de interactuar a través de la red, teniendo una visión general de todos los niveles de requerimientos dentro de un proyecto, ya que trabajan sobre una base de datos centralizada, evitando ambigüedades y redundancias de información.

6.1.2 Diseñar y construir una base de datos relacional, para centralizar los distintos

tipos de requerimientos y sus atributos:

La creación de una base de datos relacional le otorga flexibilidad a los datos almacenados, de forma tal, que éstos puedan utilizarse con distintos fines, ya que al estar todos los datos integrados se puede extraer información adicional sobre los mismos. Además se reduce el riesgo de que se presenten inconsistencias, porque si un dato está almacenado una vez cualquier actualización se debe realizar sólo una vez y está disponible para todos los usuarios inmediatamente. Una ventaja en la creación de la base de datos relacional, es que ésta mantiene la validez y consistencia en los datos almacenados, mediante restricciones o reglas que no se pueden violar y pueden ser aplicadas tanto a los datos como a las relaciones. Asimismo los usuarios pueden acceder simultáneamente a la misma información, sin que el acceso interfiera entre ellos.

6.1.3 Diseñar e implementar las interfaces web necesarias para la gestión de requerimientos:

Las interfaces web contienen elementos que permiten una comunicación activa entre los usuarios y la información almacenada en la base de datos. La creación de estas interfaces web permite a los usuarios acceder de forma interactiva a la información, ya que no es necesario recargar las páginas para responder a cada una de las acciones ejecutadas, esto es posible gracias a la utilización de una técnica de desarrollo web como AJAX que utiliza una combinación de varias tecnologías.

6.1.4 Desarrollar un módulo de administración para la herramienta web que permita

asociar a los proyectos con los usuarios:

Un aspecto importante de cualquier sistema de gestión es la seguridad de la información que se maneja, esa seguridad se puede garantizar a partir de la concesión de permisos para controlar que usuarios deben tener acceso a parte o a toda la información y quiénes no. La creación de éste modulo, permite la asociación usuarios – proyectos basado en los roles de la metodología Open UP, para determinar que acciones pueden los usuarios llevar a cabo dentro de la herramienta.

6.1.5 Diseñar y desarrollar un mecanismo para la integración de una herramienta software libre que genere documentos con la herramienta de gestión de requerimientos, reflejando los requerimientos almacenados en la base de datos:

Los documentos que se deben generar en la metodología Open UP forman parte del conjunto de artefactos o entregables a la hora de construir un sistema. Los artefactos correspondientes a la gestión de requerimientos no son más que productos tangibles que ayudan a la descripción de funciones, arquitectura y diseño del sistema. Debido a que son productos tangibles se debe permitir a los usuarios generar éstos artefactos desde la herramienta web, esto es posible con la integración de la herramienta y el procesador de texto OpenOffice Writer, integración que se logró con el uso de librerías, extensiones del lenguaje de desarrollo (***ODF, ZIP – PHP***) y con archivos ***XML*** que permiten la flexibilidad necesaria para la generación de documentos entre la herramienta y el procesador de texto.

6.1.6 Diseñar y desarrollar un mecanismo para la integración de una herramienta de código abierto para la gestión de cambios en los requerimientos, con la herramienta de gestión de requerimientos:

El cambio es inevitable cuando se construye un software, los requerimientos forman parte de la base en el desarrollo de un sistema y también pueden cambiar, la gestión de cambio en los requerimientos es muy importante porque permite identificar, controlar, auditar e informar al equipo de trabajo las modificaciones que invariablemente pueden ocurrir durante el proceso de desarrollo del software. Un proceso de gestión de cambio de requerimientos proporciona un rastreo completo y preciso de todos los cambios que son pertinentes al proyecto.

6.1.7 Diseñar y desarrollar un mecanismo para la integración de aplicaciones de Ingeniería Social que apoyen la comunicación, colaboración y coordinación del equipo de trabajo, con la herramienta de gestión de requerimientos:

La comunicación dentro de un proyecto es de vital importancia, a fin de mantener acuerdos y toma de decisiones durante su ejecución. La creación de un medio para la comunicación dentro de la herramienta web, ofrece la libertad a los usuarios a través de una interfaz simple, de realizar comentarios y expresar opiniones o dudas sobre cada uno de los niveles de requerimientos dentro del proyecto.

6.2 Recomendaciones:

En esta sección se presenta algunas mejoras que pueden realizarse a la herramienta web desarrollada como Trabajo Especial de Grado:

- Implementar un módulo que permita a los usuarios realizar modelado ***UML*** *(Unified Modeling Language)* dentro de la herramienta.
- Integrar la herramienta web con una herramienta de código abierto para pruebas de software, permitiendo definir casos de prueba para cada uno de los niveles de requerimientos creados a partir de la herramienta web.
- Integrar la herramienta web con una herramienta de código abierto para el control de versiones que permita regresar a estados anteriores (deshacer cambios) en los distintos niveles de requerimientos.
- Diseñar y desarrollar un mecanismo para que la herramienta sea configurable y pueda trabajar con manejadores distintos a MySQL.
- Utilizar gráficos interactivos en los casos que sean adecuados a fin de mejorar la usabilidad de la herramienta.
- Permitir que la herramienta gestione requerimientos para metodologías distintas a Open UP.
- Incluir un módulo en la herramienta web que permita configurar el idioma en las interfaces, datos almacenados y artefactos generados.

- Incluir un módulo en la herramienta web que permita asignar cuentas de correo electrónico a los usuarios registrados, con el fin de recibir notificaciones cada vez que ocurra un cambio en cualquier nivel de requerimientos.

Referencias Bibliográficas

- *Visión Preliminar*

A continuación se presenta al lector todos los textos, documentos, páginas web y lugares de interés que permitieron obtener parte de los conocimientos e información necesaria para el desarrollo de éste Trabajo Especial de Grado.

Referencias Bibliográficas:

[1] Beck, K. (2000). Extreme Programming Explained: Embrace Change. Estados Unidos de América. Addison-Wesley.

[2] CMU (Carnegie Mellon University). SEI (Software Engineering Institute) (2006). CMMI for Development, Versión 1.2 CMMI-DEV CMU/SEI. [En línea] Disponible en: http://www.sei.cmu.edu/publications/documents [Mayo 2008].

[3] Deek, F; McHugh, J; Eljarabi O. (2005) Strategic Software Engineering. Estados Unidos de América: Auerbach Publications. Taylor & Francis Group.

[4] Extreme Programming: A gentle introduction, [En línea].Disponible en: http://www.extremeprogramming.org

[5] Galitz W. (2007). The Essential Guide to User Interface Design An Introduction to GUI Design Principles and Techniques. Estados Unidos de América: Wiley Publishing, Inc

[6] Herramientas CASE. Instituto Nacional de estadística e informática Perú (1999). [En línea] Disponible en: http://www.innovavirtual.org .

[7] International Institute of Business Analysis. [En línea] Disponible en: http://www.theiiba.org

[8] Introduction to Open UP. [En línea] Disponible en: http://epf.eclipse.org/openup/

[9] Introduction to XP. [En línea]. Disponible en: http://epf.eclipse.org/xp/

[10] ISO 9000:2000. [En línea] Disponible en: http://www.iso.org

[11] Jacobson, I., Booch, G., Rumbaugh, J (2000). El Proceso unificado de desarrollo de software. Madrid: Addison Wesley.

[12] Jorge Ferrer Zarzuela. Metodologías Ágiles, [En línea]. Disponible en: http://libresoft.es

[13] Jeffries, R. (2001). ¿Qué es eXtreme Programming? [En línea] Disponible en: http://www.xprogramming.com

[14] Leffingwell, Dean.Managing Software Requirement. Addison Wesley

[15] Palacios, J. (2006). Sinopsis del modelo CMMI, [En línea]. Disponible en: http://www.navegapolis.net

[16] Richter, K (2008), CMMI for Acquisition (CMMI – ACQ) Version 1.2 CMU/SEI. [En línea] Disponible en: http://www.sei.cmu.edu/publications/documents [Mayo 2008].

[17] SEI. What is CMMI? , [En línea]. Disponible en: http://www.sei.cmu.edu/cmmi/

[18] Weitzenfeld, A (2005) Ingeniería de Software Orientada a Objetos con UML, JAVA e INTERNET. México: Thomson

[19] Zielczynski, P., Ph. D (2007). Requirements Management. Estados Unidos: Pearson Education.

Apéndices

• Visión Preliminar

Apéndice A. Historias de Usuario.

Apéndice B. Plantillas.

- *B.1. Documento de Visión.*
- *B.2. Documentos de Requerimientos del sistema.*
- *B.3. Casos de Uso.*
- *B.4 Historias de Usuario.*

Apéndice C. Componentes de Código.

- *C.1. Componentes para la capa de persistencia.*
- *C.2. XMLHttpRequest.*
- *C.3. AJAX.*
- *C.4 JAVASCRIPT.*
- *C.5. Componentes para la capa de Negocio.*
- *C.6. ODF.*
- *C.7. Mecanismo de integración PHP - OpenOffice Writer.*
- *C.8. Script de la Base de Datos.*

Apéndice D. Roles en Open UP y CMMI

- *D.1. Roles en Open UP.*
- *D.2. Integración del modelo de madurez de capacidades (CMMI).*

Apéndice E. Pruebas

Apéndice F. Modelo Entidad - Relación.

Apéndice G. Manual de Usuario.

A continuación se presenta al lector información vinculada al desarrollo de este Trabajo Especial de Grado. Como plantillas utilizadas, algunos aspectos de diseño, pruebas, documentación.

Apéndice A – Historias de Usuario:

Historia de Usuario	
Número: 1	**Nombre:** Registrar Usuario
Usuario: Equipo proyecto	
Modificación de Historia Número:	
Prioridad en Negocio: Alta **(Alta / Media / Baja)**	**Puntos Estimados:**
Riesgo en Desarrollo: Alta **(Alto / Medio / Bajo)**	**Puntos Reales:**
Descripción: Se introducen los datos del usuario (nombre, usuario, contraseña, pregunta y respuesta secreta). Se debe validar la información y enviar a la base de datos, para crear el registro del usuario.	
Observaciones:	

Historia de Usuario	
Número: 2	**Nombre:** Crear Proyecto
Usuario: Equipo proyecto	
Modificación de Historia Número:	
Prioridad en Negocio: Alta **(Alta / Media / Baja)**	**Puntos Estimados:**
Riesgo en Desarrollo: Alta **(Alto / Medio / Bajo)**	**Puntos Reales:**
Descripción: Se debe ingresar a la herramienta (previamente realizar autenticación de usuario) solicitar nombre y descripción del proyecto, además se debe elegir los usuarios que formaran el equipo del proyecto indicando su rol dentro del mismo. La herramienta debe registrar la información en la base de datos, y debe crear los paquetes básicos del proyecto (Visión y Características, Requerimientos del Sistema, Casos de Uso, Glosario y Wiki).	
Observaciones: Permitir asignar usuarios al proyecto en la misma página web. Posteriormente cargar toda la información básica del proyecto.	

Historia de Usuario	
Número: 3	**Nombre:** Crear Características
Usuario: Equipo proyecto	
Modificación de Historia Número:	
Prioridad en Negocio: Alta **(Alta / Media / Baja)**	**Puntos Estimados:**
Riesgo en Desarrollo: Alta **(Alto / Medio / Bajo)**	**Puntos Reales:**
Descripción: Se debe ingresar a la herramienta (previamente realizar autenticación de usuario) en un proyecto, Al solicitar Crear Característica, ingresar toda la información necesaria (nombre, descripción, tipo, paquete de ubicación, origen, estatus, riesgo, estabilidad, dificultad, prioridad, iteraciones planificadas, iteración actual, persona contacto, alguna solicitud de mejora, defecto, obsoleto) y registrar la característica en el proyecto y paquete seleccionado.	
Observaciones: El paquete por defecto donde se almacenará las características de un proyecto será en Visión y Característica; sin embargo puede ser almacenado en otro paquete creado por el usuario previamente. El **tipo** de característica son (Funcional, Usabilidad, Fiabilidad, Rendimiento, Restricción de diseño, Requerimiento de aplicación, Requerimiento físico, Requerimiento de interfaz, Soporte). El **origen** (Clientes, Socios, Competencia, Usuario Final, Help Desk). El **estatus** (Aprobado, Incorporado, Validado, Proyecto). **Riesgo** (Calendario (Alto), Calendario (Medio), Calendario (Bajo), Tecnología (Alto), Tecnología (Medio), Tecnología (Bajo)). **Estabilidad, dificultad y prioridad** (Alta, Media, Baja).	

Historia de Usuario	
Número: 4	**Nombre:** Crear Requerimientos
Usuario: Equipo proyecto	
Modificación de Historia Número:	
Prioridad en Negocio: Alta **(Alta / Media / Baja)**	**Puntos Estimados:**
Riesgo en Desarrollo: Alta **(Alto / Medio / Bajo)**	**Puntos Reales:**
Descripción: Se debe ingresar a la herramienta (previamente realizar autenticación de usuario) en un proyecto, Al solicitar Crear Requerimientos, ingresar toda la información necesaria (nombre, descripción, tipo, paquete de ubicación, característica origen, estatus, riesgo, estabilidad, dificultad, prioridad, iteraciones planificadas, iteración actual, persona contacto, alguna solicitud de mejora, defecto, obsoleto) y registrar el requerimiento en el proyecto y paquete seleccionado.	
Observaciones: El paquete por defecto donde se almacenará los requerimientos de un proyecto será en Requerimientos del sistema; sin embargo puede ser almacenado en otro paquete creado por el usuario previamente. El **tipo** de requerimientos son (Funcional y No Funcional). El **origen** (Se debe listar todas las características creadas en el proyecto). **Riesgo** (Calendario (Alto), Calendario (Medio), Calendario (Bajo), Tecnología (Alto), Tecnología (Medio), Tecnología (Bajo)). **Estabilidad, dificultad y prioridad** (Alta, Media, Baja).	

Historia de Usuario	
Número: 5	**Nombre:** Crear Casos de Uso
Usuario: Equipo proyecto	
Modificación de Historia Número:	
Prioridad en Negocio: Alta **(Alta / Media / Baja)**	**Puntos Estimados:**
Riesgo en Desarrollo: Alta **(Alto / Medio / Bajo)**	**Puntos Reales:**
Descripción: Se debe ingresar a la herramienta (previamente realizar autenticación de usuario) en un proyecto, Al solicitar Crear Casos de Uso, ingresar toda la información necesaria (nombre, descripción, actores, pre condiciones, post condiciones, Flujo básico de eventos, sub flujos, requerimientos especiales, escenarios claves), paquete de ubicación, requerimiento origen, estatus, riesgo, estabilidad, dificultad, prioridad, iteraciones planificadas, iteración actual, persona contacto) y registrar el caso de uso en el proyecto y paquete seleccionado.	
Observaciones: El paquete por defecto donde se almacenará los casos de uso de un proyecto será en Casos de Uso; sin embargo puede ser almacenado en otro paquete creado por el usuario previamente. Los flujos, sub flujos, escenarios claves deben poder crearse dinámicamente. **Estabilidad, dificultad y prioridad** (Alta, Media, Baja).	

Historia de Usuario	
Número: 6	**Nombre:** Crear Término
Usuario: Equipo proyecto	
Modificación de Historia Número:	
Prioridad en Negocio: Media **(Alta / Media / Baja)**	**Puntos Estimados:**
Riesgo en Desarrollo: Medio **(Alto / Medio / Bajo)**	**Puntos Reales:**
Descripción: S e debe ingresar a la herramienta (previamente realizar autenticación de usuario) en un proyecto, Al solicitar Crear Término, ingresar toda la información necesaria (nombre, descripción, paquete de ubicación) registrar el termino en el proyecto y paquete seleccionado.	
Observaciones: El paquete por defecto donde se almacenará los términos del proyecto será en Glosario; sin embargo puede ser almacenado en otro paquete creado por el usuario previamente	

Historia de Usuario	
Número: 7	**Nombre:** Crear Comentario
Usuario: Equipo proyecto	
Modificación de Historia Número:	
Prioridad en Negocio: Alta **(Alta / Media / Baja)**	**Puntos Estimados:**
Riesgo en Desarrollo: Alto **(Alto / Medio / Bajo)**	**Puntos Reales:**
Descripción: Se debe ingresar a la herramienta (previamente realizar autenticación de usuario) en un proyecto, Al solicitar Crear Comentario, seleccionar la característica, requerimiento o caso de uso a comentar, ingresar la información pertinente (titulo y comentario) y automáticamente deben ser almacenados en el proyecto, en el paquete Wiki según su clasificación (característica, requerimiento, caso de uso).	
Observaciones:	

Historia de Usuario	
Número: 8	**Nombre:** Crear Paquete
Usuario: Equipo proyecto	
Modificación de Historia Número:	
Prioridad en Negocio: Baja **(Alta / Media / Baja)**	**Puntos Estimados:**
Riesgo en Desarrollo: Bajo **(Alto / Medio / Bajo)**	**Puntos Reales:**
Descripción: Se debe ingresar a la herramienta (previamente realizar autenticación de usuario) en un proyecto, Al solicitar Crear Paquete, ingresar toda la información necesaria (nombre, descripción), registrar el paquete en el proyecto.	
Observaciones:	

Historia de Usuario	
Número: 9	**Nombre:** Autenticar usuario
Usuario: Equipo proyecto	
Modificación de Historia Número:	
Prioridad en Negocio: Alta **(Alta / Media / Baja)**	**Puntos Estimados:**
Riesgo en Desarrollo: Alta **(Alto / Medio / Bajo)**	**Puntos Reales:**
Descripción: Se introducen los datos del usuario (proyecto, nombre de usuario y clave). Se debe realizar una búsqueda en la base de datos, para validar los datos introducidos, y de ser correctos, buscar toda la información del proyecto y permitir el acceso. Si son incorrectos enviar un mensaje de notificación de error.	
Observaciones:	

Historia de Usuario	
Número: 10	**Nombre:** Crear Visión
Usuario: Equipo proyecto	
Modificación de Historia Número:	
Prioridad en Negocio: Alto **(Alta / Media / Baja)**	**Puntos Estimados:**
Riesgo en Desarrollo: Alto **(Alto / Medio / Bajo)**	**Puntos Reales:**
Descripción: Permitir al usuario crear el documento de visión del proyecto registrando toda la información del proyecto (introducción, el problema, el producto, involucrados) se deben listar todas las características y requerimientos del proyecto.	
Observaciones: Las características y requerimientos deben listarse automáticamente.	

Historia de Usuario	
Número: 11	**Nombre:** Consultar Características
Usuario: Equipo proyecto	
Modificación de Historia Número:	
Prioridad en Negocio: Media **(Alta / Media / Baja)**	**Puntos Estimados:**
Riesgo en Desarrollo: Medio **(Alto / Medio / Bajo)**	**Puntos Reales:**
Descripción: Listar todas las características y permitir al usuario realizar la modificación de la información básica de cualquiera de forma rápida y eficiente en el menor tiempo posible o poder suprimirlas del proyecto	
Observaciones: Al eliminar una característica se debe eliminar todos los requerimientos asociados a ella y toda la información relacionada.	

Historia de Usuario	
Número: 12	**Nombre:** Consultar Requerimientos
Usuario: Equipo proyecto	
Modificación de Historia Número:	
Prioridad en Negocio: Media **(Alta / Media / Baja)**	**Puntos Estimados:**
Riesgo en Desarrollo: Medio **(Alto / Medio / Bajo)**	**Puntos Reales:**
Descripción: Listar todas los requerimientos y permitir al usuario realizar la modificación de la información básica de cualquiera de forma rápida y eficiente en el menor tiempo posible o poder suprimirlas del proyecto	
Observaciones: Al eliminar un requerimiento se debe eliminar todos los casos de uso asociados a él y toda la información relacionada.	

Historia de Usuario	
Número: 13	**Nombre:** Consultar Casos de Uso
Usuario: Equipo proyecto	
Modificación de Historia Número:	
Prioridad en Negocio: Media **(Alta / Media / Baja)**	**Puntos Estimados:**
Riesgo en Desarrollo: Medio **(Alto / Medio / Bajo)**	**Puntos Reales:**
Descripción: Listar todas los casos de uso y permitir al usuario realizar la modificación de la información básica de cualquiera de forma rápida y eficiente en el menor tiempo posible o poder suprimirlas del proyecto	
Observaciones: Al eliminar un Caso de uso se debe eliminar toda la información o elementos asociados a él.	

Historia de Usuario	
Número: 14	**Nombre:** Consultar Término
Usuario: Equipo proyecto	
Modificación de Historia Número:	
Prioridad en Negocio: Media **(Alta / Media / Baja)**	**Puntos Estimados:**
Riesgo en Desarrollo: Medio **(Alto / Medio / Bajo)**	**Puntos Reales:**
Descripción: Listar todos los términos creados en un proyecto de forma rápida y eficiente.	
Observaciones:	

Historia de Usuario	
Número: 15	**Nombre:** Eliminar Comentario
Usuario: Equipo proyecto	
Modificación de Historia Número:	
Prioridad en Negocio: Media **(Alta / Media / Baja)**	**Puntos Estimados:**
Riesgo en Desarrollo: Medio **(Alto / Medio / Bajo)**	**Puntos Reales:**
Descripción: Listar todos los comentarios y permitir al usuario eliminar los comentarios creados por él.	
Observaciones: El usuario solo puede eliminar sus cometarios.	

Historia de Usuario	
Número: 16	**Nombre:** Consultar Paquete
Usuario: Equipo proyecto	
Modificación de Historia Número:	
Prioridad en Negocio: Media **(Alta / Media / Baja)**	**Puntos Estimados:**
Riesgo en Desarrollo: Medio **(Alto / Medio / Bajo)**	**Puntos Reales:**
Descripción: Listar todos los términos elementos contenidos dentro de los paquetes.	
Observaciones:	

Historia de Usuario	
Número: 17	**Nombre:** Trazabilidad Características - Requerimientos
Usuario: Equipo proyecto	
Modificación de Historia Número:	
Prioridad en Negocio: Alta **(Alta / Media / Baja)**	**Puntos Estimados:**
Riesgo en Desarrollo: Alto **(Alto / Medio / Bajo)**	**Puntos Reales:**
Descripción: Permitir al usuario verificar la trazabilidad entre las características y requerimientos, cualquier modificación que se presente en ellos, debe ser reflejado en la matriz de trazabilidad, permitiendo al usuario aprobar los cambios.	
Observaciones: Se debe representar la trazabilidad a través de una matriz.	

Historia de Usuario	
Número: 18	**Nombre:** Modificar Visión
Usuario: Equipo proyecto	
Modificación de Historia Número:	
Prioridad en Negocio: Media **(Alta / Media / Baja)**	**Puntos Estimados:**
Riesgo en Desarrollo: Medio **(Alto / Medio / Bajo)**	**Puntos Reales:**
Descripción: Permitir al usuario modificar el documento de visión del proyecto en cualquier momento desde la herramienta web.	
Observaciones:	

Historia de Usuario	
Número: 19	**Nombre:** Modificar Características
Usuario: Equipo proyecto	
Modificación de Historia Número:	
Prioridad en Negocio: Media **(Alta / Media / Baja)**	**Puntos Estimados:**
Riesgo en Desarrollo: Medio **(Alto / Medio / Bajo)**	**Puntos Reales:**
Descripción: Permitir al usuario realizar la modificación de la información básica de cualquiera de forma rápida y eficiente en el menor tiempo posible, a través de una lista general. También se debe permitir modificar toda la información a través de una consulta personalizada.	
Observaciones: Mostrar toda la información de la característica en los campos del formulario, una vez el usuario elija la opción de modificar.	

Historia de Usuario	
Número: 20	**Nombre:** Modificar Requerimientos
Usuario: Equipo proyecto	
Modificación de Historia Número:	
Prioridad en Negocio: Media **(Alta / Media / Baja)**	**Puntos Estimados:**
Riesgo en Desarrollo: Medio **(Alto / Medio / Bajo)**	**Puntos Reales:**
Descripción: Permitir al usuario realizar la modificación de la información básica de cualquiera de forma rápida y eficiente en el menor tiempo posible, a través de una lista general. También se debe permitir modificar toda la información a través de una consulta personalizada.	
Observaciones: Mostrar toda la información del requerimiento en los campos del formulario, una vez el usuario elija la opción de modificar.	

Historia de Usuario	
Número: 21	**Nombre:** Modificar Casos de Uso
Usuario: Equipo proyecto	
Modificación de Historia Número:	
Prioridad en Negocio: Media **(Alta / Media / Baja)**	**Puntos Estimados:**
Riesgo en Desarrollo: Medio **(Alto / Medio / Bajo)**	**Puntos Reales:**
Descripción: Permitir al usuario realizar la modificación de la información básica de cualquiera de forma rápida y eficiente en el menor tiempo posible, a través de una lista general. También se debe permitir modificar toda la información a través de una consulta personalizada.	
Observaciones: Mostrar toda la información del caso de uso en los campos del formulario, una vez el usuario elija la opción de modificar.	

Historia de Usuario	
Número: 22	**Nombre:** Modificar Término
Usuario: Equipo proyecto	
Modificación de Historia Número:	
Prioridad en Negocio: Media **(Alta / Media / Baja)**	**Puntos Estimados:**
Riesgo en Desarrollo: Medio **(Alto / Medio / Bajo)**	**Puntos Reales:**
Descripción: Permitir al usuario realizar la modificación de la información básica de cualquier término de forma rápida y eficiente en el menor tiempo posible, a través de una lista general. También se debe permitir modificar toda la información a través de una consulta personalizada.	
Observaciones: Mostrar la descripción del término en el campo del formulario, una vez el usuario elija la opción de modificar.	

Historia de Usuario	
Número: 23	**Nombre:** Modificar Paquete
Usuario: Equipo proyecto	
Modificación de Historia Número:	
Prioridad en Negocio: Media **(Alta / Media / Baja)**	**Puntos Estimados:**
Riesgo en Desarrollo: Medio **(Alto / Medio / Bajo)**	**Puntos Reales:**
Descripción: Permitir al usuario realizar la modificación de la información básica de los paquetes creados por ellos dentro de un proyecto	
Observaciones: Mostrar la descripción del término en el campo del formulario, una vez el usuario elija la opción de modificar. No se debe permitir la modificación de paquetes creados por defecto dentro de un proyecto (Visión y Características, Requerimientos del Sistema, Casos de Usos)	

Historia de Usuario	
Número: 24	**Nombre:** Trazabilidad Requerimientos – Casos de Uso
Usuario: Equipo proyecto	
Modificación de Historia Número:	
Prioridad en Negocio: Alta **(Alta / Media / Baja)**	**Puntos Estimados:**
Riesgo en Desarrollo: Alto **(Alto / Medio / Bajo)**	**Puntos Reales:**
Descripción: Permitir al usuario verificar la trazabilidad entre los requerimientos y casos de uso, cualquier modificación que se presente en ellos, debe ser reflejado en la matriz de trazabilidad, permitiendo al usuario aprobar los cambios.	
Observaciones: Se debe representar la trazabilidad en una matriz.	

Historia de Usuario	
Número: 25	**Nombre:** Doc. Visión
Usuario: Equipo proyecto	
Modificación de Historia Número:	
Prioridad en Negocio: Media **(Alta / Media / Baja)**	**Puntos Estimados:**
Riesgo en Desarrollo: Medio **(Alto / Medio / Bajo)**	**Puntos Reales:**
Descripción: Permitir al usuario exportar el documento de visión a una herramienta de código abierto para crear documentos.	
Observaciones:	

Historia de Usuario	
Número: 26	**Nombre:** Crear documento de Requerimientos del Sistema
Usuario: Equipo proyecto	
Modificación de Historia Número:	
Prioridad en Negocio: Alta **(Alta / Media / Baja)**	**Puntos Estimados:**
Riesgo en Desarrollo: Alto **(Alto / Medio / Bajo)**	**Puntos Reales:**
Descripción: Permitir al usuario crear el documento de requerimientos del sistema registrando la información necesaria (introducción, requerimientos, calidad, interfaces, reglas, limitaciones, licencias, documentación) se deben listar todos los requerimientos del sistema.	
Observaciones: Los requerimientos del sistema se deben listar automáticamente.	

Historia de Usuario	
Número: 27	**Nombre:** Eliminar Características
Usuario: Equipo proyecto	
Modificación de Historia Número:	
Prioridad en Negocio: Media **(Alta / Media / Baja)**	**Puntos Estimados:**
Riesgo en Desarrollo: Medio **(Alto / Medio / Bajo)**	**Puntos Reales:**
Descripción: Listar todas las características y permitir al usuario eliminar las características de un proyecto.	
Observaciones: Al eliminar una característica se debe eliminar todos los requerimientos asociados a ella y toda la información relacionada. Se debe solicitar confirmación al usuario antes de eliminar una característica.	

Historia de Usuario	
Número: 28	**Nombre:** Eliminar Requerimientos
Usuario: Equipo proyecto	
Modificación de Historia Número:	
Prioridad en Negocio: Media **(Alta / Media / Baja)**	**Puntos Estimados:**
Riesgo en Desarrollo: Medio **(Alto / Medio / Bajo)**	**Puntos Reales:**
Descripción: Listar todos los requerimientos y permitir al usuario eliminar suprimirlos del proyecto.	
Observaciones: Al eliminar un requerimiento se debe eliminar todos los casos de uso asociados a él y toda la información relacionada. Se debe solicitar confirmación al usuario antes de eliminar un requerimiento.	

Historia de Usuario	
Número: 29	**Nombre:** Eliminar Casos de Uso
Usuario: Equipo proyecto	
Modificación de Historia Número:	**Iteración Asignada:** 1
Prioridad en Negocio: Media **(Alta / Media / Baja)**	**Puntos Estimados:**
Riesgo en Desarrollo: Medio **(Alto / Medio / Bajo)**	**Puntos Reales:**
Descripción: Listar todos los casos de uso y permitir al usuario eliminarlos del proyecto.	
Observaciones: Al eliminar un caso de uso se debe suprimir toda la información y elementos relacionados a él. Se debe solicitar confirmación al usuario antes de eliminar.	

Historia de Usuario	
Número: 30	**Nombre:** Eliminar Término
Usuario: Equipo proyecto	
Modificación de Historia Número:	
Prioridad en Negocio: Media **(Alta / Media / Baja)**	**Puntos Estimados:**
Riesgo en Desarrollo: Medio **(Alto / Medio / Bajo)**	**Puntos Reales:**
Descripción: Listar todos los términos y permitir al usuario eliminarlos del proyecto.	
Observaciones: Al eliminar un término se debe solicitar confirmación al usuario antes realizar la acción.	

Historia de Usuario	
Número: 31	**Nombre:** Eliminar Paquete
Usuario: Equipo proyecto	
Modificación de Historia Número:	
Prioridad en Negocio: Media **(Alta / Media / Baja)**	**Puntos Estimados:**
Riesgo en Desarrollo: Medio **(Alto / Medio / Bajo)**	**Puntos Reales:**
Descripción: Permitir eliminar los paquetes creados por los usuarios.	
Observaciones: Al eliminar un paquete se debe solicitar confirmación al usuario antes realizar la acción y eliminar todo los elementos que contiene.	

Historia de Usuario	
Número: 32	**Nombre:** Doc. Glosario
Usuario: Equipo proyecto	
Modificación de Historia Número:	
Prioridad en Negocio: Media **(Alta / Media / Baja)**	**Puntos Estimados:**
Riesgo en Desarrollo: Medio **(Alto / Medio / Bajo)**	**Puntos Reales:**
Descripción: Permitir al usuario ver todos los términos registrados en el proyecto y poder exportarlos a un documento de texto en una herramienta de código abierto.	
Observaciones:	

Historia de Usuario	
Número: 33	**Nombre:** Modificar documento de Requerimientos del Sistema.
Usuario: Equipo proyecto	
Modificación de Historia Número:	**Iteración Asignada:** 1
Prioridad en Negocio: Media **(Alta / Media / Baja)**	**Puntos Estimados:**
Riesgo en Desarrollo: Medio **(Alto / Medio / Bajo)**	**Puntos Reales:**
Descripción: Permitir a los usuarios modificar el documento de requerimientos del sistema perteneciente al proyecto.	
Observaciones:	

Historia de Usuario	
Número: 34	**Nombre:** Doc. Requerimientos del Sistema
Usuario: Equipo proyecto	
Modificación de Historia Número:	**Iteración Asignada:** 1
Prioridad en Negocio: Media **(Alta / Media / Baja)**	**Puntos Estimados:**
Riesgo en Desarrollo: Medio **(Alto / Medio / Bajo)**	**Puntos Reales:**
Descripción: Permitir al usuario exportar el documento de requerimientos del sistema a una herramienta de código abierto para crear documentos.	
Observaciones:	

Historia de Usuario	
Número: 35	**Nombre:** Herramienta de Gestión de Requerimientos
Usuario: Equipo proyecto	
Modificación de Historia Número:	**Iteración Asignada:** 1
Prioridad en Negocio: Media **(Alta / Media / Baja)**	**Puntos Estimados:**
Riesgo en Desarrollo: Medio **(Alto / Medio / Bajo)**	**Puntos Reales:**
Descripción: Permitir a los usuarios observar los cambios a través de una herramienta de código abierto integrada con la herramienta en desarrollo. Se debe mostrar los cambios y los usuarios que produjeron esos cambios organizados por proyectos.	
Observaciones:	

Historia de Usuario	
Número: 36	**Nombre:** Doc. Especificación de casos de uso
Usuario: Equipo proyecto	
Modificación de Historia Número:	
Prioridad en Negocio: Media **(Alta / Media / Baja)**	**Puntos Estimados:**
Riesgo en Desarrollo: Medio **(Alto / Medio / Bajo)**	**Puntos Reales:**
Descripción: Permitir al usuario ver la especificación de un caso de uso y poder exportarlos a un documento de texto en una herramienta de código abierto.	
Observaciones:	

Apéndice B – Plantillas

B.1 Documento de Visión [8]:

<Nombre del Proyecto>

Visión

Introducción
Posicionamiento

El Problema

[Breve descripción general del proyecto]

El problema	*[describir el problema]*
Afecta	*[Las partes interesadas (involucrados) afectadas por el problema]*
Impacta	*[¿Cuál es el impacto del Problema?]*
Solución	*[Resumen de los principales beneficios de una solución satisfactoria]*

Posición del producto
[Proporcionar un resumen general, al más alto nivel, la posición del producto y la intensión en el Mercado:]

Para	*[target de clientes]*
Quien	*[Resumen de la necesidad o de la oportunidad]*
Nombre del producto	*[nombre (categoria) del producto]*
Que	*[Resumen de los principales beneficios, es decir, la razón de peso para comprar el producto]*
A diferencia de	*[Principal alternativa competitiva]*
Nuestro Producto	*[Resumen de las principales diferencias]*

Descripción de los involucrados

Resumen de los involucrados

Nombre	Descripción	Responsabilidad
[Nombre de las partes interesadas (tipo).]	*[Describa brevemente las partes interesadas.]*	*[Resumir brevemente las responsabilidades de las partes interesadas]*

Ambiente de Usuario
[Detallar el ambiente de trabajo de los usuarios.]

Resumen del Producto

Característica	Prioridad	Iteraciones Planificadas

Requerimientos	Prioridad	Iteraciones Planificadas

B.2 Documento de Requerimiento del sistema [8]:

<Nombre del Proyecto>
Requerimientos

Introducción.

Requerimientos

Requerimientos	Prioridad	Iteraciones Planificadas

Calidad del Sistema

- Usabilidad:
- Fiabilidad:
- Rendimiento:
- Soporte:

Interfaces del Sistema
[Resumen general sobre las interfaces que deben ser apoyadas por la aplicación. Protocolos, puertos y direcciones lógicas, y así sucesivamente, de modo

que el software pueda ser desarrollado y conectarse con los requerimientos de la interfaz.]

Interfaz de Usuario
[Resumen general sobre las interfaces de usuario, desarrolladas en el sistema. La intención de esta sección es describir aspectos importantes sobre las interfaces de usuario]

Aspecto y Ambiente
[Una descripción de la esencia de la interfaz. Las peticiones del cliente sobre el estilo, los colores que deben utilizarse, el grado de interacción y así sucesivamente]

Diseño y Navegación:
[Resumen general del área de la pantalla].

Coherencia:

Personalización de Usuario:
[Resumen sobre cómo debería aparecer automáticamente para los usuarios los datos; o si es posible que los usuarios puedan personalizar el contenido].

Interfaces con sistemas externos o con dispositivos

Interfaces de software

[Componentes del sistema de software; algún componente reutilizado de otra aplicación o componentes que se están desarrollando para los otros subsistemas fuera del ámbito de ésta aplicación, pero con la que debe interactuar].

Interfaces de Hardware:

[Definir las interfaces hardware que van a ser soportados por el software, incluyendo estructura lógica, direcciones físicas, el comportamiento esperado, y así sucesivamente].

Interfaces de Comunicación:
[Describir interfaces de comunicaciones a otros sistemas o dispositivos tales como redes de área local, dispositivos de control remoto, y así sucesivamente].

Reglas del Negocio

[Aspectos que definir o limitar de la empresa. Las reglas del negocio son a menudo representadas como normas de producción].

Reglas

Limitaciones del Sistema.

Cumplimiento, Licencias del Sistema

Licencia de los requerimientos del sistema

[Definir la ejecución de cualquier concesión de licencias u restricción de uso en los requisitos que han de ser exhibidos por el software].

Jurídicos, derechos de autor y otros

[Describir cualquier cumplimiento legal necesario: renuncias, garantías, avisos de copyright, patentes, wordmark, marca, logotipo o las cuestiones relativas al cumplimiento de los programas informáticos].

Normas aplicables

Documentación del Sistema

[Describir como debe ser la documentación de usuario, sistemas de ayuda, ayuda sobre los anuncios, etc. Determinar quién será el responsable de la documentación].

B.3 Casos de Uso [8]:

<Nombre del proyecto>
Casos de Uso

Descripción

<Breve descripción del caso de uso>

Actores

<Nombre Actor 1 >

Precondiciones

<pre-condición 1>

Flujo Básico de Eventos

1. <Paso 1>
2. …
3. <Paso n>

Post-condiciones

B.4 Historias de Usuario:

Historia de Usuario	
Número: **Nombre:**	
Usuario:	
Modificación de Historia Número:	
Prioridad en Negocio: (Alta / Media / Baja)	**Puntos Estimados:**
Riesgo en Desarrollo: (Alto / Medio / Bajo)	**Puntos Reales:**
Descripción:	
Observaciones:	

Apéndice C – Componentes de Código

C.1 Componente para la capa de persistencia:

```
<?php
	include_once("class_conexion.php");
	include_once("class_operacion.php");
	class caracteristica extends operacion
	{
		private $nombre;
		private $descripcion;
		private $conexion;
		private $paquete;
		private $prioridad;
		private $tipo;
		private $estatus;
		private $estabilidad;
		private $riesgo;
		private $iteracionactual;
		private $iteracion;
		private $origen;
		private $contacto;
		private $solicitud;
		private $defecto;
		private $obsoleto;
		private $dificultad;
		public $id;
		public $usu;
		function __construct()
		{
			if (func_num_args() == 16)
```

```
{       $this->nombre = func_get_arg(0);
        $this->descripcion = func_get_arg(1);
        $this->paquete = func_get_arg(2);
        $this->prioridad = func_get_arg(3);
        $this->tipo = func_get_arg(4);
        $this->estatus = func_get_arg(5);
        $this->dificultad = func_get_arg(6);
        $this->estabilidad = func_get_arg(7);
        $this->riesgo = func_get_arg(8);
        $this->iteracion= func_get_arg(9);
        $this->origen = func_get_arg(10);
        $this->contacto = func_get_arg(11);
        $this->solicitud = func_get_arg(12);
        $this->defecto = func_get_arg(13);
        $this->obsoleto = func_get_arg(14);
        $this->iteracionactual= func_get_arg(15);
} $this->conexion = new conexion(); }

function destruir()
{   $this->conexion->cerrar(); }

function crear()
{
        $this->query("INSERT INTO CARACTERISTICA (ID_PAQUETE,
        NOMBRE_CARACTERISTICA,DESCRIPCION_CARACTERISTICA,PRIO
        RIDAD_CARACTERISTICA,TIPO_CARACTERISTICA,ESTATUS_CARA
        CTERISTICA,DIFICULTAD_CARACTERISTICA,ESTABILIDAD_CARA
        CTERISTICA,RIESGO_CARACTERISTICA,PLAN_ITER_CARACTERIST
        ICA,ITERACION_ACTUAL,ORIGEN_CARACTERISTICA,CONTACTO_
        CARACTERISTICA,SOLICITUD_CARACTERISTICA,DEFECTO_CARA
        CTERISTICA,OBSOLETO_CARACTERISTICA) VALUES ('$this-
        >paquete', '$this->nombre','$this->descripcion','$this->prioridad','$this-
        >tipo','$this->estatus','$this->dificultad','$this->estabilidad','$this-
        >riesgo','$this->iteracion','$this->iteracionactual','$this->origen','$this-
        >contacto','$this->solicitud','$this->defecto','$this->obsoleto')");      }
```

```
function consultar($proyecto){
$select = "SELECT * FROM CARACTERISTICA C, PAQUETE PA";
$where = " WHERE PA.ID_PROYECTO = '$proyecto' AND PA.ID_PAQUETE = C.ID_PAQUETE";
if ($this->id != "")
{
	$where .= " AND C.ID_CARACTERISTICA = '$this->id'";
}
if ($this->paquete != "")
{
	$where .= " AND C.ID_PAQUETE = '$this->paquete'";
}
if ($this->nombre != "")
{
	$where .= " AND C.NOMBRE_CARACTERISTICA LIKE '%$this->nombre%'";
}
if ($this->descripcion != "")
{
	$where .= " AND C.DESCRIPCION_CARACTERISTICA LIKE '%$this->descripcion%'";
}
if ($this->paquete != "")
{
	$where .= " AND C.ID_PAQUETE = '$this->paquete'";
}

$result = $this->query($select.$where);
if ($this->numRows($result) > 0)
	return $result;
else
	return false;
```

```
}

function mod_bas()
{
        session_start();

        $this->usu          = $_SESSION['id_usuario'];

        $this->query("UPDATE CARACTERISTICA SET
        ID_PAQUETE = '$this->paquete',
        PRIORIDAD_CARACTERISTICA = '$this->prioridad',
        ESTATUS_CARACTERISTICA = '$this->estatus',
        DIFICULTAD_CARACTERISTICA = '$this->dificultad',
        ESTABILIDAD_CARACTERISTICA = '$this->estabilidad',
        CONTACTO_CARACTERISTICA = '$this->contacto'
        WHERE ID_CARACTERISTICA = '$this->id'");

        $this->query("INSERT INTO CAMBIO (ID_CARACTERISTICA, DESCRIPCION_CAMBIO,
ESTATUS_CAMBIO, ID_USUARIO)
        VALUES ('$this->id', 'Se modificó la característica','Por Aprobar','$this->usu')");
}

function modificar_car()
{
        session_start();

        $this->usu          = $_SESSION['id_usuario'];

        $this->query("UPDATE CARACTERISTICA SET
        ID_PAQUETE = '$this->paquete',
```

```
                NOMBRE_CARACTERISTICA = '$this->nombre',
                DESCRIPCION_CARACTERISTICA = '$this->descripcion',
                PRIORIDAD_CARACTERISTICA = '$this->prioridad',
                TIPO_CARACTERISTICA = '$this->tipo',
                ESTATUS_CARACTERISTICA = '$this->estatus',
                DIFICULTAD_CARACTERISTICA = '$this->dificultad',
                ESTABILIDAD_CARACTERISTICA = '$this->estabilidad',
                RIESGO_CARACTERISTICA = '$this->riesgo',
                PLAN_ITER_CARACTERISTICA = '$this->iteracion',
                ITERACION_ACTUAL = '$this->iteracionactual',
                ORIGEN_CARACTERISTICA = '$this->origen',
                CONTACTO_CARACTERISTICA = '$this->contacto',
                SOLICITUD_CARACTERISTICA = '$this->solicitud',
                DEFECTO_CARACTERISTICA = '$this->defecto',
                OBSOLETO_CARACTERISTICA = '$this->obsoleto'
                WHERE ID_CARACTERISTICA = '$this->id'");

                $this->query("INSERT INTO CAMBIO (ID_CARACTERISTICA, DESCRIPCION_CAMBIO,
ESTATUS_CAMBIO, ID_USUARIO)
                VALUES ('$this->id', 'Se modificó la característica','Por Aprobar','$this->usu')");
        }

        function eliminar($idcaracteristica)
        {

                $this->query("DELETE FROM CARACTERISTICA WHERE ID_CARACTERISTICA =
'$idcaracteristica'");

        }

        function cambiocaracteristica($id)
```

```
	{
		$result = $this->query("SELECT * FROM CAMBIO WHERE ID_CARACTERISTICA = '$id' AND ESTATUS_CAMBIO = 'Por Aprobar'");
		if ($this->numRows($result) > 0){

		$row = $this->fetchArray($result);
		return $row;

		}else {
			return false;
		}
	}
		function modcambiocaracteristica($idc)
	{
		$result = $this->query("UPDATE CAMBIO SET ESTATUS_CAMBIO = 'Aprobado' WHERE ID_CAMBIO = '$idc'");

	}
}
?>
```

C.2 XMLHttpRequest

```
	function createXMLHttpRequest()
	{
		if (window.ActiveXObject)
		{
			return new ActiveXObject("Microsoft.XMLHTTP");
		}
		else if (window.XMLHttpRequest)
		{
```

```
            return new XMLHttpRequest();
        }
    }
```

C.3 AJAX

```
function createXMLHttpRequest()
{       if (window.ActiveXObject)
        {   return new ActiveXObject("Microsoft.XMLHTTP");   }
        else if (window.XMLHttpRequest)
        {   return new XMLHttpRequest();   }  }
function llamarasincrono(url, id_contenedor)
{       var pagina_requerida = false
        var pagina_requerida = createXMLHttpRequest();
        pagina_requerida.onreadystatechange=function()
        {  cargarpagina(pagina_requerida, id_contenedor)          }
        pagina_requerida.open('GET', url, true);
        pagina_requerida.send(null);
}
function cargarpagina(pagina_requerida, id_contenedor)
{   if (pagina_requerida.readyState==1)
        {
            document.getElementById(id_contenedor).innerHTML = "<div align='center'
            style='width:100%; height:100%'><img src='../imagenes/loading.gif' /></div>";
        }
        if (pagina_requerida.readyState==4)
        {   if(pagina_requerida.status==200)
            {document.getElementById(id_contenedor).innerHTML =
            pagina_requerida.responseText;
            }
            else if(pagina_requerida.status==404)
            {  id_contenedor.innerHTML = "La direccion no existe";  }
```

```
                    else {  id_contenedor.innerHTML = "Error: ".pagina_requerida.status; }  }  }
function enviardata(_pagina, valorget, valorpost, capa)
{   ajax = createXMLHttpRequest();
    if(valorpost!="")
        {   ajax.open("POST", _pagina+"?"+valorpost, true);     }
    else {      ajax.open("GET", _pagina+"?"+valorget, true);   }
                ajax.onreadystatechange=function() {    cargarpagina(ajax, capa);   }
                if(valorpost!="") {
                   ajax.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");
                   ajax.send(valorpost);   }
                else {  ajax.send(null);   }  }
                        function toggle(capa)
                        {   if ((document.getElementById(capa).style.display == 'none'))
                                { document.getElementById(capa).style.display = '';         }
                            else {  document.getElementById(capa).style.display = 'none';   }   }
                                function agregar_paso(id_contenedor, numero){
                                var nodo = document.createElement("div");
                                nodo.innerHTML = "<div id='" + numero +"' class='paso'><span class='celda' id='texto" + numero + "'>Paso " + numero + "</span><span class='celda' style='width:70%'><textarea name='campo" + numero + "' id='c" + numero + "' cols='100%' rows='2'></textarea></span><span class='celda' style='width:10%'><a id='sp" + numero + "' href='javascript: crear_sub_paso(" + numero + ")'>sub paso</a></span><span class='celda' style='width:10%'><a id='borra" + numero + "' href='javascript: borrar_paso(" + numero + "); numerar()'>eliminar</a></span><input id='s" + numero + "' type='hidden' value='0' /></div>";
                var actual = document.getElementById(id_contenedor);
                if (actual.childNodes.length == 0) {  actual.innerHTML = (nodo.innerHTML);}
                else {  actual.appendChild(nodo); }  }
function borrar_paso(numero)
{       var borrar = document.getElementById(numero);
        borrar.parentNode.removeChild(borrar); }
function agregar_sub_paso(padre, hijo)
{   var nodo = document.createElement("div");
        nodo.innerHTML = "<div id='" + padre +"." + hijo + "' class='sub_paso'><span class='celda' id='texto" + padre +"." + hijo + "'>Paso " + padre +"." + hijo + "</span><span class='celda'
```

```
style='width:70%'><textarea name='campo" + padre +"." + hijo + "' id='c" + padre +"." + hijo + "' cols='100%' rows='2'></textarea></span><span class='celda' style='width:10%'><a id='borra" + padre +"." + hijo + "' href='javascript: borrar_paso(" + padre + "." + hijo + "); numerar();'>eliminar</a></span></div>";

                var actual = document.getElementById(padre);

                actual.appendChild(nodo); }
```

C.4 JAVASCRIPT

```
function checkrequired(which)

{       var pass=true;

        if (document.images)

        { for (i=0;i<which.length;i++)

                { var tempobj=which.elements[i];

                  if (tempobj.name.substring(0,8)=="required")

                { if (((tempobj.type=="text"||tempobj.type=="textarea" || tempobj.type=="password")&&
(tempobj.value==""||tempobj.value==tempobj.title))||(tempobj.type.toString().charAt(0)=="s"&&
tempobj.selectedIndex==0))

                { pass=false;

                break; } } } }

        if (!pass)

        {       shortFieldName=tempobj.name.substring(8,30).toUpperCase();

                alert("Por favor asegurese de que el campo "+shortFieldName+" haya sido llenado
apropiadamente.");

                tempobj.focus();

                return false;       }

        else

                return true; }

function iguales(campo1, campo2)

{       if (document.getElementById(campo1).value == document.getElementById(campo2).value)

        return true;

        else

        {       alert('Asegurese de que los campos tengan el mismo valor donde sea necesario');

                return false;       } }

function IsNumeric(sText)
```

```
{  var ValidChars = "0123456789";
  var IsNumber=true;
  var Char;
  for (i = 0; i < document.getElementById(sText).value.length && IsNumber == true; i++){
    Char = document.getElementById(sText).value.charAt(i);
    if (ValidChars.indexOf(Char) == -1){
      IsNumber = false;
            alert("Este campo solo acepta valores numericos");
            document.getElementById(sText).value = "";
    } }
  return IsNumber;
}
function escribe(mensaje)
{ cont =0;
 while (cont<mensaje.length)
        { letra = mensaje.substring(cont,cont+1)
         document.write(letra+"<br>")
         cont+=1
} }
```

C.5 Componentes para la capa de Negocio:

```
<?php
        session_start();
        include_once("../clases/class_requerimiento.php");
        include_once("../clases/class_conexion.php");
        $p = $_SESSION['id_proyecto'];
         $obj = new requerimiento($_POST['requirednombre'], $_POST['requireddescripcion'], $_POST['paquete'],
$_POST['prioridad'], $_POST['tipo'], $_POST['estatus'], $_POST['dificultad'], $_POST['estabilidad'],
$_POST['riesgo'], $_POST['requirediteracion'], $_POST['requiredcontacto'], $_POST['requiredsolicitud'],
$_POST['requireddefecto'], $_POST['obsoleto'],
$_POST['requirediteracion_actual'],$_POST['prioridadinter'],$_POST['caracteristica']);
        $obj->crear();
```

```
        $obj->destruir();
        header("Location: ../proyectos/consultarProyecto.php?id=$p");
?>
```

C.6 ODF:

```
<?php
require 'Segment.php';
class OdfException extends Exception
{}
class Odf
    {
      const DELIMITER_LEFT = '{';
      const DELIMITER_RIGHT = '}';
      const PIXEL_TO_CM = 0.026458333;
      private $file;
      private $contentXml;
      private $tmpfile;
      private $images = array();
      private $vars = array();
      private $segments = array();
      public function __construct($filename)
      {
        if (! class_exists('ZipArchive')) {
          throw new OdfException('Zip extension not loaded - check your php settings, PHP5.2 minimum
with zip extension
    is required');    }
        $this->file = new ZipArchive();
        if ($this->file->open($filename) !== true) {
          throw new OdfException("Error while Opening the file '$filename' - Check your odt file");
        }
```

```
            if (($this->contentXml = $this->file->getFromName('content.xml')) === false) {
                throw new OdfException("Nothing to parse - check that the content.xml file is correctly formed");
            }
            $tmp = tempnam(null, md5(uniqid())) . '.odt';
            copy($filename, $tmp);
            $this->tmpfile = $tmp;
        }
        public function setVars($key, $value, $encode = true)
        {
            if (strpos($this->contentXml, self::DELIMITER_LEFT . $key . self::DELIMITER_RIGHT) === false)
{
                throw new OdfException("var $key not found in the document");
            }
            $value = $encode ? utf8_encode(htmlspecialchars($value)) : utf8_encode($value);
            $this->vars[self::DELIMITER_LEFT . $key . self::DELIMITER_RIGHT] = $value;
            return $this;
        }
        public function setImage($key, $value)
        {
            $filename = strtok(strrchr($value, '/'), '/.');
            $file = substr(strrchr($value, '/'), 1);
            $size = @getimagesize($value);
            if ($size === false) {
                throw new OdfException("Invalid image");
            }
list ($width, $height) = $size;
$width *= self::PIXEL_TO_CM;
$height *= self::PIXEL_TO_CM;
$xml = <<<IMG
```

```
<draw:frame draw:style-name="fr1" draw:name="$filename" text:anchor-type="char" svg:width="{$width}cm" svg:height="{$height}cm" draw:z-index="3"><draw:image xlink:href="Pictures/$file" xlink:type="simple" xlink:show="embed" xlink:actuate="onLoad"/></draw:frame>IMG;

$this->images[$value] = $file;

$this->setVars($key, $xml, false);

return $this;

        }

        private function _parse()

        {

          $this->contentXml = str_replace(array_keys($this->vars), array_values($this->vars), $this->contentXml);

        }

        public function mergeSegment(Segment $segment)

        {

        if (! array_key_exists($segment->getName(), $this->segments)) {

            throw new OdfException($segment->getName() . 'cannot be parsed, has it been set yet ?');

          }

          $string = $segment->getName();

          $reg = '@<text:p[^>]*>\[!--\sBEGIN\s' . $string . '\s--\](.*)\[!--.+END\s' . $string . '\s--\]<\/text:p>@smU';

          $this->contentXml = preg_replace($reg, $segment->getXmlParsed(), $this->contentXml);

          return $this;

        }

        public function printVars()

        {

          return print_r('<pre>' . print_r($this->vars, true) . '</pre>', true);

        }

      public function __toString()

        {

          return $this->contentXml;

        }

        public function printDeclaredSegments()
```

```
    {
        return '<pre>' . print_r(implode(' ', array_keys($this->segments)), true) . '</pre>';
    }
    public function setSegment($segment)
    {
        if (array_key_exists($segment, $this->segments)) {
return $this->segments[$segment];
        }
        $reg = "#\[!--\sBEGIN\s$segment\s--\]<\/text:p>(.*)<text:p\s.*>\[!--\sEND\s$segment\s--\]#sm";
        if (preg_match($reg, html_entity_decode($this->contentXml), $m) == 0) {
            throw new OdfException("'$segment' segment not found in the document");
        }
        $this->segments[$segment] = new Segment($segment, $m[1]);
        return $this->segments[$segment];
    }
    public function saveToDisk($file = null)
    {
        if ($file !== null && is_string($file)) {
            $this->file->open($this->tmpfile, ZIPARCHIVE::CREATE);
            $this->_save();
            copy($this->tmpfile, $file);
        } else {
            $this->_save();
        }
    }
    private function _save()
    {
        $this->_parse();
        if (! $this->file->addFromString('content.xml', $this->contentXml)) {
            throw new OdfException('Error during file export');
```

```
}
    foreach ($this->images as $imageKey => $imageValue) {
      $this->file->addFile($imageKey, 'Pictures/' . $imageValue);
    }
    $this->file->close(); // seems to bug on windows CLI sometimes
  }
  public function exportAsAttachedFile()
  {
    $this->file->open($this->tmpfile, ZIPARCHIVE::CREATE);
    $this->_save();
    if (headers_sent($filename, $linenum)) {
      throw new OdfException("headers already sent ($filename at $linenum)");
    }
    header('Content-type: multipart/x-zip');
header("Content-Disposition: attachment; filename=" . md5(uniqid()) . ".odt");
    readfile($this->tmpfile);
  }
}
```

C.7 Mecanismo de Integración PHP – OpenOffice Writer:

```
<?php
session_start();
require_once('../library/odf.php');
include_once('../../clases/class_proyecto.php');
include_once("../../clases/class_conexion.php");
$proyecto = new proyecto();
$proyecto->id = $_SESSION['id_proyecto'];
$result = $proyecto->consultar();
$row = $proyecto->fetchArray($result);
```

```
$conexion = new conexion();

$p = $_SESSION['id_proyecto'];

$nombrep = $row['NOMBRE_PROYECTO'];

$odf = new odf("caracteristica.odt");
if($conexion->numRows($result2)>0){
        $nomb = $odf->setSegment('nombres');
        $orig = $odf->setSegment('org');
        $prio = $odf->setSegment('pri');
        $estat = $odf->setSegment('est');
        $difi = $odf->setSegment('dif');
        $estab = $odf->setSegment('esta');
        $ries = $odf->setSegment('rie');
        $perso = $odf->setSegment('per');
        $tipo = $odf->setSegment('tip');
        //$desc = $odf->setSegment('des');

        while ( $row2 = $conexion->fetchArray($result2)){
                $nombre = $row2 ['NOMBRE_CARACTERISTICA'];
                $prioridad = $row2['PRIORIDAD_CARACTERISTICA'];
                $estatus = $row2['ESTATUS_CARACTERISTICA'];
                $dificultad = $row2['DIFICULTAD_CARACTERISTICA'];
                $estabilidad = $row2['ESTABILIDAD_CARACTERISTICA'];
                $riesgo = $row2['RIESGO_CARACTERISTICA'];
                $persona = $row2['CONTACTO_CARACTERISTICA'];
                $tipor = $row2['TIPO_CARACTERISTICA'];
                $origen = $row2['ORIGEN_CARACTERISTICA'];
                $nomb->setVar('n', $nombre);
```

```
                $orig->setVar('c', $origen);
                $prio->setVar('p', $prioridad);
                $estat->setVar('e', $estatus);
                $difi->setVar('d', $dificultad);
                $estab->setVar('es', $estabilidad);
                $ries->setVar('r', $riesgo);
                $perso->setVar('pe', $persona);
                $tipo->setVar('t', $tipor);
                $nomb->merge();
                $orig->merge();
                $prio->merge();
                $estat->merge();
                $difi->merge();
                $estab->merge();
                $ries->merge();
                $perso->merge();
                $tipo->merge();
        }
        $odf->mergeSegment($nomb);
        $odf->mergeSegment($orig);
        $odf->mergeSegment($prio);
        $odf->mergeSegment($estat);
        $odf->mergeSegment($difi);
        $odf->mergeSegment($estab);
        $odf->mergeSegment($ries);
        $odf->mergeSegment($perso);
        $odf->mergeSegment($tipo);
}
$odf->setVars('nombrep',$nombrep);
$odf->exportAsAttachedFile(); ?>
```

C.8 Script de la Base de Datos:

```
DROP DATABASE IF EXISTS `tesis`;

CREATE DATABASE `tesis`
  CHARACTER SET 'latin1'
  COLLATE 'latin1_swedish_ci';

USE `tesis`;

#
# Structure for the `actor` table :
#

DROP TABLE IF EXISTS `actor`;

CREATE TABLE `actor` (
 `ID_ACTOR` int(11) NOT NULL auto_increment,
 `NOMBRE_ACTOR` varchar(50) default NULL,
 PRIMARY KEY (`ID_ACTOR`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

#
# Structure for the `proyecto` table :
#

DROP TABLE IF EXISTS `proyecto`;

CREATE TABLE `proyecto` (
 `ID_PROYECTO` int(11) NOT NULL auto_increment,
 `NOMBRE_PROYECTO` varchar(35) NOT NULL,
 `DESCRIPCION_PROYECTO` varchar(150) default NULL,
 PRIMARY KEY (`ID_PROYECTO`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

#
# Structure for the `paquete` table :
#

DROP TABLE IF EXISTS `paquete`;

CREATE TABLE `paquete` (
 `ID_PAQUETE` int(11) NOT NULL auto_increment,
 `ID_PROYECTO` int(11) default NULL,
 `NOMBRE_PAQUETE` varchar(35) NOT NULL,
 `IMAGEN` varchar(50) default NULL,
 `DESCRIPCION_PAQUETE` text,
 `INFO_PAQUETE` varchar(60) default NULL,
 PRIMARY KEY (`ID_PAQUETE`),
 KEY `FK_FORMADO` (`ID_PROYECTO`),
 CONSTRAINT `FK_FORMADO` FOREIGN KEY (`ID_PROYECTO`) REFERENCES `proyecto`
(`ID_PROYECTO`) ON DELETE SET NULL ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

#
```

```
# Structure for the `caracteristica` table :
#

DROP TABLE IF EXISTS `caracteristica`;

CREATE TABLE `caracteristica` (
 `ID_CARACTERISTICA` int(11) NOT NULL auto_increment,
 `ID_PAQUETE` int(11) default NULL,
 `NOMBRE_CARACTERISTICA` varchar(35) NOT NULL,
 `DESCRIPCION_CARACTERISTICA` text,
 `PRIORIDAD_CARACTERISTICA` varchar(20) default NULL,
 `TIPO_CARACTERISTICA` varchar(40) default NULL,
 `ESTATUS_CARACTERISTICA` varchar(25) default NULL,
 `DIFICULTAD_CARACTERISTICA` varchar(20) default NULL,
 `ESTABILIDAD_CARACTERISTICA` varchar(20) default NULL,
 `RIESGO_CARACTERISTICA` varchar(20) default NULL,
 `PLAN_ITER_CARACTERISTICA` int(11) default NULL,
 `ITERACION_ACTUAL` int(11) default NULL,
 `ORIGEN_CARACTERISTICA` varchar(30) default NULL,
 `CONTACTO_CARACTERISTICA` varchar(50) default NULL,
 `SOLICITUD_CARACTERISTICA` text,
 `DEFECTO_CARACTERISTICA` text,
 `OBSOLETO_CARACTERISTICA` char(2) default NULL,
 PRIMARY KEY (`ID_CARACTERISTICA`),
 KEY `FK_RELATIONSHIP_4` (`ID_PAQUETE`),
 CONSTRAINT `FK_RELATIONSHIP_4` FOREIGN KEY (`ID_PAQUETE`) REFERENCES `paquete`
(`ID_PAQUETE`) ON DELETE SET NULL ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

#
# Structure for the `requerimiento` table :
#

DROP TABLE IF EXISTS `requerimiento`;

CREATE TABLE `requerimiento` (
 `ID_REQUERIMIENTO` int(11) NOT NULL auto_increment,
 `ID_PAQUETE` int(11) default NULL,
 `ID_CARACTERISTICA` int(11) default NULL,
 `NOMBRE_REQUERIMIENTO` varchar(100) default NULL,
 `DESCRIPCION_REQUERIMIENTO` text,
 `TIPO_REQUERIMIENTO` varchar(20) default NULL,
 `PRIORIDAD_REQUERIMIENTO` varchar(20) default NULL,
 `ESTATUS_REQUERIMIENTO` varchar(20) default NULL,
 `DIFICULTAD_REQUERIMIENTO` varchar(20) default NULL,
 `ESTABILIDAD_REQUERIMIENTO` varchar(20) default NULL,
 `RIESGO_REQUERIMIENTO` varchar(20) default NULL,
 `NOMBRECONTACTO_REQUERIMIENTO` varchar(50) default NULL,
 `ITERACION_PLAN_REQUERIMIENTO` int(20) default NULL,
 `ITERACION_ACTUAL_REQUERIMIENTO` int(20) default NULL,
 `SOLICITUD_REQUERIMIENTO` text,
 `DEFECTO_REQUERIMIENTO` text,
 `OBSOLETO_REQUERIMIENTO` text,
 `PRIORIDAD_INTER_REQUERIMIENTO` text,
 PRIMARY KEY (`ID_REQUERIMIENTO`),
 KEY `FK_RELATIONSHIP_5` (`ID_PAQUETE`),
 KEY `FK_RELATIONSHIP_7` (`ID_CARACTERISTICA`),
```

```
 CONSTRAINT `requerimiento_fk` FOREIGN KEY (`ID_PAQUETE`) REFERENCES `paquete` (`ID_PAQUETE`)
ON DELETE CASCADE ON UPDATE SET NULL,
 CONSTRAINT `requerimiento_fk1` FOREIGN KEY (`ID_CARACTERISTICA`) REFERENCES `caracteristica`
(`ID_CARACTERISTICA`) ON DELETE CASCADE ON UPDATE SET NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

#
# Structure for the `caso_de_uso` table :
#

DROP TABLE IF EXISTS `caso_de_uso`;

CREATE TABLE `caso_de_uso` (
 `ID_CU` int(11) NOT NULL auto_increment,
 `ID_PAQUETE` int(11) default NULL,
 `ID_REQUERIMIENTO` int(11) default NULL,
 `NOMBRE_CU` varchar(50) default NULL,
 `PREC_CU` text,
 `POST_CU` text,
 `DESCRIPCION_CU` text,
 `PRIORIDAD_CU` varchar(20) default NULL,
 `ESTATUS_CU` varchar(20) default NULL,
 `DIFICULTAD_CU` varchar(20) default NULL,
 `ESTABILIDAD_CU` varchar(20) default NULL,
 `RIESGO_CU` varchar(20) default NULL,
 `OBSOLETO_CU` varchar(20) default NULL,
 `ARQUITECTURA` varchar(20) default NULL,
 PRIMARY KEY (`ID_CU`),
 KEY `FK_RELATIONSHIP_6` (`ID_PAQUETE`),
 KEY `FK_RELATIONSHIP_8` (`ID_REQUERIMIENTO`),
 CONSTRAINT `FK_RELATIONSHIP_6` FOREIGN KEY (`ID_PAQUETE`) REFERENCES `paquete`
(`ID_PAQUETE`) ON DELETE SET NULL ON UPDATE CASCADE,
 CONSTRAINT `FK_RELATIONSHIP_8` FOREIGN KEY (`ID_REQUERIMIENTO`) REFERENCES
`requerimiento` (`ID_REQUERIMIENTO`) ON DELETE SET NULL ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

#
# Structure for the `actor_cu` table :
#

DROP TABLE IF EXISTS `actor_cu`;

CREATE TABLE `actor_cu` (
 `ID_CU` int(11) default NULL,
 `ID_ACTOR` int(11) default NULL,
 KEY `ID_CU` (`ID_CU`),
 KEY `ID_ACTOR` (`ID_ACTOR`),
 CONSTRAINT `actor_cu_fk` FOREIGN KEY (`ID_CU`) REFERENCES `caso_de_uso` (`ID_CU`) ON DELETE
CASCADE,
 CONSTRAINT `actor_cu_fk1` FOREIGN KEY (`ID_ACTOR`) REFERENCES `actor` (`ID_ACTOR`) ON
DELETE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

#
# Structure for the `usuario` table :
#
```

```
DROP TABLE IF EXISTS `usuario`;

CREATE TABLE `usuario` (
 `ID_USUARIO` int(11) NOT NULL auto_increment,
 `NOMBRE_USUARIO` varchar(35) NOT NULL,
 `CLAVE_USUARIO` varchar(32) NOT NULL,
 `PREGUNTA_S` varchar(15) NOT NULL,
 `RESPUESTA_S` varchar(15) NOT NULL,
 `LOGIN_USUARIO` varchar(20) NOT NULL,
 PRIMARY KEY (`ID_USUARIO`),
 UNIQUE KEY `NOMBRE_USUARIO` (`NOMBRE_USUARIO`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

#
# Structure for the `cambio` table :
#

DROP TABLE IF EXISTS `cambio`;

CREATE TABLE `cambio` (
 `ID_CAMBIO` int(11) NOT NULL auto_increment,
 `ID_REQUERIMIENTO` int(11) default NULL,
 `ID_CU` int(11) default NULL,
 `ID_CARACTERISTICA` int(11) default NULL,
 `DESCRIPCION_CAMBIO` text,
 `ESTATUS_CAMBIO` varchar(20) default NULL,
 `ID_USUARIO` int(11) default NULL,
 PRIMARY KEY (`ID_CAMBIO`),
 KEY `ID_REQUERIMENTO` (`ID_REQUERIMIENTO`),
 KEY `ID_CU` (`ID_CU`),
 KEY `ID_CARACTERISTICA` (`ID_CARACTERISTICA`),
 KEY `ID_USUARIO` (`ID_USUARIO`),
 CONSTRAINT `cambio_fk` FOREIGN KEY (`ID_REQUERIMIENTO`) REFERENCES `requerimiento`
(`ID_REQUERIMIENTO`) ON DELETE SET NULL ON UPDATE SET NULL,
 CONSTRAINT `cambio_fk1` FOREIGN KEY (`ID_CU`) REFERENCES `caso_de_uso` (`ID_CU`) ON DELETE
SET NULL ON UPDATE SET NULL,
 CONSTRAINT `cambio_fk2` FOREIGN KEY (`ID_CARACTERISTICA`) REFERENCES `caracteristica`
(`ID_CARACTERISTICA`) ON DELETE SET NULL ON UPDATE SET NULL,
 CONSTRAINT `cambio_fk3` FOREIGN KEY (`ID_USUARIO`) REFERENCES `usuario` (`ID_USUARIO`) ON
DELETE SET NULL ON UPDATE SET NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

#
# Structure for the `comentario` table :
#

DROP TABLE IF EXISTS `comentario`;

CREATE TABLE `comentario` (
 `ID_COMENTARIO` int(11) NOT NULL auto_increment,
 `NOMBRE_COMENTARIO` varchar(100) default NULL,
 `DESCRIPCION_COMENTARIO` varchar(20) default NULL,
 `ID_CU` int(11) default NULL,
 `ID_REQUERIMIENTO` int(11) default NULL,
 `ID_CARACTERISTICA` int(11) default NULL,
 `ID_USUARIO` int(11) default NULL,
 PRIMARY KEY (`ID_COMENTARIO`),
```

```
 KEY `ID_CU` (`ID_CU`),
 KEY `ID_REQUERIMIENTO` (`ID_REQUERIMIENTO`),
 KEY `ID_CARACTERISTICA` (`ID_CARACTERISTICA`),
 KEY `ID_USUARIO` (`ID_USUARIO`),
 CONSTRAINT `comentario_fk` FOREIGN KEY (`ID_CU`) REFERENCES `caso_de_uso` (`ID_CU`) ON
DELETE SET NULL ON UPDATE CASCADE,
 CONSTRAINT `comentario_fk1` FOREIGN KEY (`ID_REQUERIMIENTO`) REFERENCES `requerimiento`
(`ID_REQUERIMIENTO`) ON DELETE SET NULL ON UPDATE CASCADE,
 CONSTRAINT `comentario_fk2` FOREIGN KEY (`ID_CARACTERISTICA`) REFERENCES `caracteristica`
(`ID_CARACTERISTICA`) ON DELETE SET NULL ON UPDATE CASCADE,
 CONSTRAINT `comentario_fk3` FOREIGN KEY (`ID_USUARIO`) REFERENCES `usuario` (`ID_USUARIO`)
ON DELETE SET NULL ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

#
# Structure for the `documento_req` table :
#

DROP TABLE IF EXISTS `documento_req`;

CREATE TABLE `documento_req` (
 `ID_DOC_REQ` int(11) NOT NULL auto_increment,
 `DESCRIPCION_DOC_REQ` text,
 `USABILIDAD_DOC_REQ` text,
 `FIABILIDAD_DOC_REQ` text,
 `RENDIMIENTO_DOC_REQ` text,
 `SOPORTE_DOC_REQ` text,
 `INTERFAZ_DOC_REQ` text,
 `INTER_USU_DOC_REQ` text,
 `VISION_DOC_REQ` text,
 `DISENO_DOC_REQ` text,
 `COHERENCIA_DOC_REQ` text,
 `PERSONALIZACION_DOC_REQ` text,
 `INTER_EXT_DOC_REQ` text,
 `SOFTWARE_INTER_DOC_REQ` text,
 `HARDWARE_INTER_DOC_REQ` text,
 `COMUNICACION_INTER_DOC_REQ` text,
 `REGLA_NEGOCIO_DOC_REQ` text,
 `REGLA_CLASES_DOC_REQ` text,
 `LIMITACION_DOC_REQ` text,
 `LICENCIA_DOC_REQ` text,
 `JURIDICO_DOC_REQ` text,
 `STANDARDS_DOC_REQ` text,
 `DOCUMENTO_DOC_REQ` text,
 `ID_PAQUETE` int(11) default NULL,
 PRIMARY KEY (`ID_DOC_REQ`),
 KEY `ID_PAQUETE` (`ID_PAQUETE`),
 CONSTRAINT `documento_req_fk` FOREIGN KEY (`ID_PAQUETE`) REFERENCES `paquete`
(`ID_PAQUETE`) ON DELETE SET NULL ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=latin1;


#
# Structure for the `flujo_cu` table :
#

DROP TABLE IF EXISTS `flujo_cu`;
```

```
CREATE TABLE `flujo_cu` (
 `ID_FLUJO` int(11) NOT NULL auto_increment,
 `NOMBRE_FLUJO` varchar(100) default NULL,
 `DESCRIPCION_FLUJO` varchar(1000) default NULL,
 `TIPO_FLUJO` varchar(100) default NULL,
 `ID_CU` int(11) default NULL,
 `ID_FLUJO_FK` int(11) default NULL,
 PRIMARY KEY (`ID_FLUJO`),
 KEY `ID_CU` (`ID_CU`),
 KEY `ID_FLUJO_FK` (`ID_FLUJO_FK`),
 CONSTRAINT `flujo_cu_fk` FOREIGN KEY (`ID_CU`) REFERENCES `caso_de_uso` (`ID_CU`) ON DELETE
CASCADE ON UPDATE CASCADE,
 CONSTRAINT `flujo_cu_fk1` FOREIGN KEY (`ID_FLUJO_FK`) REFERENCES `flujo_cu` (`ID_FLUJO`) ON
DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

#
# Structure for the `glosario` table :
#

DROP TABLE IF EXISTS `glosario`;

CREATE TABLE `glosario` (
 `ID_GLOSARIO` int(11) NOT NULL auto_increment,
 `NOMBRE_GLOSARIO` varchar(50) default NULL,
 `DESCRIPCION_GLOSARIO` varchar(500) default NULL,
 `ID_PAQUETE` int(11) default NULL,
 PRIMARY KEY (`ID_GLOSARIO`),
 KEY `ID_PAQUETE` (`ID_PAQUETE`),
 CONSTRAINT `glosario_fk` FOREIGN KEY (`ID_PAQUETE`) REFERENCES `paquete` (`ID_PAQUETE`) ON
DELETE SET NULL ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=latin1;


#
# Structure for the `usuario_proyecto` table :
#

DROP TABLE IF EXISTS `usuario_proyecto`;

CREATE TABLE `usuario_proyecto` (
 `ID_USUARIO` int(11) default NULL,
 `ID_PROYECTO` int(11) default NULL,
 `ROL` varchar(15) NOT NULL,
 KEY `FK_INVOLUCRA` (`ID_PROYECTO`),
 KEY `FK_PARTICIPA` (`ID_USUARIO`),
 CONSTRAINT `FK_INVOLUCRA` FOREIGN KEY (`ID_PROYECTO`) REFERENCES `proyecto`
(`ID_PROYECTO`) ON DELETE SET NULL ON UPDATE CASCADE,
 CONSTRAINT `FK_PARTICIPA` FOREIGN KEY (`ID_USUARIO`) REFERENCES `usuario` (`ID_USUARIO`)
ON DELETE SET NULL ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

#
# Structure for the `vision` table :
#

DROP TABLE IF EXISTS `vision`;
```

```
CREATE TABLE `vision` (
  `ID_VISION` int(11) NOT NULL auto_increment,
  `DESCRIPCION_VISION` text,
  `DECLA_POS_PRO_VISION` text,
  `PRO_VISION` text,
  `AFECTA_VISION` text,
  `IMPACTO_VISION` text,
  `SOLUCION_VISION` text,
  `PARA_VISION` text,
  `QUIEN_VISION` text,
  `NOMBRE_PRO_VISION` text,
  `ESO_VISION` text,
  `DIFERENCIA_VISION` text,
  `NUESTRO_PRO_VISION` text,
  `INTER_NOM_VISION` text,
  `INTER_DES_VISION` text,
  `INTER_RES_VISION` text,
  `USU_AMBIENTE_VISION` text,
  `ID_PAQUETE` int(11) default NULL,
  PRIMARY KEY (`ID_VISION`),
  KEY `ID_PAQUETE` (`ID_PAQUETE`),
  CONSTRAINT `vision_fk` FOREIGN KEY (`ID_PAQUETE`) REFERENCES `paquete` (`ID_PAQUETE`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

Apéndice D – Roles en Open UP y CMMI

D.1 Roles en Open UP:

- **Analista:**

- **Habilidades:**

Un analista necesita los siguientes conocimientos, competencias y habilidades:

- Experticia en identificar, entender problemas y oportunidades.
- Habilidad para articular las necesidades asociadas con los principales problemas a resolver u oportunidad para ser realizados.
- Habilidad para colaborar efectivamente con otros miembros del grupo a través de sesiones de trabajo colaborativo, sesiones JAD y otras técnicas.
- Buenas competencias comunicativas, verbales y de escritura.
- Conocimiento del negocio, dominio de la tecnología o habilidad para absorber y entender rápidamente tal información.

- **Propuesta de asignación:**

Este rol puede ser asignado en las siguientes formas:

- En equipos ágiles pequeños, este rol es frecuentemente compartido entre varios miembros del equipo que también desempeñan otros roles.

- Uno o más miembros del equipo desempeñan este rol exclusivamente. Esta alternativa es comúnmente adoptada cuando los requerimientos son complejos o difíciles de capturar.
- Uno o más miembros del equipo desarrollan este rol y el rol de Tester (pruebas). Esta es una buena opción para grupos de prueba pequeños o con recursos restringidos.

- **Arquitecto:**
  - **Habilidades:**

Los arquitectos deben ser personas polifacéticas, con madurez, visión y una sólida experiencia que les permita abordar temas rápidamente, hacer juicios críticos y académicos con información incompleta. Más específicamente las personas deben poseer esta combinación de estas capacidades:

- **Experiencia en dominios, tanto de problemas como de ingeniería de software**, con evidencia de una completa comprensión de los requisitos para resolver el problema y una participación activa en el desarrollo del software. Si hay un equipo, esta experiencia puede estar representada en diferentes miembros del grupo, pero al menos una persona debe poder mantener la visión global del proyecto.
- **Habilidad de Liderazgo** para motivar y mantener el ímpetu del esfuerzo técnico de los diferentes equipos y tomar decisiones críticas bajo presión, además de

hacer que estas decisiones se mantengan. Para ser efectivo, este rol debe tener la autoridad para tomar decisiones técnicas.

- **Excelentes competencias comunicativas** para inspirar confianza, persuadir, motivar y guiar. Este rol no se puede dirigir por decreto, sino únicamente por el consenso del resto del grupo de proyecto. Para ser efectivo, esta persona debe ganar el respeto de los miembros del grupo, el gerente, el cliente y la comunidad de usuarios, así como del equipo de dirección.
- **Disposición proactiva y orientado a metas** con un enfoque implacable hacia los resultados. Esta persona es la fortaleza en la dirección técnica dentro del proyecto, no un visionario o soñador. La carrera de un arquitecto exitoso es una larga serie de decisiones sub óptimas tomadas en la incertidumbre y bajo presión. Solamente los que puedan enfocarse en hacer lo que necesita hacer tendrán éxito.

Desde un punto de vista de especialización, este rol también necesita mostrar habilidades tanto en el diseño como en la implementación. Sin embargo, desde la perspectiva del diseño, el arquitecto eficaz exhibe típicamente estos rasgos:

- Tiende a generalizar y no a especificar, es quien conoce muchas tecnologías a un alto nivel, en lugar de unas pocas tecnologías a nivel detallado.
- Toma las decisiones técnicas más amplias, en lugar de demostrar profundos conocimientos y experiencia, así como competencias de liderazgo y comunicación.

- **Propuesta de asignación:**

La persona que ocupa este rol, debe estar involucrado en el proyecto desde el inicio hasta el final. Para pequeños proyectos, una sola persona podría actuar tanto como arquitecto como gerente. Sin embargo, si es posible, es mejor que estos roles sean desempeñados por personas diferentes para asegurar que el tiempo invertido en un rol no cause negligencia en el otro rol. Si se adopta esta alternativa de roles separados, ambos individuos deben asegurarse de trabajar de manera muy cercana.

- **Desarrollador:**
    - **Habilidades:**

Un desarrollador debe tener la habilidad necesario para ejecutar las siguientes tareas:

- Definir y crear soluciones técnicas con la tecnología del proyecto.
- Entender y adaptarse a la arquitectura.
- Identificar y construir casos de prueba que cubran el comportamiento requerido de los componentes técnicos.
- Comunicar las decisiones a los miembros del equipo.

Adicionalmente, crea un modelo visual del sistema, este rol necesita la habilidad para representar el diseño en Unified Modeling Language (UML).

- **Propuesta de asignación:**

En equipos ágiles pequeños, este rol es frecuentemente compartido entre varios miembros del equipo que también desempeñan otros roles. Aún en el equipo más pequeño, múltiples individuos deben trabajar juntos para crear la solución técnica. Una persona desempeñando este rol, puede tener competencias específicas en un área técnica en particular, pero también debe tener un amplio entendimiento de todas las tecnologías involucradas en el proyecto, y estar capacitado para trabajar con otros miembros del equipo técnico.

- **Gerente:**
- **Habilidades:**

Una persona que desempeña este rol necesita las siguientes habilidades:

  - Bueno en la presentación, facilitación, comunicación y negociación.
  - Capacidades para conformar equipos y liderarlos.
  - A través de su experiencia en el ciclo de vida del desarrollo de software, enseña, guía y da soporte a otros miembros del equipo.
  - Eficiencia en la resolución de conflictos y la aplicación de técnicas para resolver problemas.
- **Propuesta de asignación:** Este rol es frecuentemente asumido por una sola persona. Este rol es difícil de compartir con otros, pero podría no consumir toda la disponibilidad de una persona.

- **Interesados:**

- **Actividades adicionales que realiza:**
    - Analizar los requerimientos de arquitectura.
    - Evaluar resultados.
    - Crear casos de prueba.
    - Definir Visión.
    - Diseñar la solución.
    - Detallar los requerimientos.
    - Desarrollar la arquitectura.
    - Buscar y analizar requerimientos.
    - Implementar la solución.
    - Manejo de iteración.
    - Planificación de iteración.
    - Plan de proyecto.

- **Otros:**

Este rol permite a cualquiera en un equipo, desempeñar tareas generales como:

- Acceder a artefactos en el sistema de control de configuración para desarrollarlos y mantenerlos
- Someter solicitudes de cambios en el proyecto
- Participar en evaluaciones y revisiones

- **Pruebas:**

Este rol es principalmente responsable por las siguientes tareas:

- Identificar las pruebas que se requiere llevar a cabo.
- Identificar el acercamiento más apropiado para implementar una prueba dada.
- Implementar pruebas individuales.
- Preparar y ejecutar las pruebas.
- Registrar resultados y verificar que las pruebas hayan sido ejecutadas.
- Análisis y recuperación de errores de ejecución.
- Comunicar los resultados de las pruebas al equipo.

- **Habilidades:**

Una persona para este rol debe tener las siguientes habilidades:

- Conocimiento de tendencias de pruebas y técnicas.
- Habilidades para el diagnóstico y solución de problemas.
- Conocimiento del sistema o aplicación que está siendo probada (deseable).
- Conocimiento de redes y arquitectura de sistemas (deseable).

Donde se requiere automatizar pruebas, se requiere estas cualidades adicionales:

- Entrenamiento en el uso apropiado de herramientas de automatización de pruebas.

- Experiencia usando herramientas de automatización de pruebas.
- Habilidades de programación.
- Habilidades en depuración y diagnóstico.

Nota: El requerimiento de habilidades específicas, varía dependiendo del tipo de pruebas que se está dirigiendo. Por ejemplo, las habilidades necesarias para tener éxito, en el uso de sistemas cargados con herramientas de automatización de pruebas, son diferentes de aquellas necesarias para la automatización de pruebas de sistemas funcionales.

- **Propuesta de asignación:**

Este rol puede ser asignado en las siguientes formas:

- Asignar uno o más miembros del personal de pruebas para desempeñar este rol. Esto es una aproximación bastante frecuente y es particularmente útil en equipos pequeños, como también para grupos de cualquier tamaño, donde el equipo está conformado por un grupo experimentado de probadores o de un nivel de habilidades relativamente equivalente.
- Asignar uno o más miembros del personal de pruebas para desempeñar únicamente este rol. Esto funciona bien en grupos grandes. Es también útil para separar responsabilidades cuando alguno del personal de pruebas tiene más experiencia en la automatización de pruebas que otros miembros del grupo.
- Asignar uno o más miembros del grupo, que están ya desempeñando otro rol en el proyecto, para ser responsable por la prueba de alguna parte de las capacidades del

sistema. El miembro del equipo tendrá que tener las habilidades apropiadas para la prueba.

D.2 Integración del modelo de madurez de capacidades (CMMI):

- **Nivel de Capacidad 0.** Incompleto: Un proceso "incompleto" es un proceso que no está bien realizado o se encuentra realizado parcialmente.
- **Nivel de Capacidad 1.** Realizado: El nivel 1 se caracteriza por tener "procesos realizados". Esto implica que el proceso satisface los objetivos.
- **Nivel de Capacidad 2.** Gestionado: Este nivel se caracteriza por tener "procesos gestionados". Además de cumplir con el nivel 1, posee una "infraestructura de apoyo" para el proceso. Se debe planificar y ejecutar los procesos según sus políticas, se debe involucrar a las partes interesadas, el proceso debe ser controlado, monitoreado y revisado.
- **Nivel de Capacidad 3.** Definido: Este nivel consiste en tener "procesos definidos". Un proceso definido es gestionado (nivel de capacidad 2), son procesos que se adaptan a las directrices de la organización y contribuyen con los productos de trabajo, las medidas y con otras mejoras.
- **Nivel de Capacidad 4.** Gestionado cuantitativamente: Se caracteriza por tener procesos "gestionados cuantitativamente". Estos procesos son definidos (capacidad 3) y además son controlados mediante estadísticas y otras técnicas cuantitativas.

- **Nivel de Capacidad 5.** Optimización: Se caracteriza por "procesos de optimización". Estos procesos son cuantitativamente gestionados (capacidad 4), y se basan en la compresión de las causas de variaciones en los procesos. Posee un enfoque de mejora continua, mejoras incrementales e innovadoras.
- **Nivel de Madurez 1.** Inicial: Los procesos tienden a ser caóticos, la organización no suele proporcionar un entorno estable para apoyar los procesos.
- **Nivel de Madurez 2.** Gestionado: En este nivel los procesos se planifican y se ejecutan bajo una política, son controlados, supervisados, revisados y evalúan el cumplimiento en la descripción de sus procesos. El nivel 2 contribuye a garantizar que las buenas prácticas adquiridas en la organización, se mantengan en momentos de crisis.
- **Nivel de Madurez 3.** Definido: Los procesos están bien caracterizados y comprendidos; se describen normas, procedimientos, herramientas y métodos. Estos procesos estándares se utilizan para establecer la coherencia en toda la organización. Se establecen proyectos definidos por los procesos.
- **Nivel de Madurez 4.** Gestionado cuantitativamente: La organización y los proyectos tienen objetivos de calidad, rendimiento y gestión de procesos.
- **Nivel de Madurez 5.** Optimización: La organización mejora continuamente sus procesos, con innovación a través de mejoras tecnológicas. Existe revisión continua de objetivos y de la gestión de procesos.

Matriz de pruebas
Pruebas a Requerimientos Funcionales

Cliente	
Codigo Proyecto:	ROUP
Nombre Proyecto:	Herramienta web de código abierto para la gestión de requerimientos durante el ciclo de vida de desarrollo del software para metodología Open UP
Release:	Release 1
Fecha de Revisión:	Septiembre2008 - Febrero2009
Duración:	6 meses
Nombre Revisor:	Ana Fernandes / Carlos Miranda

FB **Flujo Básico**

Iteración	Tipo de prueba	Caso de Prueba	Escenario de Prueba	Descripción del Escenario	Flujos	Datos de Entrada	Resultados Esperados	Resultados Obtenidos	Estatus		Criticidad		Observaciones
1	V & V	Hist Uso.	ES01	Validar y Verificar que las historias de usuario cumplen con los requerimientos del proyecto.	FB	Requerimientos del proyecto	Validar que se va a construir el producto correcto y Verificar que se va a construir en forma correcta		Aprobado	0	Alta	0	
1	Funcional	OOW	ES02	Validar y Verificar la compatibilidad ODF-PHP	FB	Creación de un documento de prueba desde PHP - ODF - OOW	Documento generado con los datos obtenidos desde página web.	Documento en OpenOffice Writer	Aprobado	0	Alta	0	Se debe crear una plantilla del documentos a generar en OpenOffice Writer, para que éste sea utilizado por la librería ODF
2	Funcional	BD - CP	ES03	Se desea realizar operaciones CRUD	FB	CRUD de objetos (caracteristicas, requerimientos, casos de uso, terminos, paquetes, proyectos, comentarios, usuarios)	Que se pueda realizar operaciones CRUD en los elementos del proyecto.	Errores en la creacion de casos de Uso, modificación de requerimientos, no elimna casos de uso.	Estatus	FALSO	Alta	0	No se almacenen los flujos de casos de uso, presentan error las restricciones de la base de datos. En requerimientos al modificar existe error de asociación Requerimientos - caracteristicas. No se puede eliminar flujos del caso de uso por restriccion en la base de datos.
2	Funcional	BD - CP	ES04	Operaciones CRUD	FB	CRUD de objetos (caracteristicas, requerimientos, casos de uso, términos, paquetes, proyectos, comentarios, usuarios)	Revisión de correcciones del escenario de pruebas ES03	Creación exitosa de Caracteristicas, requerimientos, casos de uso. Error en la asociación de paquetes, no se crean comentarios	Estatus	FALSO	Alta	0	Error en la construcción de la clase clase_termino; el error se presenta al crear el comentario error de sintaxis. Los paquetes error en la clase clase_paquete, error de asociación con el proyecto.
2	Funcional	BD - CP	ES05	Operaciones CRUD	FB	CRUD de objetos (caracteristicas, requerimientos, casos de uso, terminos, paquetes, proyectos, comentarios,	Revisión de correcciones del escenario de pruebas ES04	Éxito en los casos de crear, consulta. Error al modificar requerimientos y eliminar caracteristicas	Estatus	FALSO	Alta	0	Errores por restricción en la base de datos
2	Funcional	BD - CP	ES06	Operaciones CRUD	FB	CRUD de objetos (caracteristicas, requerimientos, casos de uso, términos, paquetes, proyectos, comentarios,	Revisión de correcciones del escenario de pruebas ES05	Éxito operaciones CRUD para todos los elementos	Aprobado	0	Alta	0	CRUD exitoso para todos los elementos
3	Funcional	AJAX - JS	ES07	Creación del árbol	FB	Los elementos creados en el proyecto	Mostrar el árbol con todos los elementos del proyecto	En ocaciones no es visible el árbol en el navegador FIREFOX.	Estatus	FALSO	Alta	0	La página consultarProyecto refresca el árbol pero en ese momento aún no se ha ejecutado el código que genera el árbol, causa que generaba que elframe quedara en blanco.
3	Funcional	AJAX - JS	ES08	Creación del árbol	FB	Los elementos creados en el proyecto	Revisión correcciones del escenario de pruebas ES07	Si aparece el árbol en los navegadores	Aprobado	0	Alta	0	
3	Funcional	AJAX - JS	Es09	Creación del objeto XMLHttpRequest	FB		Poder utilizar este objeto para la comunicación asincrona con el servidor	Error en la compatibilidad Cross-Browser	Estatus	FALSO	Alta	0	No funciona debido a que IE y FIREFOX manejan la clase XMLHttpRequest diferente.
3	Funcional	AJAX - JS	ES10	Creación del objeto XMLHttpRequest	FB		Poder utilizar este objeto para la comunicación asincrona con el servidor	Creación éxitosa del objeto XMLHttpRequest para ambos manejadores.	Aprobado		Alta		

4	Funcional	CN	ES11	Servicios de la herramienta	FB	Correcto funcionamiento de servicios	Funcionamiento exitoso		Aprobadu		Alta		
4	Funcional	CN	ES12	Interfaces	FB	Proyecto UCAB (todos los niveles de requerimientos)	Compatibilidad Cross-Browser	Diferente interpretación de código según el navegador	Estatus	FALSO	Alta	0	Desarrollar las interfaces de modo que se pueda observar la compatibilidad Cross- Browser
4	Funcional	CN	ES13	Interfaces	FB	Proyecto UCAB (todos los niveles de requerimientos)	Revisión correcciones del escenario de pruebas ES12	Compatibilidad Cross - Browser	Aprobado	0	Alta	0	
4	Funcional	CN	ES14	Interfaces	FB	Proyecto UCAB (todos los niveles de requerimientos)	Recarga del árbol ante cada acción del usuario	Se recarga el árbol antes que se actualice los datos en la base de datso	Estatus	FALSO	Alta	0	Ejecución de JAVASCRIPT antes que el código PHP. Aplicar un delay en las funciones de JAVASCRIPT
4	Funcional	CN	ES15	Interfaces	FB	Proyecto UCAB (todos los niveles de requerimientos)	Recarga del árbol ante cada acción del usuario	Recarga exitosa		FALSO	Alta	0	
4	Funcional	CN	ES16	Interfaces	FB	Proyecto UCAB (todos los niveles de requerimientos)	Codificación de caracteres especiales	Los navegadores no mostraban adecuadamente los caracteres especiales sustituyendolos por signos de interrogación (¿?) y otros.	Estatus	FALSO	Alta	0	Codificación errónea, las paginas estan codificadas en ISO-8859- 1 y los datos durante la utilización de AJAX se codifican en UTF-8
4	Funcional	CN	ES17	Interfaces	FB	Proyecto UCAB (todos los niveles de requerimientos)	Codificación de caracteres especiales	Se muestran adecuadamente todos los caracteres especiales	Aprobado	0	Alta	0	
4	Funcional	CN	ES18	Interfaces	FB	Proyecto UCAB (todos los niveles de requerimientos)	Correcta modificación y eliminación de las tablas generales de consulta de los distintos niveles de requerimientos	No modifica y elimina pero no refresca la fila de la tabla	Estatus	FALSO	Alta	0	Error en la capa de negocio al construir los objetos (caracteristicas y requerimientos). No realiza la modificación porque el objeto no esta creado
4	Funcional	CN	ES19	Interfaces	FB	Proyecto UCAB (todos los niveles de requerimientos)	Corrección del escenario de pruebas ES18	Éxito en todos los casos	Aprobado	0	Alta	0	
4	Funcional	CN	ES20	Interfaces	FB	Proyecto UCAB (todos los niveles de requerimientos)	Listado de elementos de cada paquete	Error en los casos de paquetes que poseen requerimientos y caracteristicas	Estatus	FALSO	Alta	0	Los formularios poseen el mismo nombre, y al ejecutar alguna acción no funciona adecuadamente
4	Funcional	CN	ES21	Interfaces	FB	Proyecto UCAB (todos los niveles de requerimientos)	Corrección del escenario de pruebas ES20	Éxito se muestra el contenido en todos los paquetes del proyecto	Aprobado	0	Alta	0	
4	Funcional	CN	ES22	Interfaces	FB	Proyecto UCAB (todos los niveles de requerimientos)	Modificar los términos en consulta personalizada	Error al no seleccionar el paquete de ubicación	Estatus	FALSO	Alta	0	Validar que el paquete por defecto de un término es Glosario
4	Funcional	CN	ES23	Interfaces	FB	Proyecto UCAB (todos los niveles de requerimientos)	Modificar las características en consulta personalizada	Error al no seleccionar el paquete de ubicación	Estatus	FALSO	Alta	0	Validar que el paquete por defecto de una característica es Visión y Caracteristicas
4	Funcional	CN	ES24	Interfaces	FB	Proyecto UCAB (todos los niveles de requerimientos)	Modificar requerimientos en consulta personalizada	Error al no seleccionar el paquete de ubicación	Estatus	FALSO	Alta	0	Validar que el paquete por defecto de un requerimiento es Requerimientos del sistema
4	Funcional	CN	ES25	Interfaces	FB	Proyecto UCAB (todos los niveles de requerimientos)	Modificar los casos de uso en consulta personalizada	Error al no seleccionar el paquete de ubicación y listar los flujos del caso de uso	Estatus	FALSO	Alta	0	Validar que el paquete por defecto de un caso de uso es Casos de Uso
4	Funcional	CN	ES26	Interfaces	FB	Proyecto UCAB (todos los niveles de requerimientos)	Corrección escenario de prueba ES22	Éxitoso, se modifican los terminos correctamente	Aprobado	0	Alta	0	
4	Funcional	CN	ES27	Interfaces	FB	Proyecto UCAB (todos los niveles de requerimientos)	Corrección escenario de prueba ES23	Éxitoso, se modifican las caracteristicas	Aprobado	0	Alta	0	
4	Funcional	CN	ES28	Interfaces	FB	Proyecto UCAB (todos los niveles de requerimientos)	Corrección escenario de prueba ES24	Éxitoso, se modifican los requerimientos correctamente	Aprobado	0	Alta	0	
4	Funcional	CN	ES29	Interfaces	FB	Proyecto UCAB (todos los niveles de requerimientos)	Corrección escenario de prueba ES25	Éxitoso se modifican los casos de uso correctamente	Aprobado	0	Alta	0	
5	Funcional	WTGCR	ES30	Comentarios	FB	Proyecto UCAB	Realizar comentarios para los distintos niveles de requerimientos	No se muestra todos los comentarios para una caracteristica, requerimiento o caso de uso	Estatus	FALSO	Alta	0	Invocación errónea de las funciones de ajax

5	Funcional	WTGCR	ES31	Comentarios	FB	Proyecto UCAB	Corrección escenario de prueba ES30	Éxitoso se listan todos los comentarios realizados sobre cualquier nivel de requerimiento, en todos	Aprobado	0	Alta	0	
5	Funcional	WTGCR	ES32	Trazabilidad	FB	Proyecto UCAB	Mostrar la trazabilidad entre Caracteristicas - Requerimientos	Error no se muestra la matriz de trazabilidad	Estatus	FALSO	Alta	0	Error en el ciclowhile de la clase PrazabilidadRequerimiento. Error de la condiciones del ciclo.
5	Funcional	WTGCR	ES33	Trazabilidad	FB	Proyecto UCAB	Corrección escenario de prueba ES32	Éxitoso se muestra la trazabilidad entre Caracteristicas - Requerimientos	Estatus	FALSO	Alta	0	
5	Funcional	WTGCR	ES34	Cambios Trazabilidad	FB	Proyecto UCAB	Mostrar cambios en la matriz de trazabilidad	Exitoso para el caso cambio requerimiento y cambio caracteristica, pero no se muestra cuando ocurre cambio en ambos	Estatus	FALSO	Alta	0	Uso del ID de las caracteristica para el caso de cambio en ambos niveles de requerimientos
5	Funcional	WTGCR	ES35	Cambios Trazabilidad	FB	Proyecto UCAB	Corrección escenario de prueba ES34	Éxitoso se muestran los cambios correctamente	Aprobado	0	Alta	0	
5	Funcional	WTGCR	ES36	Cambios Trazabilidad	FB	Proyecto UCAB	Permitir aprobación de cambios	Éxitoso se aprueban los cambios	Aprobado	0	Alta	0	
5	Funcional	WTGCR	ES37	GCR5	FB	Proyecto UCAB	Integración de las herramientas	Integración correcta, se muestran los cambio generados en los niveles de	Aprobado	0	Alta	0	
6	Funcional	OOW2	ES38	Documentos	FB	Proyecto UCAB	Generación de documentos	No se listan las caracteristicas en el documento de Visión, ni los requerimientos en el documento de Requerimientos del	Estatus	FALSO	Alta	0	Error en los delimitadores de la plantilla de documento de OpenOffice Writer que necesita la libreria OOF
6	Funcional	OOW2	ES39	Documentos	FB	Proyecto UCAB	Corrección escenario de prueba ES38	Éxitoso se generan los documentos	Aprobado	0	Alta	0	
6	Funcional	OOW2	ES40	Documentos	FB	Proyecto UCAB	Generar documento con las listas de caracteristicas, requerimientos y	Éxitoso se generan los documentos	Aprobado	0	Alta	0	

Caso de estudio: Proyecto Imatours.

Descripción: Este proyecto está basado en la creación de un sistema para la gestión administrativa de la agencia de viajes IMATOURS C.A. El sistema debe ser web y debe permitir toda la gestión administrativa de la agencia de viajes, además se debe realizar un módulo extra, es decir, una página web que permita a los usuarios registrarse, realizar compras y reservaciones a través de internet.

Nombre	Origen	Prioridad	Estatus	Dificultad	Estabilidad	Riesgo	P. contacto	Tipo	Descripción
Comparación de precios	Cliente	Alta	Aprobado	Alta	Media	Calendario (Alto)	Lic. Julián Colmenares	Funcional	El usuario será capaz de comparar los precios de vuelos procedentes de otros aeropuertos, cercanos a la región (por vuelos entrantes y salientes)
For.de fechas según el browser	Cliente	Media	Aprobado	Alta	Alta	Calendario (Medio)	Adriana Pérez	Restricción de Diseño	Fechas se muestran de acuerdo con el formato del navegador web.

Cancelar la compra de boletos	Cliente	Alta	Incorporado	Media	Media	Calendario (Medio)	Lic. Bernardo Serpa	Funcional	Se podrá cancelar una compra de boleto de vuelo en cualquier momento antes de la confirmación definitiva de la compra.
Cancelar hotel o vehículo	Cliente	Alta	Aprobado	Alta	Alta	Calendario (Medio)	Lic. Julián Colmenares	Funcional	El usuario podrá cancelar una reservación de hotel o vehículos
Compra de boletos	Cliente	Alta	Aprobado	Alta	Alta	Calendario (Alto)	Lic. Julián Colmenares	Funcional	El usuario podrá comprar boletos y seleccionar su puesto en el avión
Usuarios	Cliente	Alta	Aprobado	Alta	Alta	Calendario (Alto)	Sra. Rosa Hernández	Funcional	Gestionar de usuarios de a través de la web

Requerimientos:

Nombre	*Car. Origen*	Prioridad	*Estatus*	*Dificultad*	*Estabilidad*	*Riesgo*	*P. contacto*	*Tipo*
Precios de vuelos	Comparación de precios	Alta	Aprobado	Alta	Media	Calendario (Alto)	Lic. Julián Colmenares	Funcional
Formato de fechas	For.de fechas según el browser	Media	Aprobado	Alta	Alta	Calendario (Medio)	Adriana Pérez	No Funcional
Cancelar compra	Cancelar la compra de boletos	Alta	Validado	Alta	Alta	Calendario (Medio)	Lic. Bernardo Serpa	Funcional
Cancelar reserva	Cancelar hotel o vehículo	Media	Aprobado	Alta	Alta	Calendario (Medio)	Lic. Bernardo Serpa	Funcional
Orden de vuelos	Comparación de precios	Alta	Aprobado	Alta	Alta	Calendario (Bajo)	Lic. Julián Colmenares	Funcional
Boletos	Compra de boletos	Media	Incorporado	Alta	Alta	Calendario (Medio)	Lic. Julián Colmenares	Funcional
Visualización de calendario	For.de fechas según el browser	Baja	Aprobado	Alta	Alta	Tecnología (Bajo)	Lic. Julián Colmenares	Funcional
Formato de vuelos	Comparación de precios	Media	Aprobado	Alta	Alta	Calendario (Alto)	Lic. Bernardo Serpa	Funcional
Crear usuarios	Usuarios	Alta	Validado	Alta	Alta	Calendario (Alto)	Lic. Julián Colmenares	Funcional

Casos de Uso:

Nombre	*Requerimiento Origen*	*Prioridad*	*Estatus*	*Dificultad*	*Riesgo*
Consultar Precios	Precios de vuelos	Alta	Aprobado	Media	Calendario (Alto)
Reservar vuelo	Boletos	Alta	Aprobado	Alta	Calendario (Alto)
Registrar Usuario	Crear usuarios	Alta	Validado		Calendario (Medio)

Especificación de Casos de Uso:

Caso de Uso Consultar Precios:

Nombre: Consultar Precios

Descripción: Los usuarios al realizar una reservación pueden listar los precios de su vuelo con diferentes aeropuertos cercanos (salida y llegada)

Actores:

- Viajero.
- Sistema.
- Administrador

Precondición: Estar registrado en la página web

Flujo Básico de Eventos
Paso1:El viajero selecciona la opción del menú " Compare en aeropuertos cercanos"
Paso2:El sistema debe mostrar una lista con los aeropuertos que se encuentren a menos de 100 millas
Paso3:El viajero selecciona los aeropuertos a ser considerados
Sub Paso3.1:El viajero puede regresar el menú anterior
Paso4:El sistema realiza la búsqueda de precios, horarios, vuelos con las opciones del viajero
Paso5:El viajero selecciona su mejor opción
Paso6:Sigue caso de uso RESERVAR VUELO

Post Condición: El usuario queda en la página

Caso de Uso Reservar Vuelo:

Nombre: Reservar vuelo

Descripción: Reserva de vuelos por parte de los usuarios puede elegir el aeropuerto de llegada, aeropuerto de salida, fecha, hora

Actores:

- Viajeros.

- Personal de la Agencia.
- Sistema.

Precondición: El administrador debe iniciar sesión en el sistema, los viajeros deben acceder a la página web.

Flujo Básico de Eventos
Paso1:Viajero entra en la URL de la agencia
Paso2: Sistema muestra la página de inicio.
Paso3:El viajero debe seleccionar una opción en el menú indicando aeropuertos de salida y llegada, horario, fecha, nº de pasajeros
Sub Paso3.1:Caso de uso CONSULTAR PRECIOS
Paso4: Viajero selecciona "Buscar vuelos".
Paso5: El sistema de muestra los vuelos de salida ordenados por precio.
Sub Paso5.1: El viajero cambia la clasificación de los vuelos.
Sub Paso5.2: El sistema presenta los vuelos ordenados por un determinado criterio.
Paso6: El Viajero selecciona un vuelo.
Paso7: El sistema muestra vuelos de ida y vuelta.
Paso8: El viajero selecciona un vuelo de regreso.
Paso9: El sistema muestra los detalles del vuelo.
Paso10:El viajero confirma el vuelo

Paso11: El viajero debe indicar el nombre de de usuario y contraseña para continuar con la compra del boleto.

Sub Paso11.1:El viajero selecciona "Nuevo Usuario"

Sub Paso11.2:El viajero debe indicar nombre de usuario, nombre, apellido, contraseña, e-mail, dirección, teléfonos

Sub Paso11.3:El sistema verifica que la dirección de correo es única y debe tomarse como ID de usuario

Paso12: El sistema muestra los asientos disponibles.

Paso13: El viajero selecciona los asientos.

Paso14: El viajero proporciona información de la tarjeta de crédito y la dirección de facturación.

Paso15: El sistema proporciona un número de confirmación.

Post Condición: Registro del vuelo

Caso de Uso Crear Usuario:

Nombre: Crear Usuario

Descripción: Registrar usuario

Actores:

- Viajeros

- Administrador
- Sistema

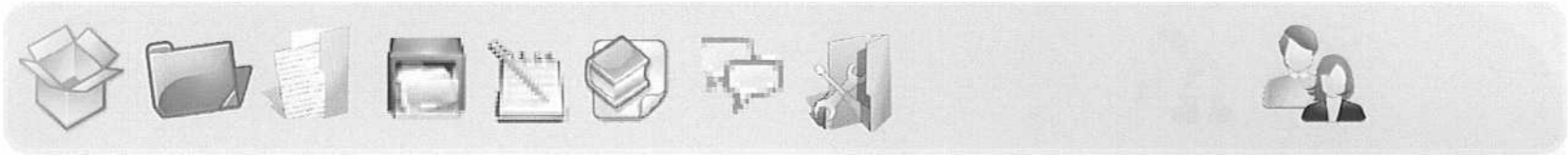
Precondición: Ingresar a la página web

Flujo Básico de Eventos
Paso1:El usuario debe acceder a la página web
Paso2:El sistema muestra un menú de opciones
Paso3:El usuario selecciona del meno REGISTRARSE
Sub Paso3.1:El sistema muestra formulario indicado
Sub Paso3.2:El usuario introduce los datos

Post Condición: El usuario puede realizar reservas y compras

F.2 Diseño general de las interfaces:

Barra de Herramientas



Información del Proyecto

Crear Paquete

Crear Característica

Crear Requerimiento

Crear Caso de Uso

Crear Término

Crear Comentario

Administrar Cuenta

Cerrar Sesión

Herramienta de Gestión de Cambio

Acceso al Sistema

Proyecto:

Login:

Clave:

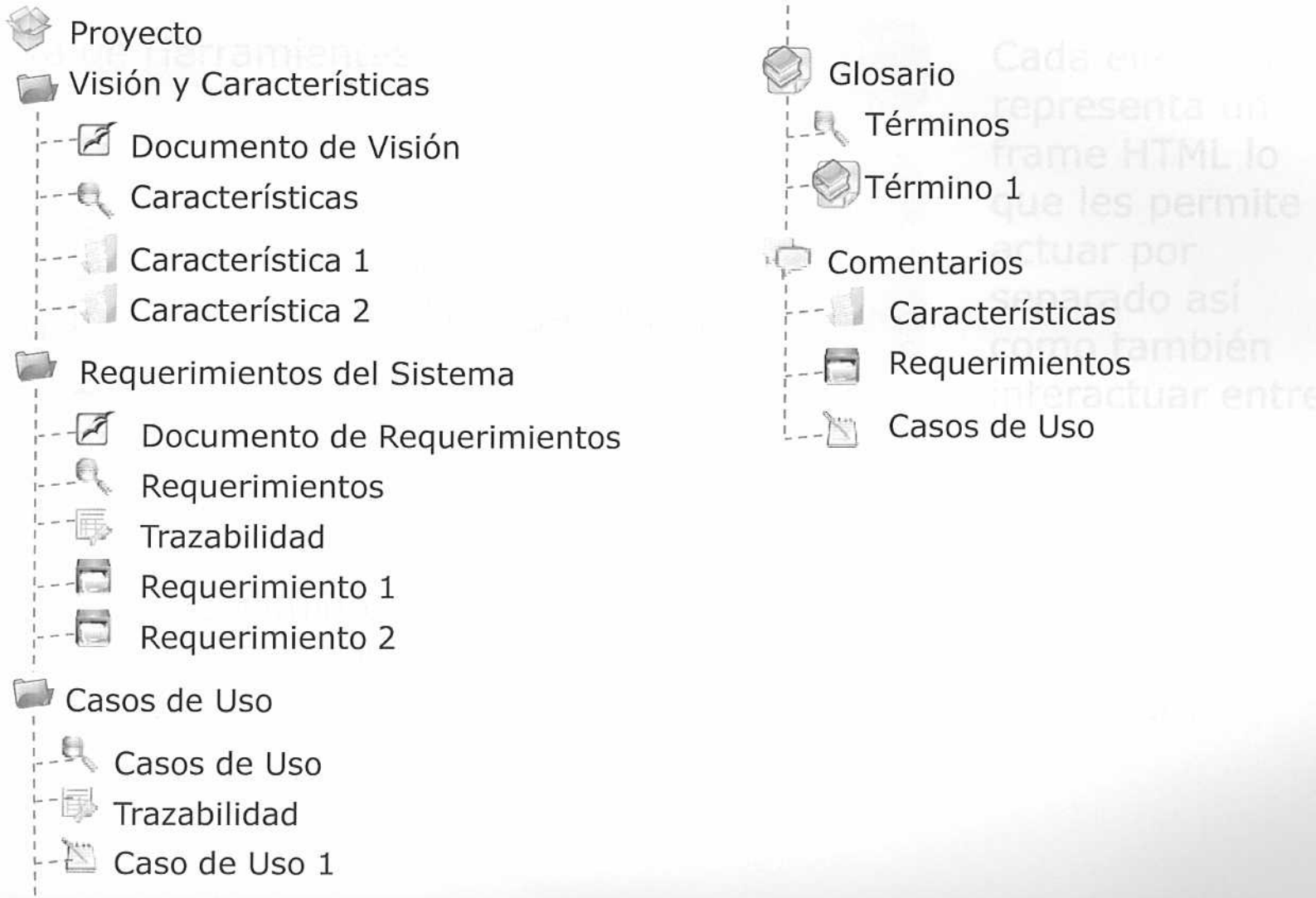
Entrar

Olvido su Clave?

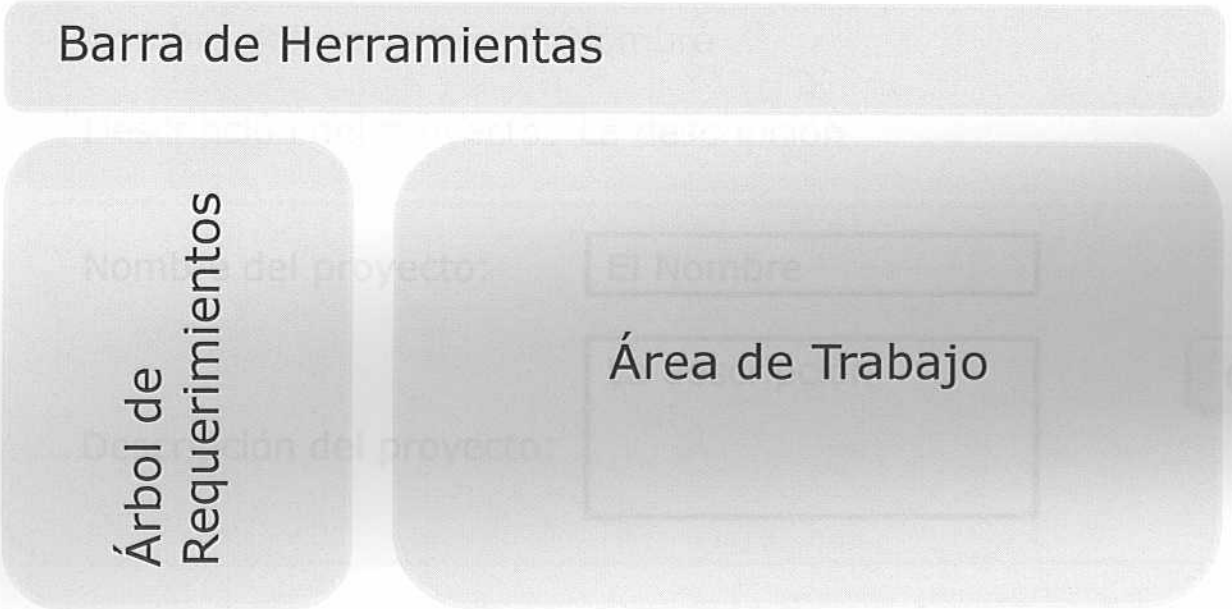
Crear Cuenta

Descargar la Herramienta

Árbol de Requerimientos



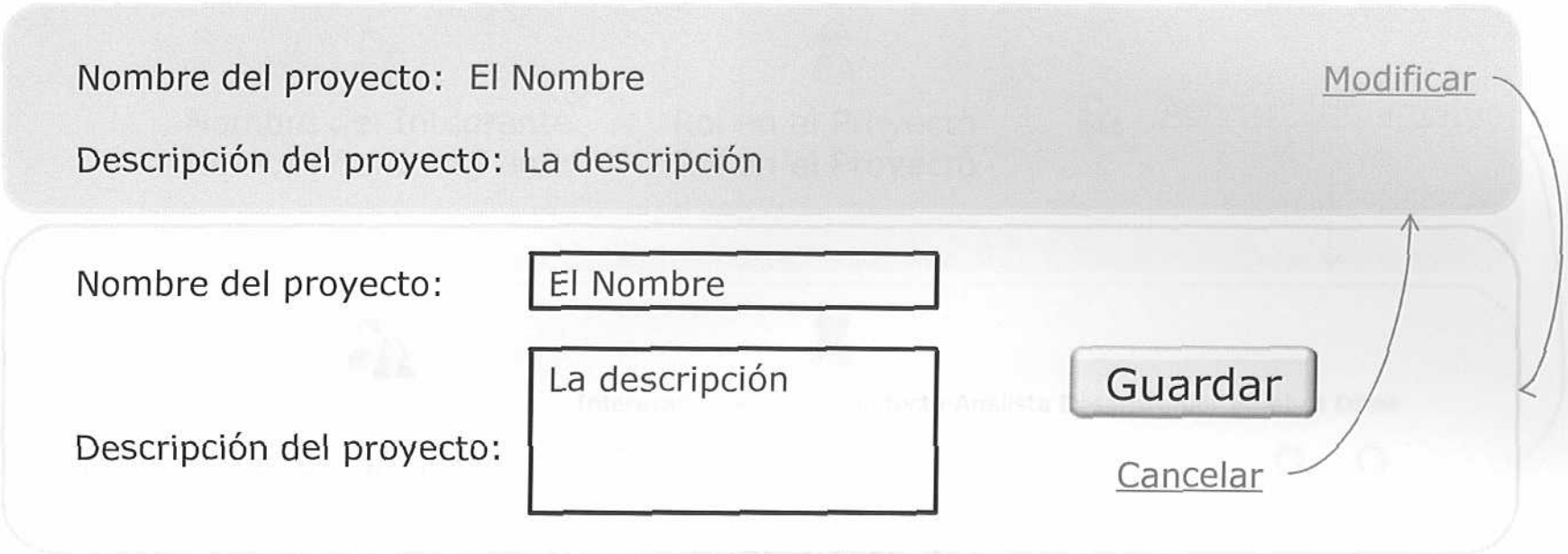
Pantalla Principal



Cada elemento representa un frame HTML lo que les permite actuar por separado así como también interactuar entre si.

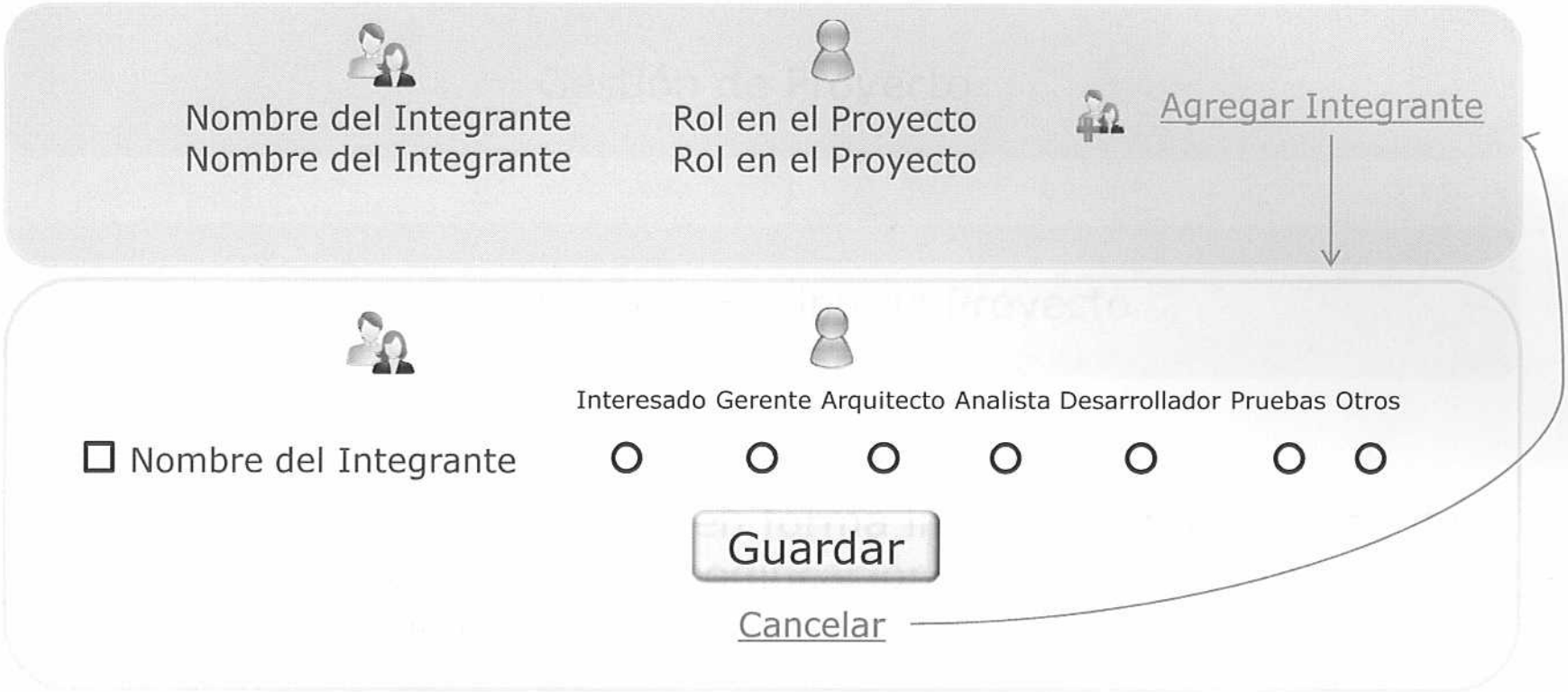
La Barra de Herramientas se mantiene fija, el Árbol de Requerimientos se refresca cuando alguna tarea así lo demanda y el Área de Trabajo varia según la actividad que se este realizando.

Gestión del Proyecto



El elemento azul es visible para el usuario mientras que el blanco se encuentra invisible, si se presiona el link de modificar ambos elementos cambian su visibilidad y el usuario solo ve el formulario, si se presiona cancelar vuelve a su estado inicial.

Gestión del Equipo de Proyecto



Los links de agregar integrante y cancelar cambian la visibilidad de los elementos.

Consulta de Proyecto

Gestión de Proyecto

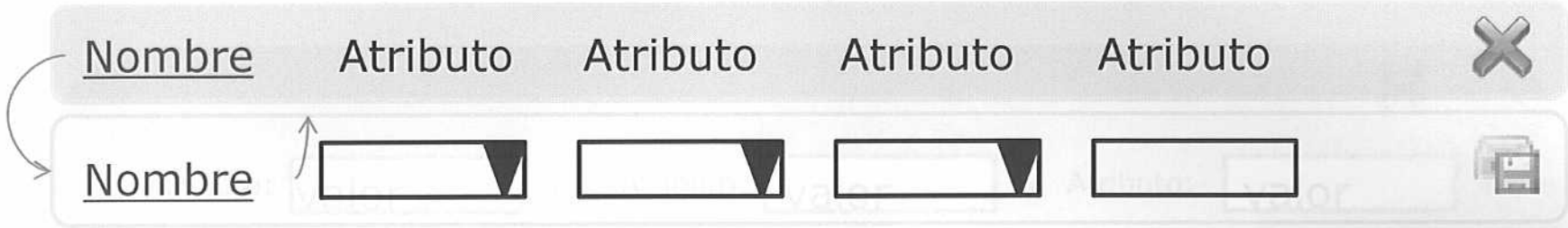
Gestión del Equipo de Proyecto

Ambos elementos se recargan en forma independiente en caso de que el usuario realice alguna modificación. Esta interfaz se carga en el Área de Trabajo

Crear Elementos de los diferentes niveles de Requerimientos

Los elementos con * son obligatorios. La distribución de los campos varia para los diferentes niveles. Al presionar en paquete por defecto este cambia por una casilla de selección que contiene los paquetes en los que se puede almacenar. Esta interfaz se carga en el Área de Trabajo

Consultar Elementos de los diferentes niveles de Requerimientos



Cada requerimiento genera un par de elementos, al hacer click sobre el nombre cambia la visibilidad, esta interfaz se carga en el Área de trabajo y permite a los usuarios modificar la información principal de los requerimientos así como eliminarlos. Cada par de elementos funciona por separado.

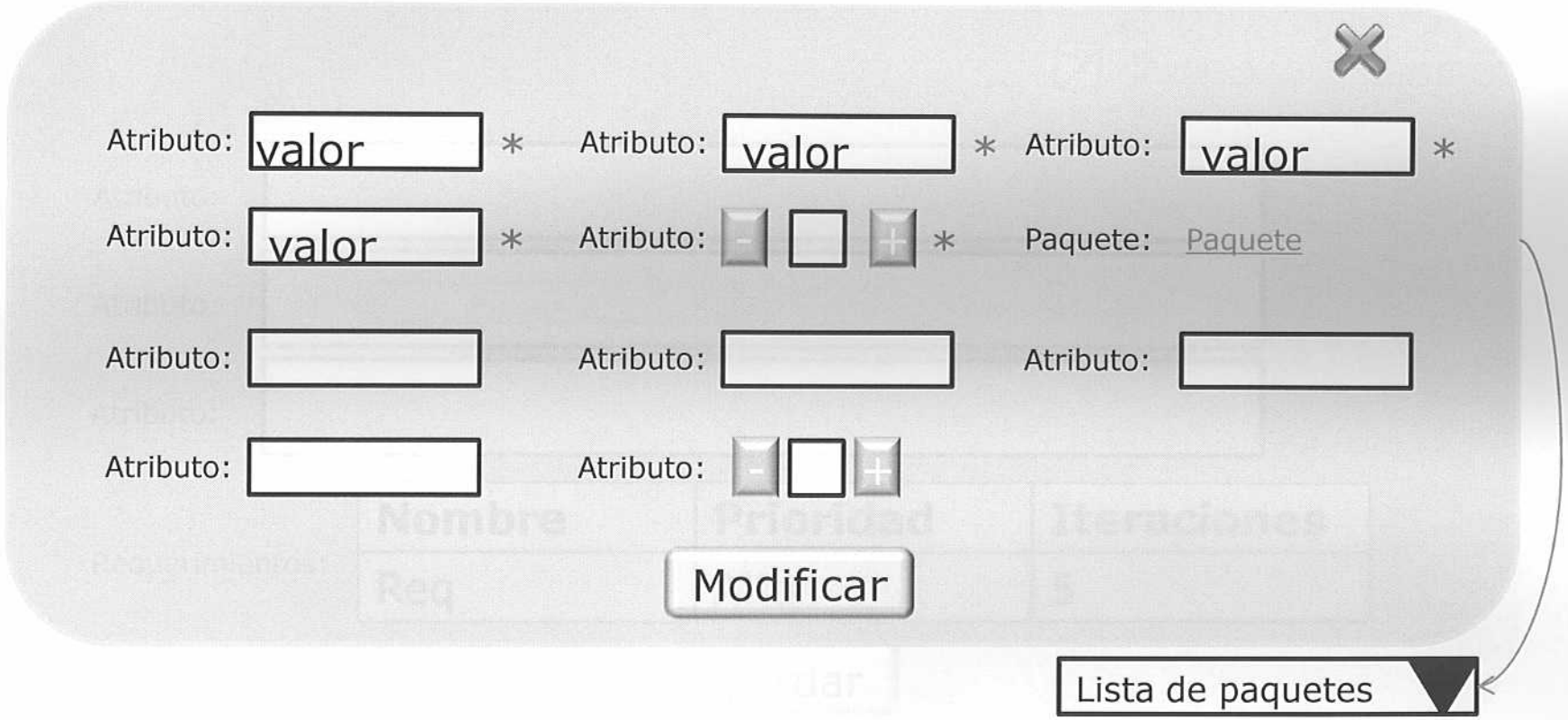
Enviar a OpenOffice

Resultado

Resultado

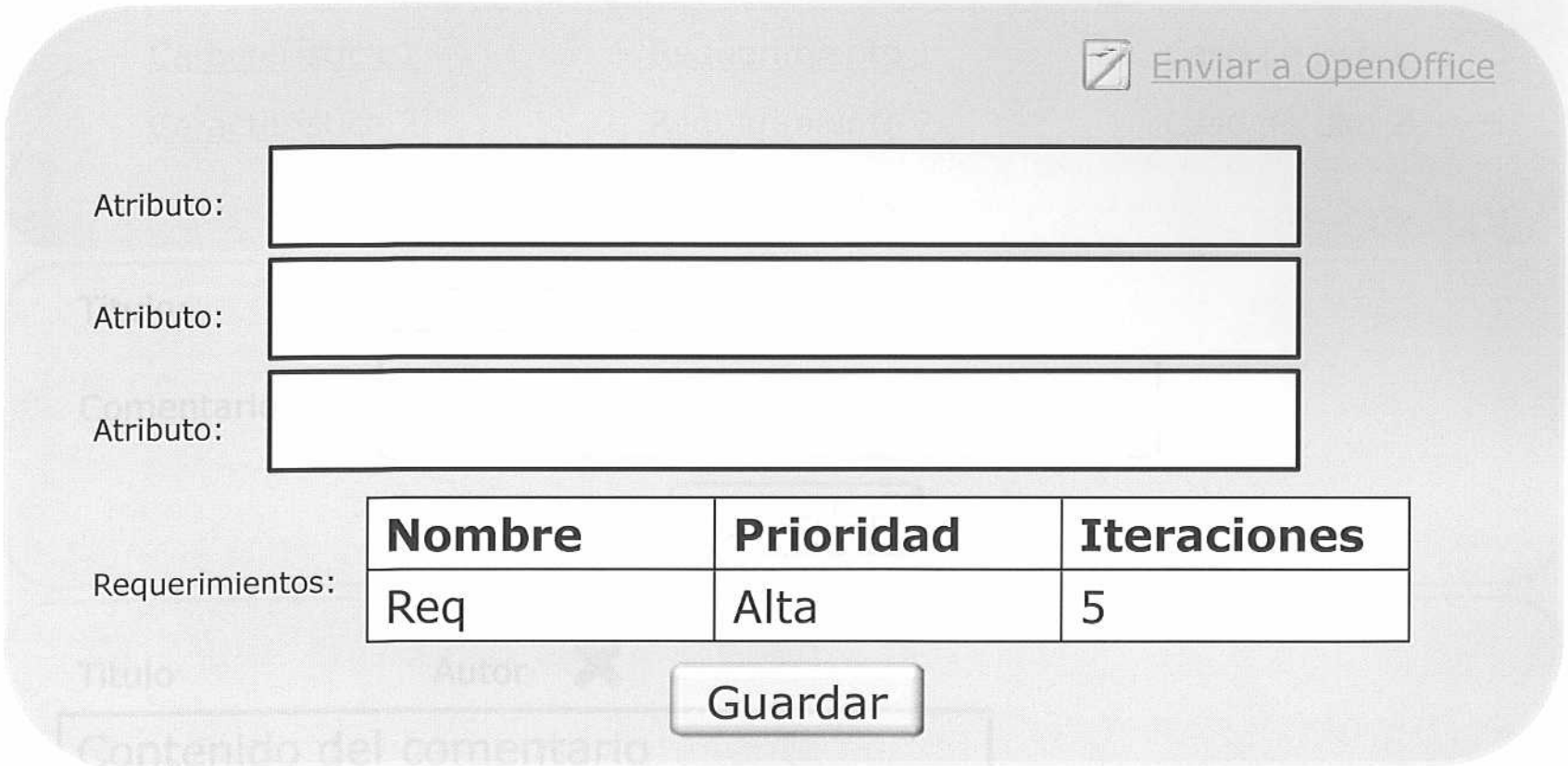
Resultado

Modificar Elementos de los diferentes niveles de Requerimientos



Esta interfaz permite a los usuarios cambiar cualquier valor de los requerimientos así como eliminarlos. Se carga en el Área de Trabajo.

Documentos



Los atributos varían según el nivel de Requerimiento. Esta interfaz se carga en el Área de Trabajo.

Comentarios

Características

Requerimientos

Casos de Uso

Característica 1

Característica 2

Requerimiento 1

Requerimiento 2

Requerimiento 3

Caso de Uso 1

Caso de Uso 2

Título:

Comentario:

Guardar

Titulo Autor

Contenido del comentario

Trazabilidad

Nivel de requerimiento 1 – Nivel de Requerimiento 2	N1	N1	N1	N1	N1
N2					
N2					
N2					
N2					

 Existe relación

 Requerimiento padre presenta cambio

 Requerimiento hijo presenta cambio

 Ambos Requerimientos presentan cambio

Gestión de Cambio

Proyecto

Características

Nombre 

Estatus Usuario Tipo Fecha

Requerimientos

Nombre 

Estatus Usuario Tipo Fecha

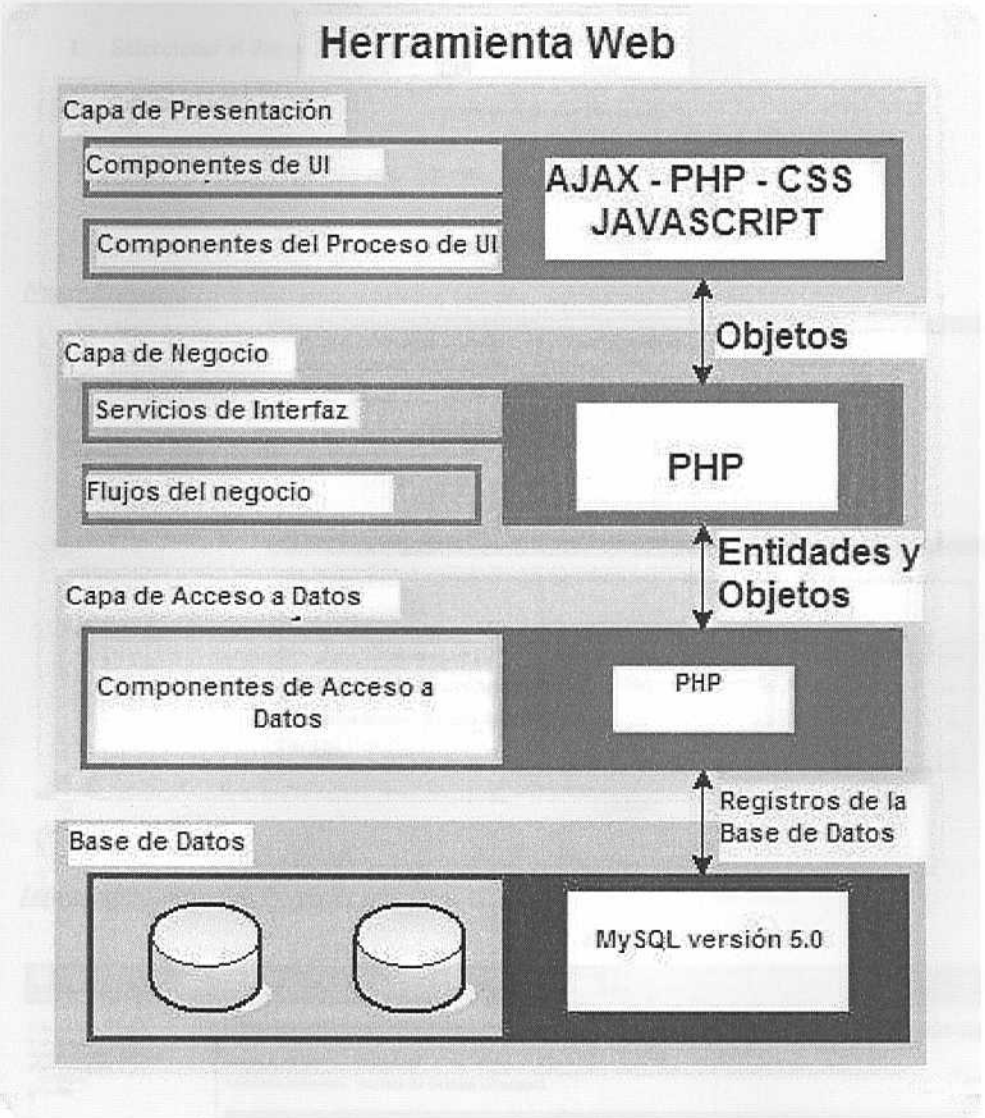
Casos de Uso

Nombre 

Estatus Usuario Tipo Fecha

 Contrae o expande la lista del cambio según sea el caso.

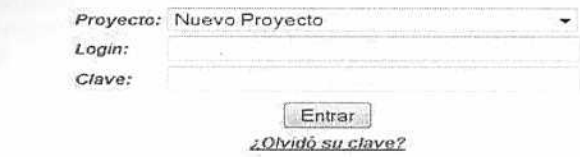
F.4 Diagrama de Arquitectura:



Manual de Usuario:

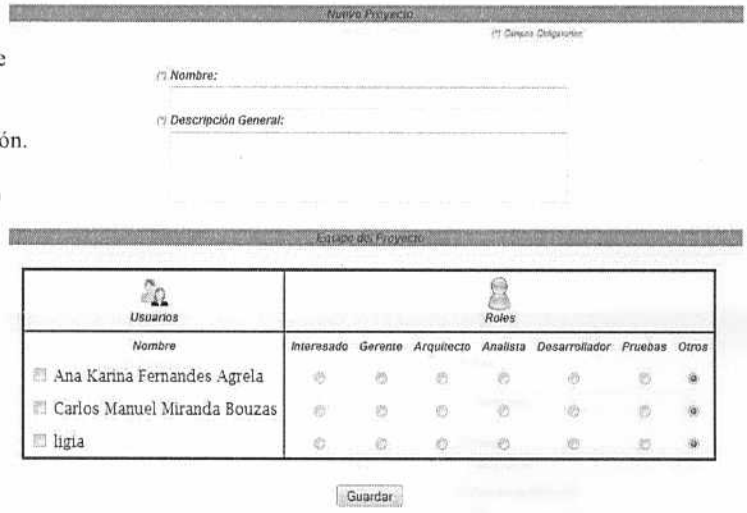
Ingresar a la Herramienta:

1. Seleccionar el Proyecto.
2. Ingresar *login.*
3. Ingresar *clave.*

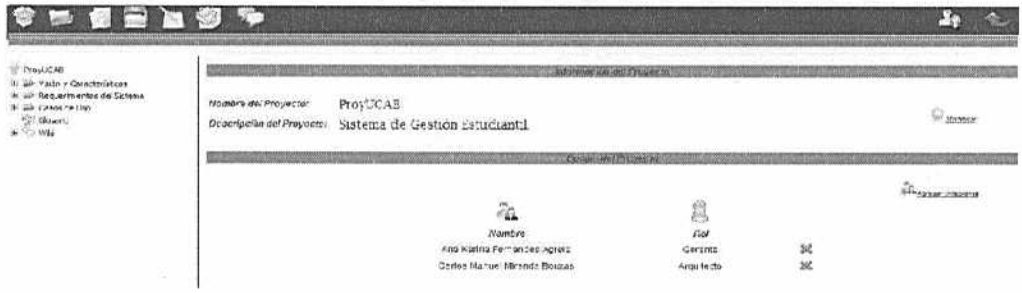


Nuevo Proyecto:

1. Ingresar nombre del proyecto
2. Breve descripción.
3. Elegir el equipo del proyecto.



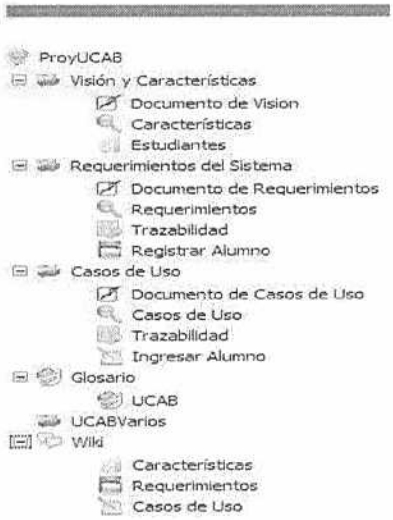
Información básica del Proyecto:



Elementos del Proyecto:

1. Elegir los elementos del proyecto que desea gestionar.

2. Aquí se listan las características, requerimientos, casos de uso, término, comentarios, documento de visión, documento de requerimientos del sistema y trazabilidad



Nueva Característica:

1. Seleccionar *Nueva Característica* de la barra de menú superior.

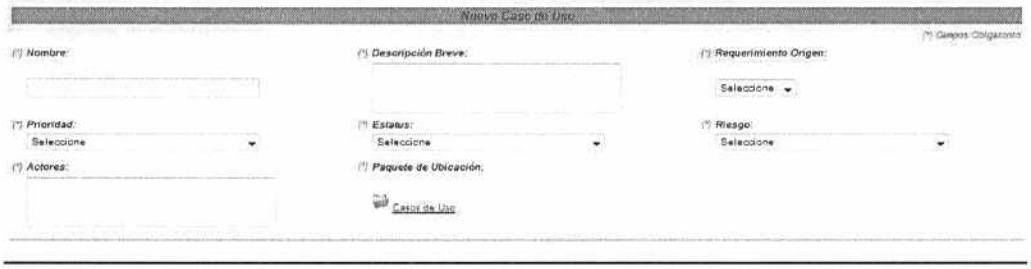
2. Ingresar los campos obligatorios *nombre, descripción, tipo, estatus, prioridad, riesgo, origen, iteraciones planificadas, ubicación.*

3. No es obligatorio completar el resto de los campos que proporciona información complementaria sobre la característica.

Nuevo Requerimiento:

1. Seleccionar *Nuevo Requerimiento* de la barra de menú superior.
2. Ingresar la información obligatoria del requerimiento *nombre, descripción, tipo, característica origen, prioridad, estatus, riesgo, iteraciones planificadas, ubicación.*
3. Si desea complete el resto de la información del requerimiento.

Nuevo Caso de Uso:



1. Seleccione la opción de *Nuevo Caso de Uso* de la barra de menú superior.

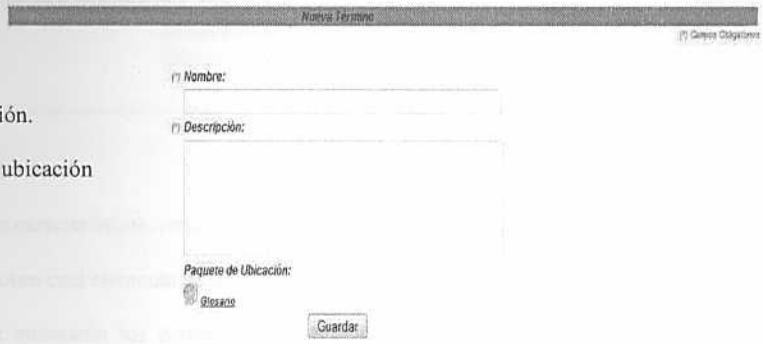
2. Registre la información obligatoria del caso de uso *nombre, descripción, requerimiento origen, prioridad, estatus, riesgo, actores, ubicación.*

3. Adicionalmente puede indicar la *precondición, flujo y sub flujos básicos de eventos*

como otra información relevante del caso de uso.

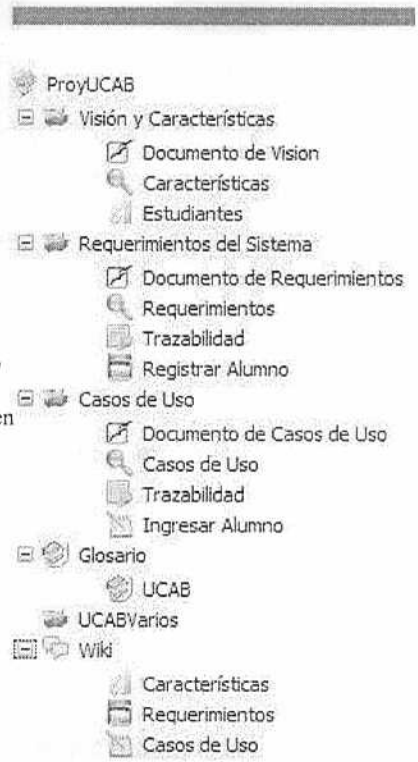
Nuevo Término:

1. Ingresar el nombre del término.
2. Ingrese la descripción.
3. Elija el paquete de ubicación



Árbol:

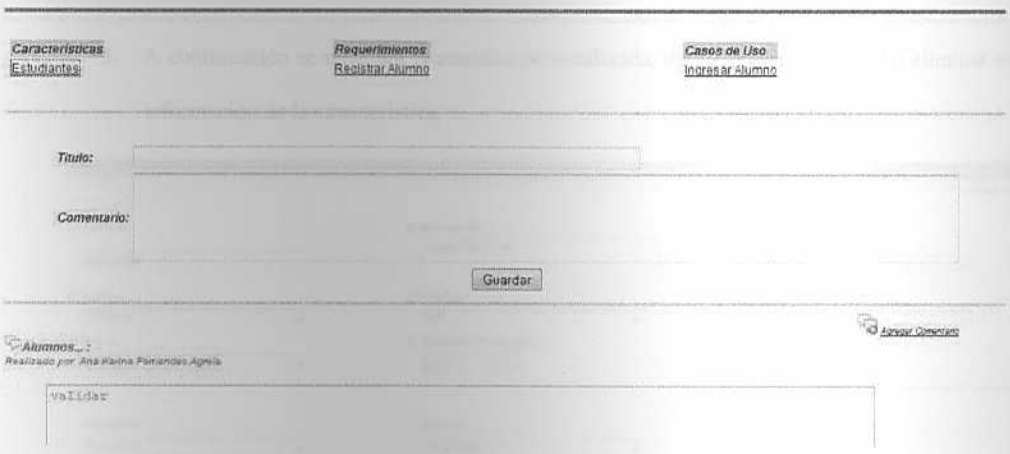
1. Todo los elementos del proyecto se organizan por paquetes en el árbol.
2. Si desea realizarse una modificación de cualquier elemento solo basta con seleccionarlo del árbol.
3. El documento de visión y el documento de requerimientos del sistema se encuentran en los respectivos paquetes en el árbol.



Comentarios:

Características	*Requerimientos*	*Casos de Uso*
Estudiantes	Registrar Alumno	Ingresar Alumno

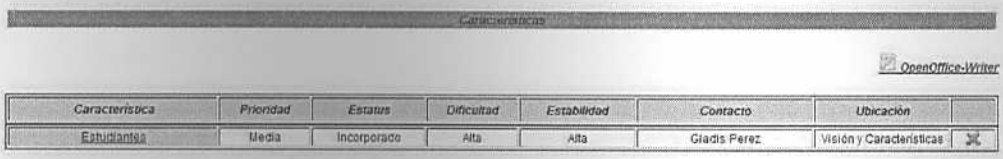
1. Seleccionar *WIKI* del árbol.
2. Se listaran todas las características, requerimientos y casos de uso del proyecto.
3. Debe seleccionar sobre cual elemento desea realizar el comentario.
4. A continuación se mostrarán los comentarios que hay sobre esos elementos y la opción de agregar uno nuevo.



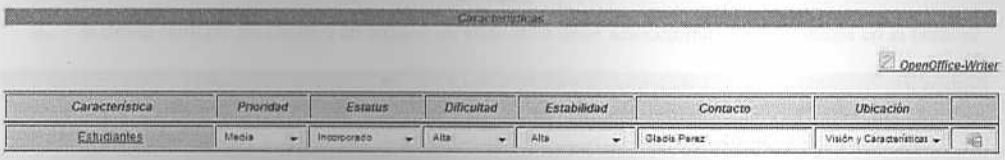
5. Ingrese el título (breve información sobre el comentario).
6. Ingrese el comentario.

Gestión de Características:

1. Puede listar todas las características del proyecto a través de la opción *Características* del árbol

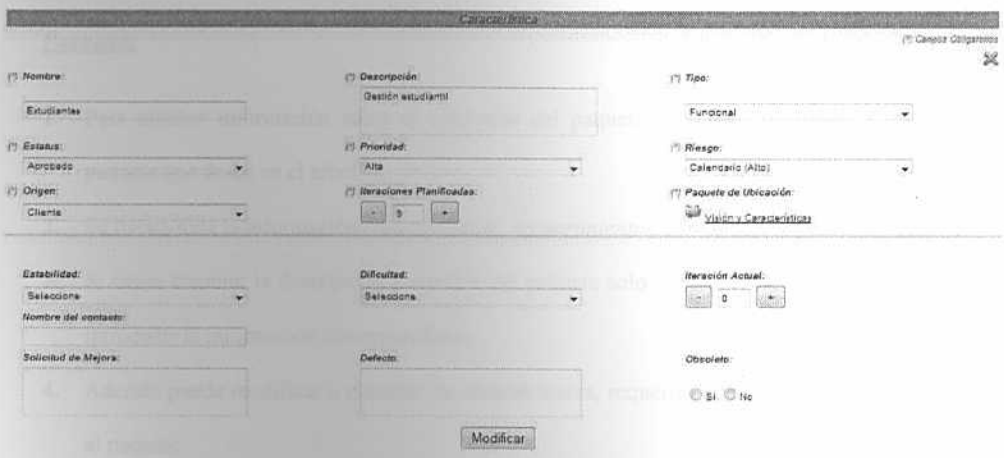


2. Si desea realizar un cambio en alguna de ellas solo debe seleccionarlas ubicándose en el nombre.



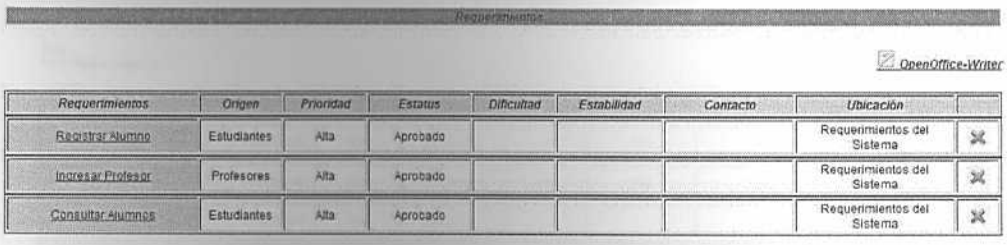
3. Si desea eliminar alguna característica solo debe presionar el icono equis (X).
4. Si desea modificar toda la información de la característica debe dirigirse al árbol, y presionar sobre el nombre de la característica.

5. A continuación se mostrará la consulta personalizada, donde podrá modificar o eliminar toda la información de la característica.

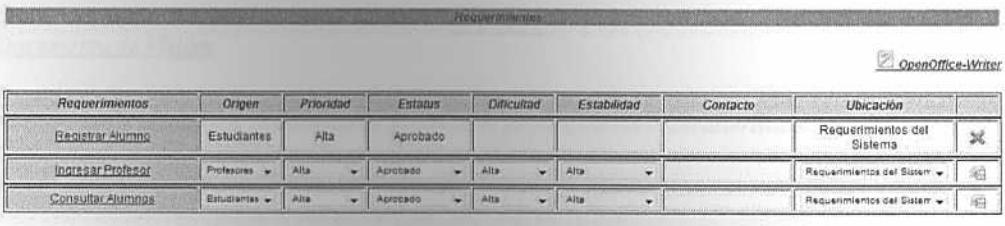


Gestión de Requerimientos:

1. Puede listar todos los requerimientos del proyecto a través de la opción *Requerimientos* del árbol



2. Si desea realizar un cambio en alguno de ellos solo debe seleccionarlos ubicándose en el nombre.

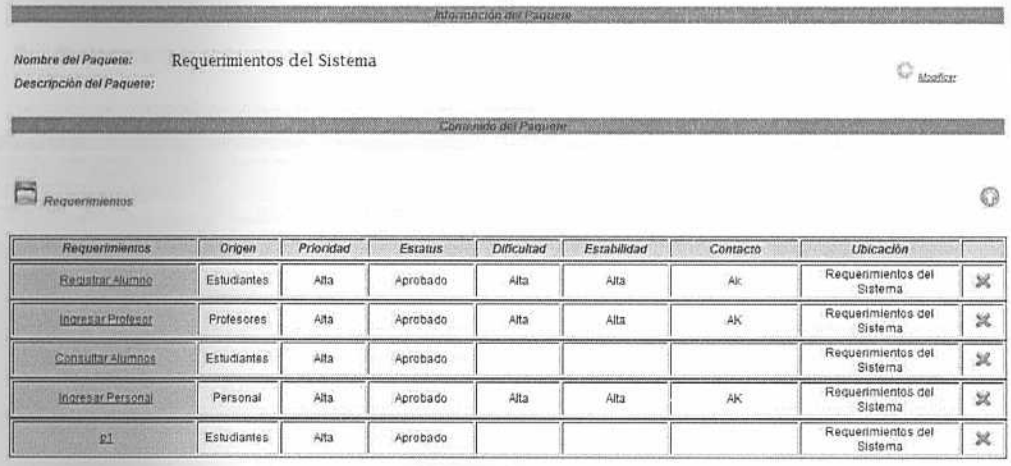


3. Si desea eliminar algún requerimiento solo debe presionar el icono equis (X).

4. Si desea modificar toda la información de la característica debe dirigirse al árbol, y presionar sobre el nombre de la característica.

Paquetes:

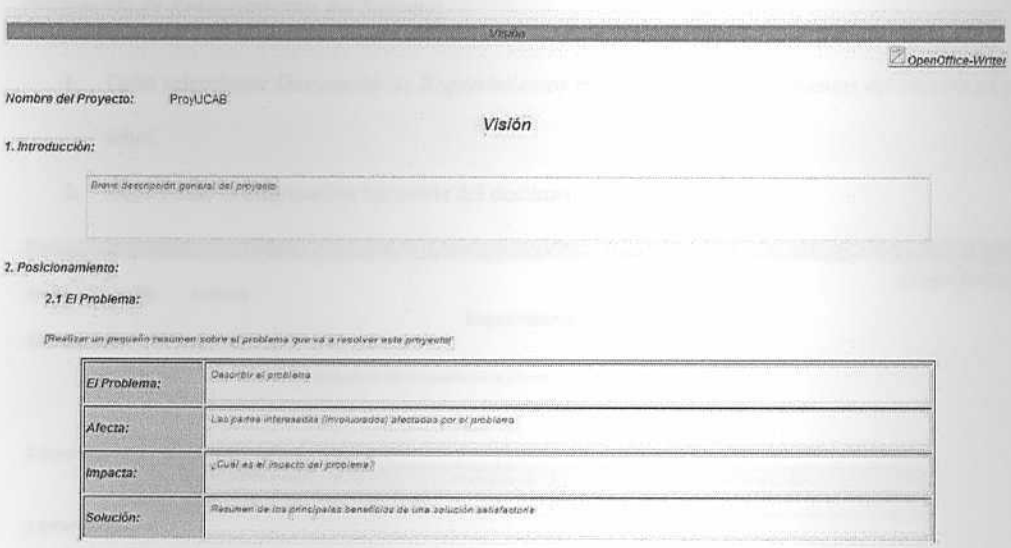
1. Para obtener información sobre el contenido del paquete solo debe presionar el nombre del paquete que desee en el árbol.
2. Se listará toda la información *características, requerimientos, términos* que posea el paquete.
3. Si desea cambiar la descripción o nombre del paquete solo debe presionar el icono *Modificar* e introducir la información correspondiente.
4. Además puede modificar o eliminar las características, requerimientos y términos pertenecientes al paquete.



Requerimientos	Origen	Prioridad	Estatus	Dificultad	Estabilidad	Contacto	Ubicación	
Registrar Alumno	Estudiantes	Alta	Aprobado	Alta	Alta	Ak	Requerimientos del Sistema	
Ingresar Profesor	Profesores	Alta	Aprobado	Alta	Alta	AK	Requerimientos del Sistema	
Consultar Alumnos	Estudiantes	Alta	Aprobado				Requerimientos del Sistema	
Ingresar Personal	Personal	Alta	Aprobado	Alta	Alta	AK	Requerimientos del Sistema	
p1	Estudiantes	Alta	Aprobado				Requerimientos del Sistema	

Documento de Visión:

1. Debe seleccionar *Documento de Visión* en el paquete *Visión y Características* en el árbol.
2. Debe llenar la información necesaria de la Visión del Proyecto.



Visión

OpenOffice-Writer

Nombre del Proyecto: ProyUCAB

Visión

1. Introducción:

Breve descripción general del proyecto

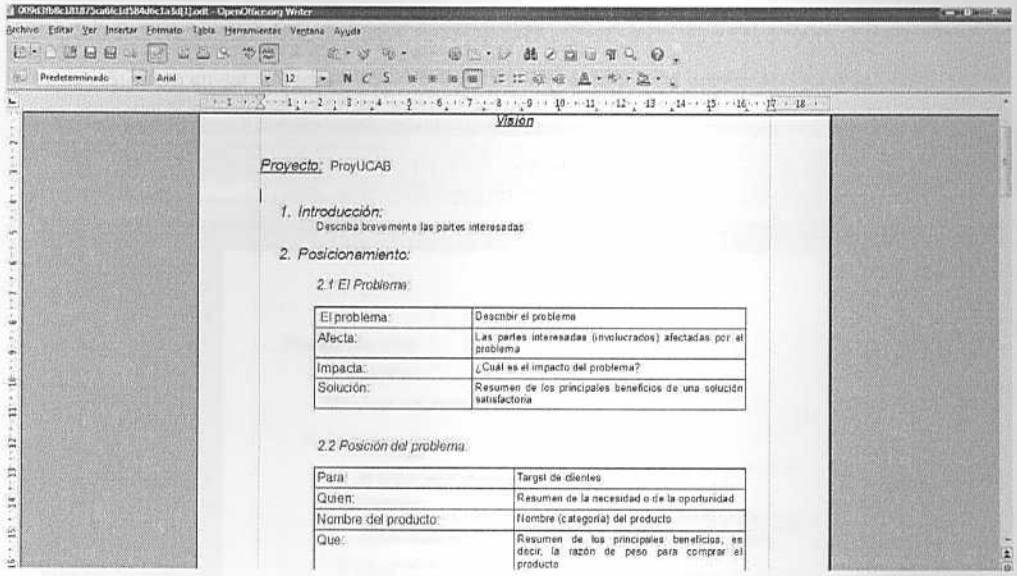
2. Posicionamiento:

2.1 El Problema:

[Realizar un pequeño resumen sobre el problema que va a resolver este proyecto]

El Problema:	Describir el problema
Afecta:	Las partes interesadas (involucrados) afectadas por el problema
Impacta:	¿Cuál es el impacto del problema?
Solución:	Resumen de los principales beneficios de una solución satisfactoria

3. Si desea enviar el documento a OpenOffice Writer sólo debe presionar el icono de OpenOffice.



Archivo Editar Ver Insertar Formato Tabla Herramientas Ventana Ayuda

Predeterminado Arial 12

Visión

Proyecto: ProyUCAB

1. Introducción:
Describa brevemente las partes interesadas

2. Posicionamiento:

2.1 El Problema:

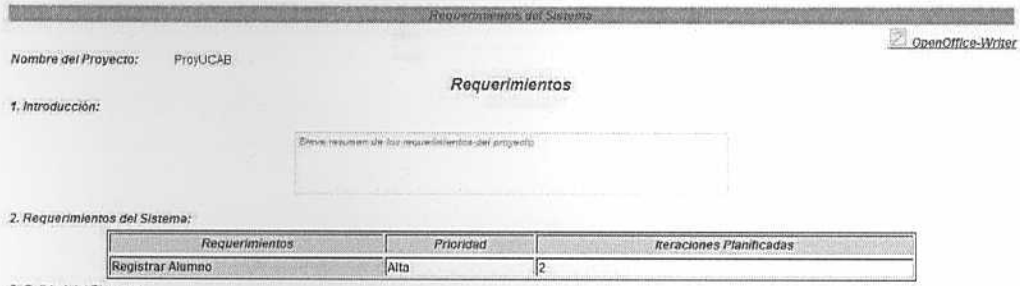
El problema:	Describir el problema
Afecta:	Las partes interesadas (involucrados) afectadas por el problema
Impacta:	¿Cuál es el impacto del problema?
Solución:	Resumen de los principales beneficios de una solución satisfactoria

2.2 Posición del problema:

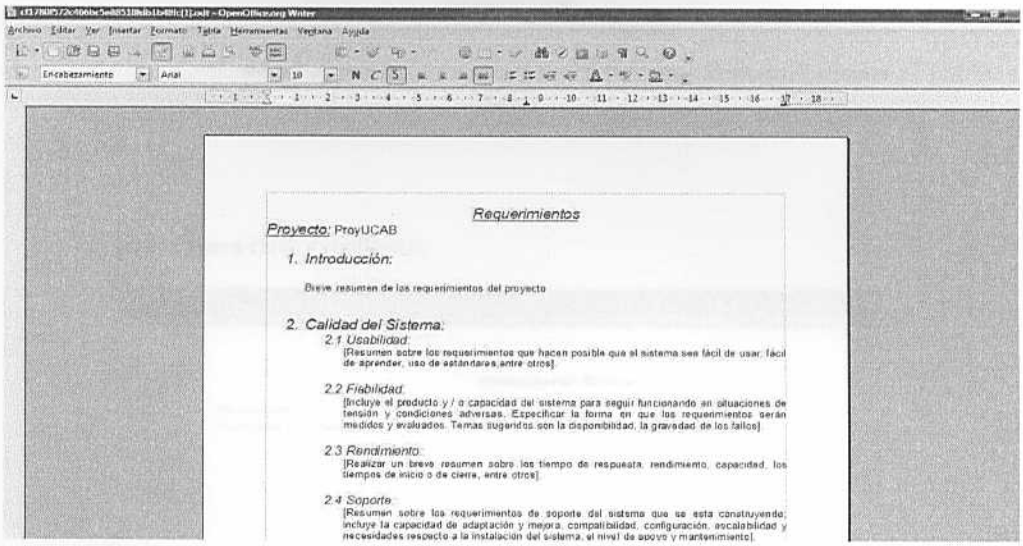
Para:	Target de clientes
Quien:	Resumen de la necesidad o de la oportunidad
Nombre del producto:	Nombre (categoría) del producto
Que:	Resumen de los principales beneficios, es decir, la razón de peso para comprar el producto

Documento de Requerimientos del Sistema:

1. Debe seleccionar *Documento de Requerimientos* en el paquete *Requerimientos del Sistema* en el árbol.
2. Debe llenar la información necesaria del documento.

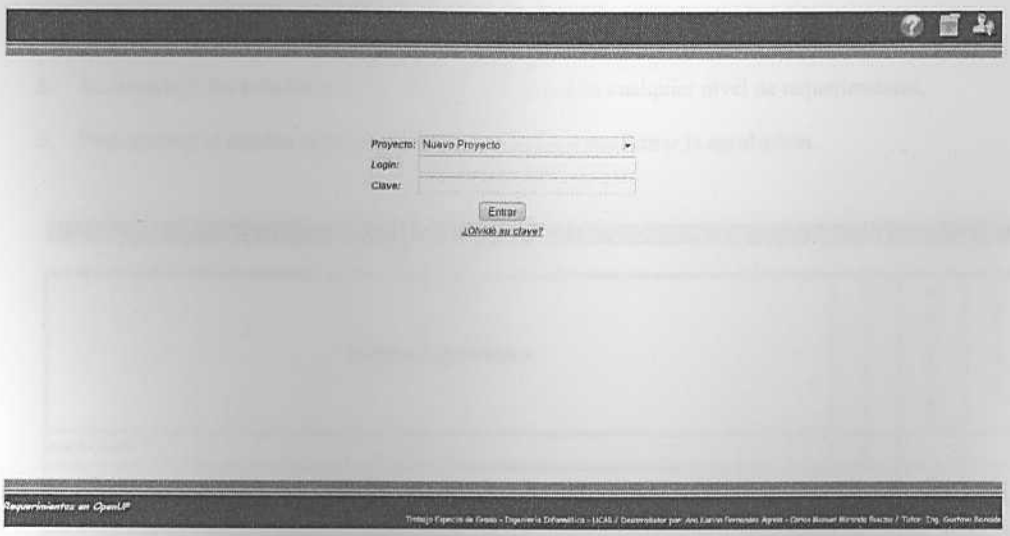


3. Si desea enviar el documento a OpenOffice Writer sólo debe presionar el icono de OpenOffice.

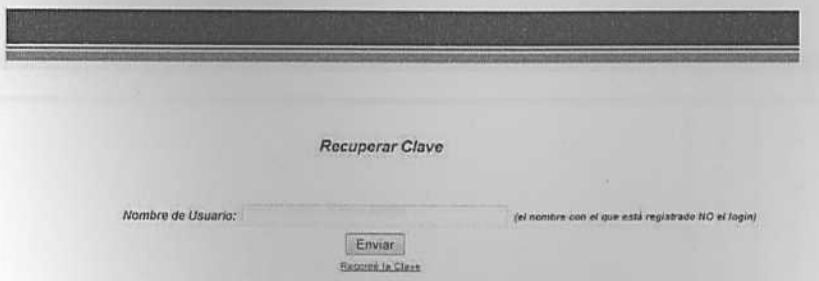


Gestión de Clave:

1. Debe pulsar el enlace *¿Olvidó su clave?*



2. Debe ingresar el nombre de usuario



3. Ingrese la nueva clave y confírmela.

Trazabilidad:

1. Para observar la trazabilidad entre los distintos niveles de requerimientos, sólo debe presionar el enlace trazabilidad en el árbol.
2. Se mostraran las relaciones y los cambios efectuados en cualquier nivel de requerimientos.
3. Para aprobar el cambio solo debe presionar el icono y confirmar la aprobación.

Trazabilidad

Características - Requerimientos	Estudiantes	Profesores	Personal
Registrar Alumno	▲		
Ingresar Profesor		▲	
Consultar Alumnos	▲		
Ingresar Personal			▲

Trazabilidad

Requerimientos - Casos de Uso	Registrar Alumno	Ingresar Profesor	Consultar Alumnos	Ingresar Personal
Ingresar Alumno	▲			
Ingresar Alumno	⇆			
Reistrar Docente		▲		
Listado de alumnos			▲	