UNIVERSIDAD MAYOR DE SAN ANDRÉS

FACULTAD DE CIENCIAS PURAS Y NATURALES

CARRERA DE INFORMATICA

PROYECTO DE GRADO

SISTEMA WEB DE CONTROL Y SEGUIMIENTO DE PREDIOS USANDO AGENTES INTELIGENTES

CASO: Instituto Nacional de Reforma Agraria – Jefatura Regional Valles

PARA OPTAR AL TÍTULO DE LICENCIATURA EN INFORMÁTICA

MENCIÓN: INGENIERÍA DE SISTEMAS INFORMÁTICOS

POR: JOSE LUIS CONDORI TOLA

TUTOR METODOLOGICO: M.Sc. MIGUEL COTAÑA MIER

ASESOR: Lic. GERMAN HUANCA TICONA

LA PAZ – BOLIVIA

2013

UNIVERSIDAD MAYOR DE SAN ANDRÉS
FACULTAD DE CIENCIAS PURAS Y NATURALES
CARRERA DE INFORMÁTICA

LA CARRERA DE INFORMÁTICA DE LA FACULTAD DE CIENCIAS PURAS Y NATURALES PERTENECIENTE A LA UNIVERSIDAD MAYOR DE SAN ANDRÉS AUTORIZA EL USO DE LA INFORMACIÓN CONTENIDA EN ESTE DOCUMENTO SI LOS PROPÓSITOS SON ESTRICTAMENTE ACADÉMICOS.

LICENCIA DE USO

El usuario está autorizado a:

a) visualizar el documento mediante el uso de un ordenador o dispositivo móvil.
b) copiar, almacenar o imprimir si ha de ser de uso exclusivamente personal y privado.
c) copiar textualmente parte(s) de su contenido mencionando la fuente y/o haciendo la referencia correspondiente respetando normas de redacción e investigación.

El usuario no puede publicar, distribuir o realizar emisión o exhibición alguna de este material, sin la autorización correspondiente.

Dedicatoria

A mis queridos padres Rufina y Carlos por brindarme su apoyo incondicional en todo momento para seguir adelante…

Gracias.

AGRADECIMIENTOS

Primero agradecer a Dios por darme un día más de vida, que guía mis pasos y por dotarme de personas maravillosas e importantes que se encuentran a mi lado.

A mi tutor metodológico M.Sc. Miguel Cotaña Mier por guiar el desarrollo de este proyecto.

Al Licenciado German Huanca Ticona, por el apoyo, supervisión y correcciones que me brindo a lo largo de esteproyecto.

A la Jefa Regional VallesAbog. Maria Luz Lopez Pecho, al Ing. Arturo Mamani Velasquez, por la oportunidad y apoyo que me brindaron parala culminación de este proyecto.

A Alison Danilzapor el apoyo, compañía y amor que me dedica día adía.

A mis amigos por la comprensión en todos los años de estudio queestuvimos juntos.
….y a todos aquellos que hicieron posible la confección y elaboración de esteproyecto.

RESUMEN

En la actualidad las tecnologías de información han aumentado y evolucionadorápidamente y esto conlleva a que toda entidad que haga manejo de una grancantidad de información, tenga que encontrar la forma de organizarla y controlarlaeficientemente.

Los sistemas de control de información han llegado a ser una herramientaindispensable para estas entidades, por lo cual la mayor parte se inclina por haceruso de un sistema que logre la organización de su información.

En este caso se realizará un Sistema Web de Control y Seguimiento de Predios Usando Agentes Inteligentes para el Instituto Nacional de Reforma Agraria específicamente en la Jefatura Regional Valles. Los problemas en esta jefatura radicaban en quela gran cantidad de información mal almacenada producía una gran pérdida detiempo al momento de dar reportes de lospredios enviados por las oficinas departamentales de Cochabamba, Chuquisaca y Tarija; mucho más al tenerque realizar seguimiento del estado actual de los predios y poder viabilizar el proceso de cada predio.

En el presente documento se dará una explicación de cómo se dará solución alos problemas de esta jefatura, comenzando con la identificación de los problemasprincipales. Una explicación de las herramientas a utilizarse y finalmente la forma enque se hará uso de estas herramientas para lograr un sistema confiable y eficiente.

El resultado de este proyecto de implementación de un sistema, será el de lograrque la jefatura logre una gran evolución en el ámbito de la información.

INDICE

INDICE DE FIGURAS

INDICE DE TABLAS

CAPITULO I
MARCO REFERENCIAL

1.1. INTRODUCCION

Las Nuevas Tecnologías de la Información y comunicación (NTIC´s) se han venido desarrollando a lo largo de los años y han impactado en el movimiento y desarrollo de la humanidad. Han cambiado los estilos de vida y han logrado establecerse en todo el mundo. De la mano de la globalización están saturadas en lo principal de las necesidades humanas; es decir, son básicas para el exitoso progreso de las sociedades, al igual que el conocimiento de su correcto uso.

Así mismo, las TIC´s están presentes en todas las esferas de nuestra vida, prácticamente es imposible vivir sin ellas, puesto que se han implantado, y se han impuesto en la vida humana; la gente de negocios debe asimilar vastas cantidades de información de nuevos productos, del mercado y de la competencia. Y mientras Internet ofrece grandes oportunidades, presenta a la vez más retos competitivos.

La gran cantidad de información generada en las instituciones públicas y privadas exige una solución innovadora, ya que si bien existen ya muchas formas de solucionar el problema del manejo manual de su información, esta no es suficiente por ser sistemas estáticos. Por tal motivo una nueva propuesta informática es la implementación de agentes inteligentes.

Los agentes inteligentes es una entidad software que “piensa”, pues tienen la capacidad de aprender y tomar decisiones básicas. Dicho aprendizaje se da por observación del mundo digital en que habitan, y a través de interfaces de usuario. Se usan para realizar tareas específicas a nombre del usuario, funcionando de forma semi-autónoma y pudiendo comunicarse con el usuario y con los recursos del sistema.

La Jefatura Regional Valles trabaja con una vasta cantidad de información y es por eso que se encuentra en la necesidad de implementar un software que facilite su buen desempeño y haga de esta entidad una Jefatura competitiva y eficiente.

Para este caso y en respuesta a sus necesidades se desarrollará un software usando agentes inteligentes que le permitirá automatizar y administrar la información generada en la Jefatura Regional Valles.

1.2. ANTECEDENTES

El desarrollo de la programación orientada a agentes, son un poco diferentes a los Objetos tradicionales que acostumbramos modelar. Los objetos que modelamos generalmente son documentos u objetos tangibles: fichas de inventario, transacciones, operaciones y cualquier cosa cuyos atributos sólo se llenan y a partir de eso desarrollan un ciclo de vida bien definido y relativamente simple en la mayoría de los casos. Los agentes por el contrario son abstracciones intangibles y cuyos elementos más importantes son las metas que persiga y la adaptabilidad que tenga para conseguir esas metas, un proceso diferente y un elemento central diferente. Parece que las metodologías de diseño clásicas se nos quedan cortas con este nuevo concepto y por lo tanto nace una nueva generación de metodologías de desarrollo: AUML que es una extensión del UML clásico para modelar Agentes, será de mucha utilidad en este caso por tener una similitud con UML y así no causar dificultades de comprensión.

Este agente inteligente será un programa especialmente concebido para realizar ciertas tareas de manera autónoma. Esta herramienta permitirá ganar tiempo en la vigilancia y la colecta de información de interés para la Jefatura Regional Valles. Se definirán parámetros de la tarea que realizará de manera autónoma el agente, luego el agente informa de los resultados al usuario.

Por otro lado se ha visto que en otros casos se emplea las metodologías de diseño tradicionales. Estas metodologías hacen que el trabajo sea más largo debido a sus numerosas normas y políticas que logran cierta resistencia a los cambios. Por este motivo se ha optado por la metodología ágil SCRUM que brindará mayor flexibilidad y mejora continua de los procesos.

También se vio que en proyectos similares no toman en cuenta el principio de usabilidad, muchas veces se desarrolla sistemas con innumerables procesos que no le son de utilidad al usuario más al contrario los confunde y sólo es un espacio sin usar en el software. En este proyecto se pondrá los procesos que específicamente pidió la Jefatura Regional Valles para cubrir sus necesidades y de tal manera que le será fácil de manejar.

En este sentido la Jefatura Regional Valles en coordinación con las Oficinas departamentales de Cochabamba, Chuquisaca y Tarija; tiene como principal actividad agilizar la proceso de titulación de predios el cual empieza con el envió de personal tanto jurídico como técnico a las departamentales con el objetivo de aprobar los predios que posteriormente serán enviados a la jefatura regional valle, luego elaborar y enviar a firma de la resolución de saneamiento, envió de la resolución a las departamentales para su correspondiente notificación a los beneficiaros y enviar la resolución a la Unidad de Titulación. De acuerdo al proceso mencionado con anterioridad también la Jefatura Regional Valles debe proporcionar reportes sobre estados de predios a los beneficiarios

y unidades relacionadas y de esta forma cumplir las metas de acuerdo al Programa Operativo Anual POA de la institución.

1.2.1. PROYECTOS SIMILARES

Gestión de Historias Clínicas y Cuadros Estadísticos Aplicando Agentes Inteligentes

Autor: Vanesa Paola Cárdenas Capari

Institución: Universidad Mayor de San Andrés – Carrera de Informática

Año: 2011

El sistema se desarrolló con la metodología ágil Scrum y cumple con los requerimientos del área de afiliación, asignación de consultas y el diagnóstico médico aplicando agentes inteligentes para el análisis de datos del paciente y el diagnóstico médico

Diseño de un Sistema Inteligente de Gestión de Almacenes para Empresas Manufactureras de Plástico Caso Madepa S.A. Unidad de Negocios Plásticos

Autor: Rubén Winsor Del Arroyo Verastegui

Institución: Universidad Católica Boliviana "San Pablo" – Carrera de Ingeniería de Sistemas

Año: 2010

Este sistema se desarrollo con la metodología RUP y usa agentes inteligentes para la búsqueda de información en los catálogos de su inventario y de esta forma Gestionar de manera eficiente sus almacenes.

Sistema de Información para el Control y Seguimiento de Proyectos Vía Web

Autor: Jhonny Wilson Ibarra García

Institución: Universidad Mayor de San Andrés – Carrera de Informática

Año: 2007

El sistema de información vía web realiza el Control y Seguimiento de Proyectos del Municipio de Mecapaca, facilitando información precisa y confiable que coadyuve a la toma de decisiones.

1.3. PLATEAMIENTO DEL PROBLEMA

En la actualidad la Jefatura Regional Valles carece de información estructurada, altamente confiable y respaldada de predios enviados por las departamentales y de esta manera poder definir estrategias para el cumplimiento de las metas definidas en el Programa Operativo Anual de la Jefatura Regional Valles.

También cabe mencionar que realizado un análisis de la situación actual se localizaron los siguientes problemas los mismos enraizados en la falta de automatización, integración y control de la información.

- No se cuenta con el registro de predios enviados por las departamentales
- No existe un registro de asignación de predios al personal
- Control deficiente de estados de saneamiento de predios
- Deficiente control de metas programadas en el POA
- Demora en la búsqueda de predios
- Dificultad para transmitir información al beneficiario respecto al estado de saneamiento de su predio
- No se cuenta con privilegios y niveles de acceso a la información
- Excesivo tiempo a momento de emitir reportes

1.4. FORMULACION DEL PROBLEMA

Tomando en cuenta lo mencionado anteriormente, podemos formularnos la siguiente pregunta.

¿DE QUE MANERA EL SISTEMA PODRA MEJORAR EL MANEJO DE INFORMACION DE PREDIOS EN LA JEFATURA REGIONAL VALLES DEL INSTITUTO NACIONAL DE REFORMA AGRARIA?

1.5. PLANTEAMIENTO DE OBJETIVOS

1.5.1. OBJETIVO GENERAL

Implementar un **"Sistema Web de Control y Seguimiento de Predios Usando Agentes Inteligentes"**, para el Instituto Nacional de Reforma Agraria – Jefatura Regional Valles.

1.5.2. OBJETIVOS ESPECIFICOS

- Implementar agentes inteligentes que proporcionen listados de predios viables
- Diseñar un modulo el cual registre y controle de forma eficiente los predios enviados por las departamentales
- Implementar un modulo que permita la asignación de predios al personal
- Controlar de manera eficiente los estados de saneamiento de predios
- Desarrollar una interfaz web donde se podrá obtener información mediante búsquedas avanzadas
- Diseñar una interfaz web para la generación de reportes
- Establecer privilegios y niveles de acceso a la información
- Aplicar la metodología ágil SCRUM para el desarrollo del sistema

1.6. JUSTIFICACION

1.6.1. ECONOMICA

El presente proyecto se justifica económicamente, porque el costo de desarrollo es bajo en relación a los beneficios que obtendrá al implementarlo en la Jefatura Regional Valles, puesto que su desarrollo e implementación será realizada con herramientas

actuales las cuales no implican costo ya que las mismas son software libre, por lo tanto no existe ningún inconveniente.

Así también la implementación del Sistema Web de Control y Seguimiento de Predios Usando Agentes Inteligentes, la Jefatura Regional Valles podrá cumplir de forma eficiente las metas fijadas de acuerdo al Plan Operativo Anual POA de la institución lo cual lo justifica económicamente.

1.6.2. SOCIAL

Se justifica socialmente por que la información que proporciona el Sistema Web de Control y Seguimiento de Predios Usando Agentes Inteligentes, se constituye en listados o reportes y de esta manera brindar una mejor y eficiente comunicación con el Director General de Saneamiento. Puesto que este proyecto tiene el fin de incrementar el prestigio y confiabilidad de la Jefatura Regional Valles, proporcionando información para mejorar el proceso de saneamiento de tierras en Bolivia.

Así también se justifica a nivel Jefatura Regional Valles puesto que el sistema brindara al personal un control eficiente de los predios asignados y de esta forma viabilizar el envió del predio a firma de resolución de saneamiento, envío de resolución a notificación y remisión a la Unidad de Titulación.

Con la implementación del Sistema Web de Control y Seguimiento de Predios los beneficiarios tendrán atención rápida y precisa respecto al estado y avance de su predio.

1.6.3. PRACTICA

La implementación del Sistema Web de Control y Seguimiento de Predios ayudara a minimizar el tiempo que un predio demora desde su ingreso hasta la remisión a la Unidad de Titulación y de esta forma cumplir con las metas programadas en el POA. Así

también contar con reportes con información precisa y confiable de los predios radicados en la Jefatura Regional Valles.

También poder reducir el riesgo de extravió de las carpetas correspondientes a los predios puesto que el sistema nos permitirá contar con un registro de asignación de predios al personal de la Jefatura Regional Valles.

1.6.4. TECNOLOGICA

Las Tecnologías de la Información y las Comunicación (TIC) son incuestionables y están ahí, forman parte de la cultura tecnológica que nos rodea y con la que debemos convivir. Amplían nuestras capacidades físicas y mentales. Y las posibilidades de desarrollo social. Sus principales aportaciones a las actividades humanas se concretan en una serie de funciones que nos facilitan la realización de nuestros trabajos porque, sean éstos los que sean, siempre requieren una cierta información para realizarlo, un determinado proceso de datos y a menudo también la comunicación con otras personas; y esto es precisamente lo que nos ofrecen las TIC.

Consideremos las actividades o tareas que un agente humano cualquiera llega a realizar: buscar información de un predio determinado por un usuario, revisar la viabilidad de un predio, generar una lista de los predios más requeridos y viables, entre muchas otras. Todas estas tareas y muchas otras más pueden automatizarse de manera que se realicen incluso sin interacción con los usuarios en ciertos casos, esto se logrará con el uso de Agentes Inteligentes.

1.7. ALCANCES Y APORTES

1.7.1. ALCANCES

El Sistema Web de Control y Seguimiento de Predios Usando Agentes Inteligentes, abarcara información de los predios enviados a la Jefatura Regional Valles por las

direcciones departamentales de Cochabamba, Chuquisaca y Tarija. Generara reportes en cualquier momento de forma tabular y no así información representada geográficamente.

El presente sistema será implementado bajo un interfaz web a nivel intranet y de esta manera tener una herramienta de control y seguimiento. Así también el sistema realizara la formulación solo de las metas programadas de la Jefatura Regional Valles.

También cabe mencionar que el sistema no implementara el proceso de generación del título ejecutorial de los predios puesto que este proceso corresponde a la Unidad de Titulación.

1.7.2. APORTES

La implementación del presente sistema permitirá cubrir las necesidades para el mejor control y seguimiento que debe realizar la Jefatura Regional Valles a predios y las metas programadas en el POA Institucional, logrando así un excelente funcionamiento.

Se dará uso de una metodología ágil SCRUM, para el desarrollo del sistema por ser ésta una manera óptima.

Para el mejor funcionamiento del sistema se ha propuesto utilizar agentes inteligentes. Los agentes inteligentes liberan de la tarea de discriminar entre un conjunto de información que se tiene acerca de un tema para poder obtener la información que sea más conveniente para su uso.

Un agente utiliza la información adquirida en el pasado para tomar decisiones en situaciones futuras en las que se encuentre.

1.8. METODOS Y MEDIOS DE INVESTIGACION

El método de investigación el cual se utilizara para el desarrollo del Proyecto de Grado será el Método Científico, ya que se pretende seguir una disciplina sistemática y controlada durante todo el desarrollo del presente proyecto de grado.

Definido el objetivo general y objetivos específicos se procederá a recopilar, organizar y describir metodologías y modelos de flujo de información lo cual fundamenta que el tipo de investigación a utilizar serán la exploratoria en un inicio y descriptiva posteriormente.

CAPITULO II
MARCO TEORICO

2.1. ANTECEDENTES

2.1.1. INSTITUTO NACIONAL DE REFORMA AGRARIA

El Instituto Nacional de Reforma Agraria, es una entidad pública descentralizada del Ministerio de Desarrollo Rural, Agropecuario y Medio Ambiente, con jurisdicción nacional, personalidad jurídica y patrimonio propio

Su misión proporcionar servicios que coadyuven a la seguridad jurídica del derecho propietario de la tierra, a través del Catastro Rural Legal y el Saneamiento; la distribución de la tierra, la emisión de títulos, certificaciones catastrales rurales legales de saneamiento; el adecuado desarrollo de programas de asentamientos humanos a favor de todas aquellas personas involucradas en el tema agrario de una manera relevante y pertinente a sus demandas.

Su visión del INRA es ser una entidad transparente, con alta credibilidad y confiabilidad en la ejecución de las normas técnicas y jurídicas, garantizando el derecho propietario sobre la tenencia de la tierra de manera eficiente y eficaz en un marco de equidad social.

El Instituto Nacional de Reforma Agraria (INRA) tiene la siguiente estructura orgánica:

- Dirección Nacional
- Direcciones Departamentales
- Jefaturas Regionales (Altiplano, Valles y Llanos)

Mediante Resolución Administrativa No 053/2006 de 09 de marzo se define la Estructura Orgánica interna vigente de la Dirección Nacional y Direcciones Departamentales.

En este sentido la Jefatura Regional Valles en coordinación con las Oficinas departamentales de Cochabamba, Chuquisaca y Tarija; tiene como principal actividad agilizar la proceso de titulación de predios el cual empieza con la firma de Resolución final de saneamiento, envió de la resolución a las departamentales para su correspondiente notificación a los beneficiaros, solicitar certificación al Tribunal Agrario Nacional y enviar la resolución a la Unidad de Titulación. De acuerdo al proceso mencionado con anterioridad también la Jefatura Regional Valles debe proporcionar reportes sobre estados de predios a los beneficiarios y unidades relacionadas.

2.2. INGENIERIA DE SOFTWARE

La noción de ingeniería de software fue propuesta inicialmente en 1968 en una conferencia para discutir lo que en ese entonces se llamo la “crisis del software”. Esta crisis del software fue el resultado de la introducción de las nuevas computadoras hardware basadas en circuitos integrados. Su poder hizo que las aplicaciones hasta ese entonces irrealizables fueran una propuesta factible. El software resultante fue de órdenes de magnitud más grande y más complejo que los sistemas de software previos.

La experiencia previa en la construcción de estos sistemas mostro que un enfoque informal para el desarrollo de software no era muy bueno. Los grandes proyectos a menudo tenían años de retraso. Constaban mucho más de lo presupuestado, eran irrealizables, difíciles de mantener y con un desempeño pobre. El desarrollo de software estaba en crisis. Los costos de hardware se tambaleaban mientras que los del software se incrementaban con rapidez. Se necesitaban nuevas técnicas y métodos para controlas la complejidad inherente a los sistemas grandes.

Estas técnicas y métodos han llegado a ser parte de la ingeniería del software y son ampliamente utilizadas.

Sin embargo, cuanto más incremente nuestra capacidad para producir software, también lo hará la complejidad de los sistemas de software solicitados. Las nuevas tecnologías resultantes de la convergencia de las computadoras y de los sistemas de comunicación y complejas interfaces graficas de usuario impusieron nuevas demandas a los ingenieros de software.

Debido a que muchas compañías no aplican de forma efectiva las técnicas de la ingeniería del software, demasiados proyectos todavía producen software que es irrealizable, entregando tarde y sobre presupuestado [Sommerville2005].

Algunas definiciones sobre Ingeniería de Software que nosotros podemos a preciar son los siguientes:

- Ingeniería del Software es la aplicación práctica del conocimiento científico en el diseño y construcción de programas de computadora y la documentación asociada requerida para desarrollar, operar (funcionar) y mantenerlos. Se conoce también como desarrollo de software o producción de software. [Koch2002]

- Ingeniería del Software trata del establecimiento de los principios y métodos de la ingeniería a fin de obtener software de modo rentable que sea fiable y trabajo en maquinas reales. [Baresi2001]

- La aplicación de un enfoque sistemático, disciplinado y cuantificable al desarrollo, operación (funcionamiento) y mantenimiento del software; es decir, la aplicación de ingeniería al software.

Como se puede apreciar existen varias definiciones y podremos encontrar muchas sirve ingeniería de software, pero lo cierto es que sin la ingeniería del software los productos que se desarrollarían ahora tendrían bastantes errores, serian de alto costo, difíciles de modificar en su funcionamiento y otros tantos inconvenientes.

También podemos decir que la ingeniería del software es una disciplina de la ingeniería que comprende todos los aspectos de la producción de software desde las etapas iníciales de la especificación del sistema hasta el mantenimiento de este, después de que se utiliza.

La ingeniería del software no solo comprende los procesos técnicos del desarrollo de software, sino también con actividades tales como la gestión de proyectos de software y el desarrollo de herramientas, métodos y teorías de apoyo a la producción de software [Sommerville2005].

2.3. METODOLOGIA DE DESARROLLO SCRUM

Scrum es una metodología de desarrollo muy simple, que requiere trabajo duro porque no se basa en el seguimiento de un plan, sino en la adaptación continua a las circunstancias de la evolución del proyecto. Scrum es una metodología ágil, y como tal:

- Es un modo de desarrollo de carácter adaptable más que predictivo.
- Orientado a las personas más que a los procesos.
- Emplea la estructura de desarrollo ágil: incremental basada en iteraciones y revisiones.

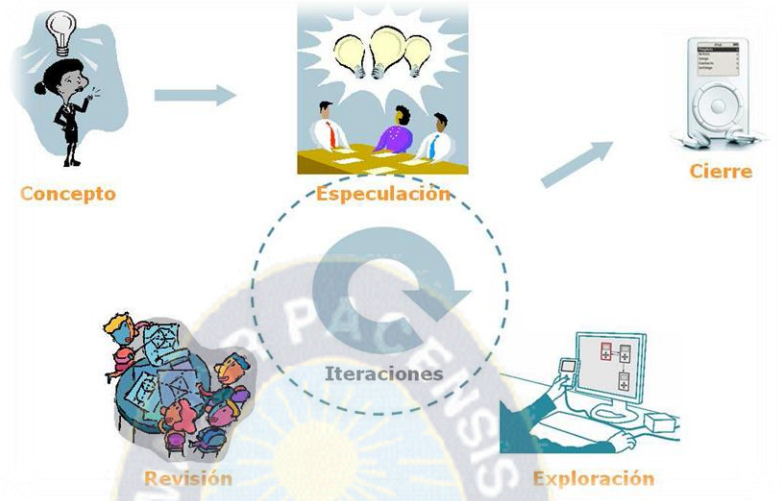


Figura 2.1 Estructura del desarrollo Ágil

Fuente: [Kniberg2007]

Se comienza con la visión general del producto,especificando y dando detalle a las funcionalidadeso partes que tienen mayor prioridad dedesarrollo y que pueden llevarse a cabo en unperiodo de tiempo breve (normalmente de 30días).Cada uno de estos periodos de desarrollo es unaiteración que finaliza con la producción de unincremento operativo del producto.Estas iteraciones son la base del desarrollo ágil, yScrum gestiona su evolución a través dereuniones breves diarias en las que todo elequipo revisa el trabajo realizado el día anterior y el previsto para el día siguiente.

Figura 2.2 Estructura central de Scrum

Fuente: [Kniberg2007]

2.3.1. CONTROL DE LA EVOLUCION DEL PROYECTO

Scrum controla de forma empírica y adaptable la evolución del proyecto, empleando las siguientes prácticas de la gestión ágil:

- **Revisión de las iteraciones:** Al finalizar cada iteración (normalmente 30 días) se lleva a cabo una revisión con todas las personas implicadas en el proyecto. Este es el periodo máximo que se tarda en reconducir una desviación en el proyecto o en las circunstancias del producto.
- **Desarrollo incremental:** Durante el proyecto, las personas implicadas no trabajan con diseños o abstracciones. El desarrollo incremental implica que al final de cada iteración se dispone de una parte del producto operativa que se puede inspeccionar y evaluar.
- **Desarrollo evolutivo:** Los modelos de gestión ágil se emplean para trabajar en entornos de incertidumbre e inestabilidad de requisitos.

  Intentar predecir en las fases iníciales cómo será el producto final, y sobre dicha predicción desarrollar el diseño y la arquitectura del producto no es realista, porque las circunstancias obligarán a remodelarlo muchas veces.

  Para qué predecir los estados finales de la arquitectura o del diseño si van a estar cambiando. En Scrum se toma a la inestabilidad como una premisa, y se adoptan técnicas de trabajo para permitir esa evolución sin degradar la calidad de la arquitectura que se irá generando durante el desarrollo.

  El desarrollo Scrum va generando el diseño y la arquitectura final de forma evolutiva durante todo el proyecto. No los considera como productos que deban realizarse en la primera "fase" del proyecto.
- **Auto–organización:** Durante el desarrollo de un proyecto son muchos los factores impredecibles que surgen en todas las áreas y niveles. La gestión predictiva confía la responsabilidad de su resolución al gestor de proyectos. En Scrum los equipos son auto-organizados (no auto-dirigidos), con margen de decisión suficiente para tomar las decisiones que consideren oportunas.

- **Colaboración:** Las prácticas y el entorno de trabajo ágiles facilitan la colaboración del equipo. Ésta es necesaria, porque para que funcione la auto-organización como un control eficaz cada miembro del equipo debe colaborar de forma abierta con los demás, según sus capacidades y no según su rol o su puesto.

2.3.2. VISION GENERAL DEL PROCESO

Scrum denomina "sprint" a cada iteración dedesarrollo y recomienda realizarlas conduraciones de 30 días.El sprint es por tanto el núcleo central queproporciona la base de desarrollo iterativo eincremental.

Figura 2.3 Estructura general de Scrum

Fuente: [Kniberg2007]

Los elementos que conforman el desarrolloScrum son:

Las reuniones

- **Planificación de sprint:** Jornada de trabajo previa al inicio de cada sprint en la que se determina cuál va a ser el trabajo y los objetivos que se deben cumplir en esa iteración.

- **Reunión diaria:** Breve revisión del equipo del trabajo realizado hasta la fecha y la previsión para el día siguiente.
- **Revisión de sprint:** Análisis y revisión del incremento generado.

Los elementos

- **Pila del producto:** lista de requisitos de usuario que se origina con la visión inicial del producto y va creciendo y evolucionando durante el desarrollo.
- **Pila del sprint:** Lista de los trabajos que debe realizar el equipo durante el sprint para generar el incremento previsto.
- **Incremento:** Resultado de cada sprint

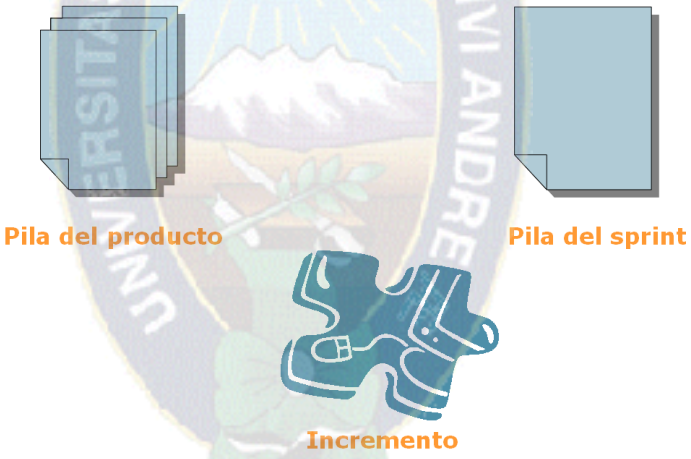


Figura 2.4 Elementos de Scrum

Fuente: [Kniberg2007]

Los roles

Scrum clasifica a todas las personas que intervienen o tienen interés en el desarrollo del proyecto en: propietario del producto, equipo, gestor de Scrum (también Scrum Manager o Scrum Máster) y "otros interesados".

Los tres primeros grupos (propietario, equipo y gestor) son los responsables del proyecto, los que según la comparación siguiente (y sin connotaciones peyorativas) serían los "cerdos"; mientras que el resto de interesados serían las gallinas.
Cerdos y gallinas.

Esta metáfora ilustra de forma muy gráfica la diferencia de implicación en el proyecto entre ambos grupos:

Una gallina y un cerdo paseaban por la carretera. La gallina dijo al cerdo: "Quieres abrir un restaurante conmigo". El cerdo consideró la propuesta y respondió: "Sí, me gustaría. ¿Y cómo lo llamaríamos?". La gallina respondió: "Huevos con beicon". El cerdo se detuvo, hizo una pausa y contestó: "Pensándolo mejor, creo que no voy a abrir un restaurante contigo. Yo estaría realmente comprometido, mientras que tu estarías sólo implicada".

COMPROMETIDOS (cerdos)	**IMPLICADOS (gallinas)**
Propietario del producto Equipo Scrum Manager	Otros interesados (Dirección general Dirección comercial Marketing Usuarios, etc.)

Tabla 2.1 Tabla de roles de Scrum

Fuente: [Kniberg2007]

Propietario del producto: El responsable de obtener el mayor valor de producto para los clientes, usuarios y resto de implicados.

Equipo de desarrollo: grupo o grupos de trabajo que desarrollan el producto.

Scrum Manager: gestor de los equipos que es responsable del funcionamiento de la metodología Scrum y de la productividad del equipo de desarrollo.

2.3.3. VALORES

Scrum es una "carrocería" para dar forma a los principios ágiles. Es una ayuda para organizar a las personas y el flujo de trabajo; como lo pueden ser otras propuestas de formas de trabajo ágil: Cristal, DSDM, etc.

La carrocería sin motor, sin los valores que dan sentido al desarrollo ágil, no funciona.

- Delegación de atribuciones (*empowerment*) al equipo para que pueda auto-organizarse y tomar las decisiones sobre el desarrollo.
- Respeto entre las personas. Los miembros del equipo deben confiar entre ellos y respetar sus conocimientos y capacidades.
- Responsabilidad y auto-disciplina (no disciplina impuesta).
- Trabajo centrado en el desarrollo de lo comprometido.
- Información, transparencia y visibilidad del desarrollo del proyecto.

2.4. INGENIERIA WEB

La World Wide Web e Internet han introducido a la población en general en el mundo de la informática. Compramos fondos de inversión colectivos y acciones, descargamos música vemos películas, obtenemos asesoramiento médico, hacemos reservas de habitaciones en hoteles, vendemos artículos personales, planificamos vuelos en líneas aéreas, conocemos gente, hacemos gestiones bancarias, recibimos cursos universitarios, hacemos la compra es decir, en el mundo virtual se puede hacer todo lo que se necesite. Se puede decir que Internet y la Web son los avances más importantes en la historia informática. Durante los primeros años del siglo veintiuno estas tecnologías han llegado casi a formar parte de nuestra vida diaria.

El componente más usado en el internet es definitivamente la Web. Su característica sobresaliente es el texto remarcado, un método para referencias cruzadas instantáneas. En la mayoría de los Sitios Web, ciertas palabras aparecen en texto de otro color diferente al resto del documento. Por lo general, este texto es subrayado. Al seleccionar

una palabra o frase, uno es transferido al sitio o página relacionada a esa frase. En algunas ocasiones hay botones, imágenes o porciones de imágenes que pueden activarse mediante un clic.

Usando el Web, se tiene acceso a millones de páginas de información. La exploración en la Web se realiza por medio de un software especial denominado Browser o explorador. La apariencia de un Sitio web puede variar ligeramente dependiendo del explorador que use el usuario.

2.5. METODOLOGIA DE DESARROLLO DE APLICACIONES WEB: UWE

UWE (UML-Based Web Engineering) es una propuesta basada en UML y en el proceso unificado para modelar aplicaciones web. Esta propuesta está formada por una notación para especificar el dominio (basada en UML) y un modelo para llevar a cabo el desarrollo del proceso de modelado. Los sistemas adaptativos y la sistematización son dos aspectos sobre los que se enfoca UWE. Además de estar considerado como una extensión del estándar UML, también se basa en otros estándares como por ejemplo: XMI como modelo de intercambio de formato, MOF para la meta modelado, los principios de modelado de MDA, el modelo de transformación del lenguaje QVT y XML. El modelo que propone UWE está compuesto por 6 etapas o sub-modelos:

- **Modelo de Casos de Uso:** modelo para capturar los requisitos del sistema.
- **Modelo de Contenido**: es un modelo conceptual para el desarrollo del contenido.
- **Modelo de Usuario:** es modelo de navegación, en el cual se incluyen modelos estáticos y modelos dinámicos.
- **Modelo de estructura:** en el cual se encuentra la presentación del sistema y el modelo de flujo.
- **Modelo Abstracto:** incluye el modelo a de interfaz de usuario y el modelo de ciclo de vida del objeto.

- **Modelo de Adaptación.**

En cuanto a los requisitos, UWE los clasifica dependiendo del carácter de cada uno. Además distingue entre las fases de captura, definición y validación de requisitos.

2.5.1. UWE Y SU RELACION CON UML

UWE define una extensión del Lenguaje Unificado de Modelado (UML). Ésta, es considerada como una extensión ligera de peso e incluye en su definición tipos, etiquetas de valores y restricciones para las características especificas del diseño Web, las cuales, unidas a las definiciones de UML forman el conjuntos de objetos de modelado que se usarán para el desarrollo del modelo utilizado en UWE.

Las funcionalidades que cubren UWE abarcan áreas relacionadas con la Web como la navegación, presentación, los procesos de negocio y los aspectos de adaptación.

Una de las ventajas de que UWE extienda el estándar UML es la flexibilidad de éste para la definición de un lenguaje de modelado especifico para el dominio web y sobretodo la aceptación universal de dicho estándar en el campo de la ingeniería del software.

Otra gran ventaja es que actualmente existen múltiples de herramientas CASE basadas en UML, con lo cual es relativamente sencillo su utilización y ampliación para utilizar los objetos de modelado definidos en UWE. Estas herramientas se verán en el siguiente punto.

2.5.2. EXTENSIONES UML

Los mecanismos de extensibilidad incorporados permiten a UML ser una especie de especificación abierta que puede cubrir aspectos de modelado no especificados y

contribuir en la solución de problemas de modelado. Estos mecanismos permiten extender la notación y semántica de UML.

2.5.3. MODELOS DE UWE

En esta sección se explicarán los modelos para cada una de los aspectos web que cubre la metodología UWE, recordemos que estos aspectos eran navegación, presentación, los procesos de negocio y adaptación. Así procedemos a explicar con un breve ejemplo cada uno de estos modelos.

2.5.4. MODELO DE CONTENIDO

Este modelo especifica cómo se encuentra relacionados los contenidos del sistema, es decir, define la estructura de los datos que se encuentran alojados en el sitio web.

2.5.5. MODELO DE NAVEGACION

Este modelo indica como el sistema de páginas web del sitio está relacionado internamente. Es decir cómo se enlazan los elementos de navegación.

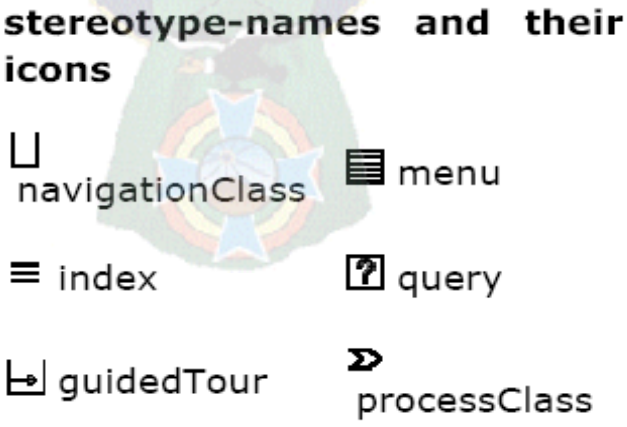


Figura 2.5 Estereotipos del modelo de navegación

Fuente: [Baresi2001]

Para ello se utilizan unidades de navegación llamadas “nodos” conectadas por enlaces de navegación. Estos nodos pueden ser mostrados en la misma página web, no tienen porque estar en páginas diferentes.

2.5.6. MODELO DE PRESENTACION

En este modelo se representan las clases de navegación y de procesos que pertenecen a cada página web. Estos son los elementos que introduce la metodología UWE en este modelo.

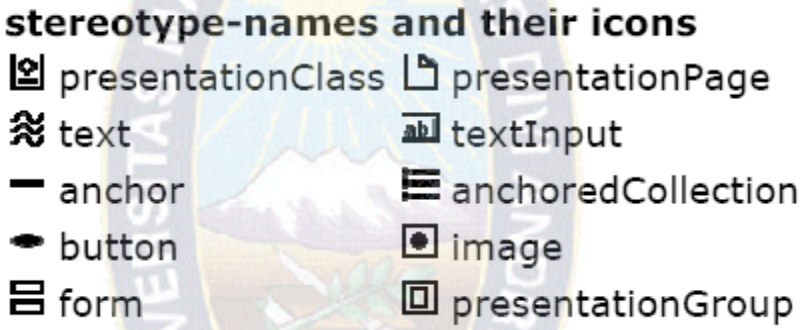


Figura 2.6 Estereotipos del modelo de presentación

Fuente: [Baresi2001]

2.5.7. MODELO DE PROCESO

Esta modelo específica las acciones que realiza cada clase de proceso, en este modelo se incluye:

- **Modelo de estructura de procesos:** Que define las relaciones entre las diferentes clases proceso.
- **Modelo de flujo de procesos:** Que especifica las actividades conectadas con cada proceso. Describe los comportamientos de una clase proceso. Lo que ocurre en detalle dentro de cada una.

2.6. AGENTES INTELIGENTES

Los agentes inteligentes cumplen con los requerimientos para los cuales fueron entrenados. El usuario "delega" en el agente una o varias tareas que debe llevar a cabo quedando a la espera de los resultados. Dichas tareas son a menudo fácil de especificar pero en algunos casos complejas de realizar. Hay que tomar en cuenta que los investigadores en el campo de los agentes inteligentes de software han dado varias definiciones al término, cada uno desde su óptica particular, fundamentada básicamente en la línea de investigación en la cual trabajan (inteligencia Artificial, Ingeniería de Software, Sistemas Autónomos, etc.).

Definición: Un agente software inteligente es un programa que puede realizar tareas específicas para un usuario y posee un grado de inteligencia suficiente para ejecutar parte de sus tareas de forma autónoma y para interactuar con su entorno de forma útil. [Brenner1998]

Figura 2.7 Definición de Agente Inteligente
Fuente: [Brenner1998]

2.6.1. PROPIEDADES DE LOS AGENTES

En base a la definición anterior es posible extraer algunas características que deben tener los agentes: deben ser parte de un ambiente dinámico, deben poder censar su entorno y actuar sobre él, y deben responder según sus objetivos para los cuales fueron diseñados.

Deben entonces poseer una serie de atributos o propiedades que lo definen como agente:

- **Autonomía:** Capacidad de actuar sin la intervención directa de una persona o de otro agente. Un agente debe poder controlar sus propias acciones y estado interno. Una vez que el usuario activa el agente indicado algún objetivo de alto nivel, este actúa independientemente, seleccionando estrategias y monitoreando el progreso en busca de la meta. Si falla con una estrategia, usara otra, pero sin intervención humana o con la mínima indispensable.
- **Aprendizaje:** Un agente aprende en la medida que su conocimiento es completo y no presenta inconsistencias. Este aprendizaje es de parte del usuario, de su ambiente y de los otros agentes con que convive.
- **Colaboración:** Un agente debe colaborar. Debe tener habilidad para interactuar con otros agentes o incluso con alguna persona (el usuario), para solicitar información o bien para exponer los resultados obtenidos de la ejecución de las tareas agendadas. La naturaleza de la comunicación dependerá del tipo de agente con quien se comunique (humanos o no), en ambos casos se deberá establecer un protocolo común de intercambio de información entre ambas partes.

2.6.2. TAXONOMIAS DE AGENTES

Los agentes pueden clasificarse de varias maneras, teniendo en cuenta algunas de las propiedades que poseen o bien haciendo hincapié en alguna en particular. De esta manera puede armarse un árbol taxonómico que abarque todas las combinaciones de propiedades y tareas que se quieran. Nosotros consideramos una clasificación de acuerdo a líneas de investigación y desarrollo de agentes, donde la prioridad es la función u objetivo principal del agente. De acuerdo a lo anterior consideramos la siguiente clasificación:

- Agentes de interfaz
- Agentes colaborativos

- Agentes móviles
- Agentes de recuperación de información

2.6.3. AGENTES DE INTERFAZ

Un agente de interfaz es un software casi – inteligente que asiste a un usuario cuando interactúa con una o más aplicaciones. Son asistentes personales que reducen el trabajo por la sobrecarga de información, por ejemplo el filtrado de los mensajes de correo electrónico o la recuperación de archivos de internet.

2.6.4. AGENTES COLABORATIVOS

Los agentes colaborativos constituyen un sistema multi – agente, es decir, existe más de un agente dedicado a satisfacer los requerimientos de sus usuarios. Para ello, es necesario contar con esquemas de comunicación entre agentes que permitan la cooperación y el intercambio de conocimiento. Además deben poseer un alto grado de autonomía para interactuar con los demás agentes.

2.6.5. AGENTES MOVILES

Los agentes móviles son procesos capaces de "viajar" por una red de computadoras, interactuando con hosts externos, recolectando información en nombre de su dueño y retornando a "casa" luego de completar las tareas establecidas.

2.6.6. AGENTES DE RECUPERACION DE INFORMACION

El objetivo principal de los agentes dedicados específicamente a la recuperación de información es obtener información por el usuario.

Las tecnologías de la información han expandido los horizontes de los usuarios en cuanto a las formas de generar y acceder a la misma. Pero esta amplia variedad de información distribuida plantea desafíos en cuanto a las formas de manejar su complejidad y heterogeneidad.

2.7. LENGUAJE UNIFICADO DE MODELADO PARA AGENTES (AUML)

El lenguaje de Modelado Unificado UML está ganando una amplia aceptación de la representación de los artefactos de la ingeniería de software orientado a objetos. Los agentes son el siguiente paso más allá de los objetos que nos lleva a explorar las extensiones de UML y expresiones dentro de UML para dar cabida a las necesidades distintivas de los agentes.

En comparación con el enfoque tradicional de los objetos, los agentes son autónomos e interactivos. Con base en los estados internos, sus actividades incluyen objetivos y condiciones que rige la ejecución de las tareas definidas. Mientras que los objetos están fuera del control necesario para ejecutar sus métodos, los agentes conocen las condiciones y los efectos esperados de sus acciones y por lo tanto, asumir la responsabilidad de sus necesidades.

2.7.1. DIAGRAMA DE CASOS DE USO DE AGENTES

El análisis de casos de uso ha demostrado ser útil y exitoso en la especificación de requisitos de los sistemas orientados a objetos. La pregunta lógica es entonces puede este tipo de análisis ser utilizado para los sistemas orientados a agentes?, se propone que con algunas modificaciones y la extensión utilizar el análisis de casos e uso puede convertirse en una herramienta útil para el requerimiento de uno o varios agentes y la especificación de la conducta.

El concepto de un agente de formar parte de un entorno que es consciente e interactúa con loa gentes de distingue a los actores. Los actores son entidades externas (los seres humanos u otros sistemas), que interactúan con un sistema de software en particular. Los agentes por otra parte pueden ser parte del sistema de software. El entorno en el que se encuentra puede ser un entorno sintético virtual, un entrono real o una combinación de los dos dependiendo del tipo de agente.

2.7.1.1. IDENTIFICAR LOS AGENTES Y ACTORES

El analista decide que actores y que agentes van a estar involucrados en el sistema de software que se especifica. Para identificar a los actores en un sistema el analista debe determinar quién será el usuario del sistema y con qué sistemas externos el software estará interactuando. La identificación de los agentes en sistema es un poso más difícil.

2.7.1.2. IDENTIFICAR CASOS DE USO PARA LOS AGENTES Y ACTORES

Para cada actor en el sistema de los casos de uso se especifican dibujando un diagrama de casos de uso con una serie de elipses unidas por líneas para el actor. Cada elipse representa un caso de uso o alguna funcionalidad proporcionada por el sistema para el actor. El mismo método puede ser aplicado a la determinación de los casos de uso del agente para losagentes.

Sin embargo en lugar de considerar la funcionalidad de un caso de uso del agente los tipos de comportamientos de un agente que exhibe en el contexto de un entorno debe ser considerado. Por lo tanto para cada agente:

- Decidir sobre las conductas que deben ser exhibidos por cada agente – cada tipo de comportamiento se convierte en un caso de uso de agentes.
- Decidir sobre el tipo de actor – agente y la interacción multi – agente que habrá en el sistema.
- Cada interacción se convierte en un caso de uso de agentes.

2.7.1.3. DOCUMENTACION PARA CADA CASO DE USO

Una vez que surge un panorama general de la funcionalidad que proporcionan los actores y los comportamientos generales de los agentes, es hora de documentar los casos de uso. Para la especificación del agente de orientación de casos de uso la siguiente

estructura ha resultado la captura de manera adecuada los comportamientos requeridos para los proyectos en los que se ha utilizado el análisis de casos de uso:

- **Usar el nombre de caso –** El nombre del caso de uso
- **Texto descriptivo –** Un párrafo de una o dos descripciones de lo que el caso de uso implica, que describa el comportamiento de los agentes en el contexto del entorno del agente (es decir la interacción con otros objetos) y la interacción con otros agentes y actores.
- **Agentes –** Una lista de los agentes implicados en este caso de uso.
- **Actores –** Una lista de los actores involucrados en este caso de uso.
- **Asociaciones de casos de uso –** una lista de otros casos de uso relacionados con este caso de uso.
- **Medio ambiente –** Una descripción del entorno y los objetos en el medio ambiente que el agente interactúa con este caso de uso. Un diagrama de clases UML a menudo es útil en esta sección para mostrar como el agente se adapta al medio ambiente.
- **Pre-Requisitos –** Un alista de todas las condiciones que debe tener para ser verdad antes de que el agente pueda iniciar este caso de uso.
- **Post-Condiciones –** Una lista de todas las condiciones que debe tener para ser verdad después de que el agente ha completado un comportamiento definido por este caso de uso.
- **Flujo de Eventos –** Una lista numerada de actividades o eventos que inicia el agente, definiendo el comportamiento del agente en este caso de uso. Esta sección también debe incluir un diagrama de actividades UML que represente el flujo de los acontecimientos.

2.7.2. DIAGRAMA DE CLASES DE AGENTES

Los diagramas de clases UML se puede utilizar en el marco de desarrollo de programación orientada a agentes. Se denota una cierta clase de agente, la segunda clase

de algún agente para satisfacer roles específicos y funciones ejecutadas por los casos de agentes. De acuerdo con la declaración rendida por encima de lo que se tiene que especificar las clases de agentes.

2.8. METODOLOGIA ORIENTADA A AGENTES MaSE

MaSe ()MutiagentSystemEngineering es una metodología complete basada en el ciclo de vida clásico del software con un entorno propio de desarrollo para analizar diseñar y construir sistemas multivalentes heterogéneos.

En esta metodología los agentes no son considerados entes autónomos proactivos y sociales sino simples procesos que se comunican para conseguir el objetivo global del sistema. Los agentes son vistos únicamente como una abstracción conveniente que podría o no tener inteligencia. [Gallego2004]

Según Sycara [Sycara1998], las seis batallas principales en el camino de los sistemas multiagente son:

- Descomposición de los problemas y asignación de tareas a agentes individuales.
- Coordinación del control de los agentes y las comunicaciones.
- Hacer que múltiples agentes actúen de una forma coherente.
- Razonamiento acerca de otros agentes y el estado de coordinación.
- Resolver problemas de agentes con objetivos conflictivos.
- Ingeniar sistemas multiagente prácticos.

El objetivo de MaSE es resolver el sexto punto de Sycara [Sycara1998] y proporcionar un marco de trabajo para la resolución de los cinco primeros puntos. Para esto MaSE utiliza una serie de modelos gráficos que describen el sistema y sus interfaces interagenteasi como el funcionamiento interno de los agentes de forma independiente a la plataforma.

MaSE se divide fundamentalmente en dos fases: análisis y diseño. A su vez la primera fase se subdivide en 3 pasos: captura de objetivos, aplicación de casos de uso y refinamiento de roles. La fase de diseño también esta subdividida aunque en 4 pasoso: creación de las clases de agentes, construcción de las conversaciones, ensamblaje de las clases y diseño del sistema.

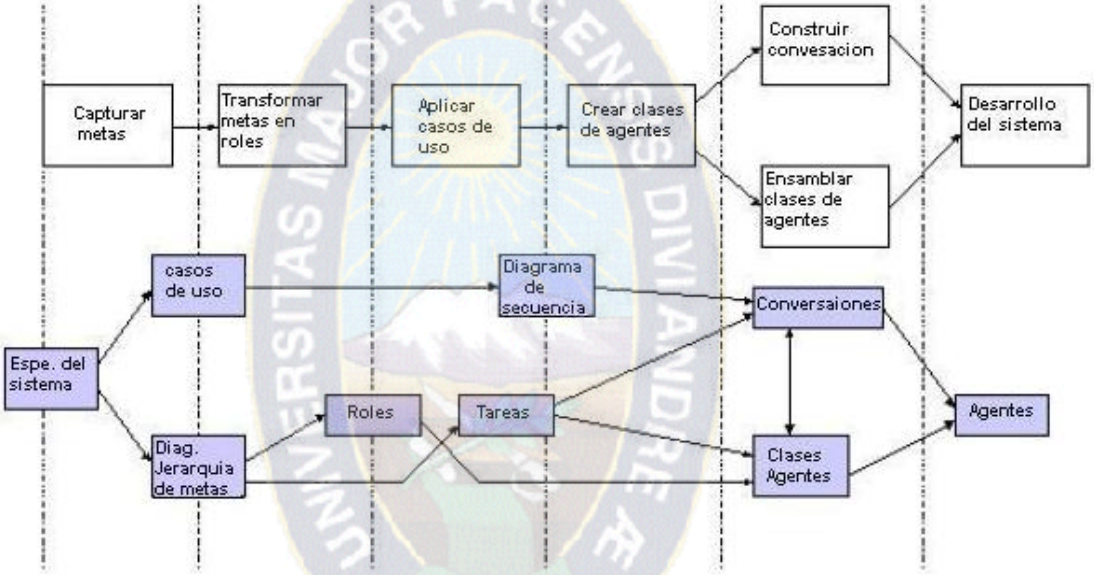


Figura 2.8 Proceso de desarrollo en MaSE

Fuente: [Gallego2004]

2.8.1. CAPTURA DE OBJETIVOS

En esta fase el analista debe cumplir 2 tareas: identificar y estructurar los objetivos. Primero a partir de los documentos de requerimientos se extraen los objetivos principales del sistema sin tener en cuenta las actividades o desarrollos que llevan hasta estos. Esto se hace así porque los objetivos son lo que menos tiende a cambiar en el tiempo. Después estos objetivos son analizados y estructurados según su importancia en un diagrama de jerarquía de objetivos. Los subobjetivos son necesarios para la consecución de los objetivos principales.

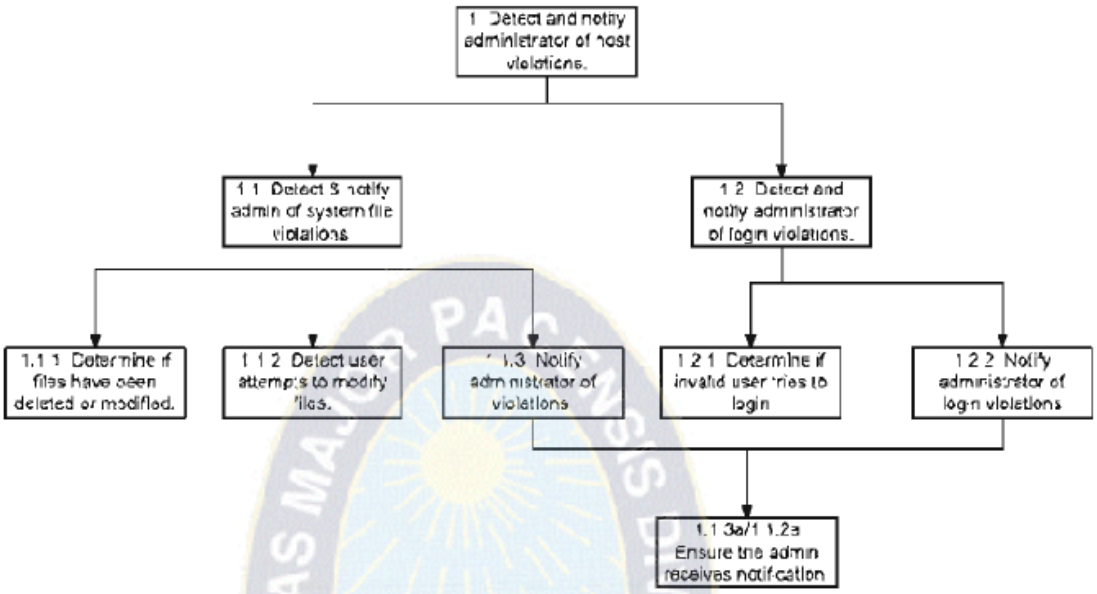


Figura 2.9 Diagrama de jerarquía de objetivos en MaSE

Fuente: [Gallego2004]

Es importante notar que todos los objetivos son siempre de nivel de sistema. En algunos casos el analista puede asociar un rol a cada objetivo.

2.8.2. APLICACIÓN DE CASOS DE USO

Este paso es crucial para convertir objetivos en roles y tareas asociadas. El analista dibuja casos de uso para explicar el comportamiento deseado del sistema. Después estructura los casos de uso en diagramas de secuencia mostrando la secuencia de eventos entre roles y como resultado define la mínima comunicación necesaria entre ellos. [Gallego2004]

2.8.3. REFINAMIENTO DE ROLES

El tercer paso en MaSE es para asegurar que hemos identificado todos los roles necesarios y desarrollar las tareas que definen el comportamiento de los mismos y los patrones de comunicación.

Se asegura que todos los objetivos son tenidos en cuenta asignando un rol a cada uno siendo desempeñado este último por lo menos un agente en el diseño final. Pese a que generalmente cada objetivo es mapeado a un rol individual existen situaciones en las que conviene asignar más de un objetivo a un mismo rol por motivos de eficiencia. Este tipo de decisiones son basadas en conceptos clásicos de ingeniería del software como la cohesión funcional, comunicacional, procedural o temporal. También podría se por distribución de recursos o pro cuestiones especiales de las interfaces.

Finamente los roles son captados en el modelo de roles. [Gallego2004]

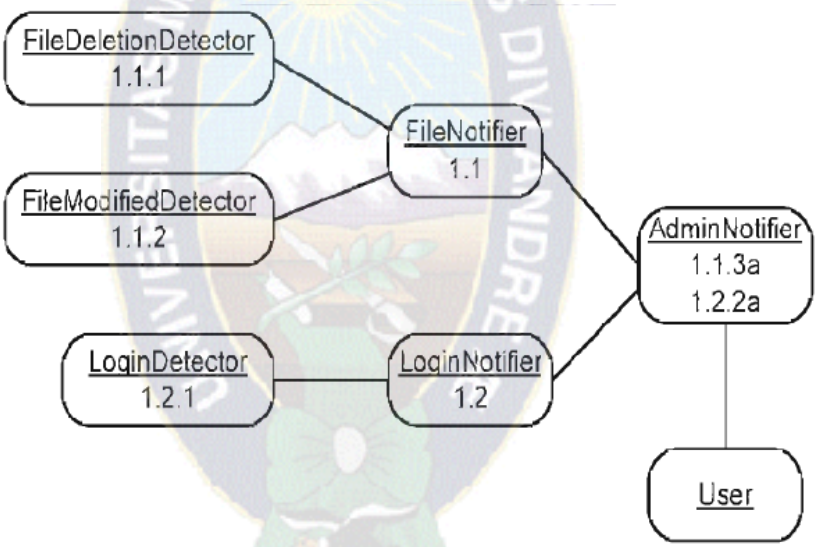


Figura 2.10 Modelo de roles en MaSE

Fuente: [Gallego2004]

Una vez obtenidos los roles la parte más difícil de MaSE consiste en convertirlos en clases de agentes con su comportamiento interno y sus conversaciones entre ellos. Para conseguir esto necesitamos definir tareas de alto nivel que puedan ser transformadas en funcionalidades específicas de agentes. Estas funcionalidades nos ayudan a definir los componentes internos de los agentes y los detalles de las conversaciones en que participan. Todo esto queda reflejado en el modelo de roles detalladoen la siguiente figura.

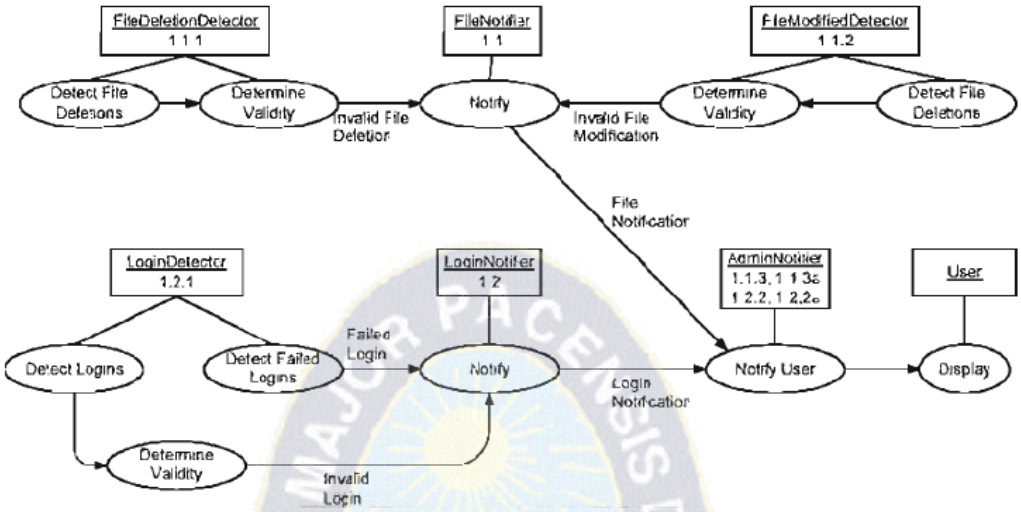


Figura 2.11 Modelo de roles detallado en MaSE

Fuente: [Gallego2004]

2.8.4. CREACION DE LAS CLASES DE AGENTES

Las clases de agentes son identificadas a partir de los roles y descritas en el diagrama de clases de agentes como muestra la (Figura 2.12). De nuevo lo más corriente es establecer una correspondencia uno a uno varios roles en una clase o crear varias clases de un mismo rol. Una vez creado el diagrama de clases la organización del sistema queda definida. En este punto es bueno fusionar roles que tengan en común un gran volumen de tráfico de mensajes para optimizar el sistema.

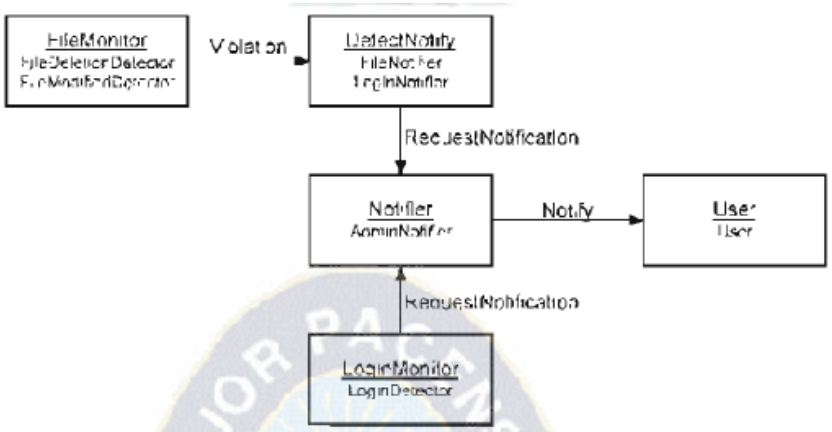


Figura 2.12 Diagrama de jerarquía de objetivos en MaSE

Fuente: [Gallego2004]

2.8.5. CONSTRUCCION DE CONVERSACIONES

Este paso se encuentra íntimamente ligado con el ensamblaje de agentes. Una conversación MaSE define un protocolo de coordinación entre dos agentes. Específicamente una conversación consiste en dos diagramas de clases de comunicación uno para el emisor y otro para el receptor. Un diagrama de clases de comunicación es un par de maquinas de estados finitos que definen una conversación entre dos clases agente participantes. [Gallego2004]

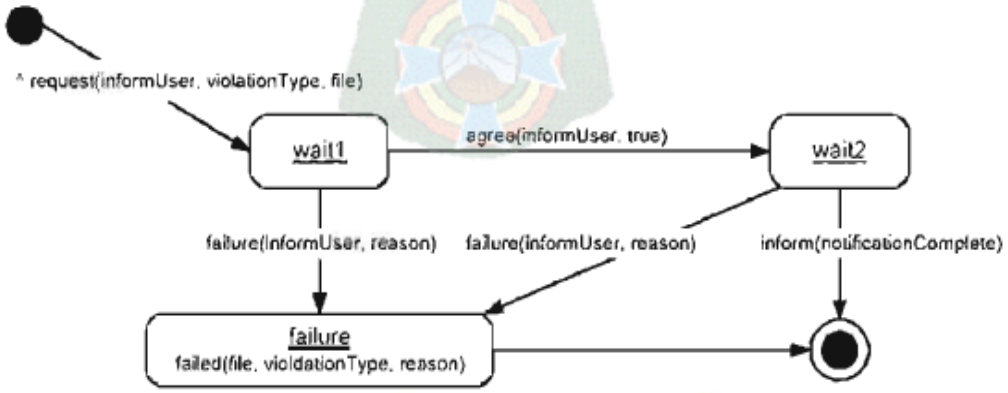


Figura 2.13 Diagrama de clases de comunicación en MaSE

Fuente: [Gallego2004]

2.8.6. ENSAMBLAJE DE AGENTES

En este paso se crea el interior de los agentes. Este proceso se simplifica usando un lenguaje de modelado arquitectónico que combina la naturaleza abstracta de los lenguajes tradicionales de descripción arquitectónica con el lenguaje de restricción de objetos que permite al diseñador especificar detalles de bajo nivel. [Gallego2004]

2.8.7. DESPLIEGUE FINAL DEL SISTEMA

En esta fase se decide la configuración final del sistema a ser implementado. Hasta ahora en MaSE solo se consideran sistemas estáticos no móviles. En MaSE se define la arquitectura global del sistema mediante diagramas de despliegue para mostrar el número, tipo y localización de los agentes.

2.9. TECNOLOGIA Y HERRAMIENTAS DE DESARROLLO

2.9.1. PATRON MODELO VISTA CONTROLADOR (MVC)

Modelo Vista Controlador (MVC) es un patrón de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario y la lógica del control en tres componentes distintos. El patrón de llamada y retorno MVC, se ve frecuentemente en aplicaciones web, donde la vista es la pagina HTML y el código que provee de datos dinámicos a la pagina. 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.

2.9.1.1. DESCRIPCION DEL PATRON

- **Modelo:** Esta es la representación especifica de la información con la cual es sistema opera. En resumen el modelo se limita a lo relativo de la vista y su controlador facilitando las presentaciones visuales complejas. El sistema también puede operar con más datos no relativos a la presentación, haciendo uso integrado de otras lógicas de negocio y de datos afines con el sistema modelado.

- **Vista:** este presenta el modelo en un formato adecuado para interactuar, usualmente la interfaz de usuario.
- **Controlador:** Este responde a eventos, usualmente acciones del usuario e invoca peticiones al modelo y probablemente a la vista.

Muchos de los sistemas informáticos utilizan un Sistema de Gestión de Base de Datos para manipular los datos: en líneas generales del MVC corresponde al modelo.

La unión entre la capa de presentación y la capa de negocio conocido en el paradigma de la Programación por capas representaría la integración entre **Vista** y su correspondiente **Controlador**de eventos y acceso a datos, el patrón MVC no pretende discriminar entre la capa de negocio y la capa de presentación pero si pretende separar la capa visual grafica de su correspondiente programación y acceso a datos algo que mejora el desarrollo y mantenimiento de la Vista y el Controlador en paralelo ya que ambos cumplen ciclos de vida muy distintos entre sí.

Aunque se puede encontrar diferentes implementaciones del patrón MVC el flujo que sigue el control generalmente es el siguiente:

El usuario interactúa con la interfaz de usuario de alguna forma (por ejemplo el usuario pulsa un botón, enlace, etc.)

El controlador accede al modelo actualizándolo posiblemente modificándolo de forma adecuada a la acción solicitada por el usuario (por ejemplo el controlador actualiza el carro de la compra del usuario). Los controladores complejos están a menudo estructurados usando un patrón de comando que encapsula las acciones y simplifica su extensión.

El controlador delega a los objetos de la vista la tarea de desplegar la interfaz de usuario. La vista obtiene sus datos del modelo para generar la interfaz apropiada para el usuario donde se reflejan los cambios en el modelo (por ejemplo produce un listado del contenido del carro de la compra). El modelo no debe tener conocimiento directo sobre la vista. Sin embargo se podría utilizar el patrón Observador para proveer cierta dirección entre el modelo y la vista, permitiendo al modelo notificar a los interesados de cualquier cambio. Un objeto vista puede registrarse con el modelo y esperar a los cambios, pero aun así el modelo en si mismo sigue sin saber nada de la vista. El controlador no pasa objetos de dominio (el modelo) a la vista aunque puede dar la orden a la vista para que se actualice.

La interfaz de usuario espera nuevas interacciones del usuario comenzando el ciclo nuevamente.

2.9.2. LENGUAJE DE PROGRAMACION PHP

PHP es un lenguaje de programación de uso general de código del lado del servidor originalmente diseñado para el desarrollo web de contenido dinámico. Fue uno de los primeros lenguajes de programación del lado del servidor que se podían incorporar directamente en el documento HTML en lugar de llamar a un archivo externo que procese los datos. El código es interpretado por un servidor web con un módulo de procesador de PHP que genera la página Web resultante. PHP ha evolucionado por lo que ahora incluye también una interfaz de línea de comandos que puede ser usada en aplicaciones gráficas independientes. PHP puede ser usado en la mayoría de los servidores web al igual que en casi todos los sistemas operativos y plataformas sin ningún costo.

2.9.3. CARACTERISTICAS

- Orientado al desarrollo de aplicaciones web dinámicas con acceso a información almacenada en una base de datos.
- Es considerado un lenguaje fácil de aprender, ya que en su desarrollo se simplificaron distintas especificaciones, como es el caso de la definición de las variables primitivas, ejemplo que se hace evidente en el uso de phparrays.
- El código fuente escrito en PHP es invisible al navegador web y al cliente, ya que es el servidor el que se encarga de ejecutar el código y enviar su resultado HTML al navegador. Esto hace que la programación en PHP sea segura y confiable.
- Capacidad de conexión con la mayoría de los motores de base de datos que se utilizan en la actualidad, destaca su conectividad con MySQL y PostgreSQL.
- Capacidad de expandir su potencial utilizando módulos (llamados ext's o extensiones).
- Es libre, por lo que se presenta como una alternativa de fácil acceso para todos.
- Permite aplicar técnicas de programación orientada a objetos. Incluso aplicaciones como Zendframework, empresa que desarrolla PHP, están totalmente desarrolladas mediante esta metodología.
- No requiere definición de tipos de variables aunque sus variables se pueden evaluar también por el tipo que estén manejando en tiempo de ejecución.
- Si bien PHP no obliga a quien lo usa a seguir una determinada metodología a la hora de programar, aún haciéndolo, el programador puede aplicar en su trabajo cualquier técnica de programación o de desarrollo que le permita escribir código ordenado, estructurado y manejable. Un ejemplo de esto son los desarrollos que en PHP se han hecho del patrón de diseño Modelo Vista Controlador (MVC), que permiten separar el tratamiento y acceso a los datos, la lógica de control y la interfaz de usuario en tres componentes independientes.
- Debido a su flexibilidad ha tenido una gran acogida como lenguaje base para las aplicaciones WEB de manejo de contenido, y es su uso principal.

2.9.4. SISTEMA GESTOR DE BASE DE DATOS MYSQL

El sistema de base de datos operacional MySQL es hoy en día uno de los más importantes en lo que hace al diseño y programación de base de datos de tipo relacional. Cuenta con millones de aplicaciones y aparece en el mundo informático como una de las más utilizadas por usuarios del medio. El programa MySQL se usa como servidor a través del cual pueden conectarse múltiples usuarios y utilizarlo al mismo tiempo.

Una de las características más interesantes de MySQL es que permite recurrir a bases de datos multiusuario a través de la web y en diferentes lenguajes de programación que se adaptan a diferentes necesidades y requerimientos. Por otro lado, MySQL es conocida por desarrollar alta velocidad en la búsqueda de datos e información, a diferencia de sistemas anteriores.

2.10. METRICAS DE CALIDAD ISO/IEC 9126

ISO 9126 es un estándar internacional para la evaluación del Software. Esta supervisado por el proyecto SQuaRE ISO 25000:2005, el cual sigue los mismos conceptos.

El estándar está dividido en cuatro partes las cuales dirigen respectivamente lo siguiente: modelo de calidad, métricas externas, métricas internas y calidad en las métricas de uso.

El modelo de calidad establecido en la primera parte del estándar ISO 9126-1, clasifica la calidad del software en un conjunto estructurado de características y sub características.

2.10.1. CARACTERISTICAS

Funcionalidad: Un conjunto de atributos que se relacionan con la existencia de un conjunto de funciones y sus propiedades especificas. Las funciones son aquellas que satisfacen las necesidades implícitas o explicitas.

- Idoneidad
- Exactitud
- Interoperabilidad
- Seguridad
- Cumplimiento de normas

Fiabilidad: Un conjunto de atributos relacionados con la capacidad del software de mantener su nivel de prestación bajo condiciones establecidas durante un periodo establecido.

- Madurez
- Recuperación
- Tolerancia a fallos

Usabilidad: Un conjunto de atributos relacionados con el esfuerzo necesario para su uso y en la valoración individual de tal uso por un establecido o implicado conjunto de usuarios.

- Aprendizaje
- Compresión
- Operatividad
- Atractividad

Eficiencia: Conjunto de atributos con la relación entre el nivel de desempeño del software y la cantidad de recursos necesitados bajo condiciones establecidas.

- Comportamiento en el tiempo
- Comportamiento de recursos

Mantenibilidad: Conjunto de atributos relacionados con la facilidad de extender modificar o corregir errores en un sistema software.

- Estabilidad
- Facilidad de análisis
- Facilidad de cambio
- Facilidad de pruebas

Portabilidad: Conjunto de atributos relacionados con la capacidad de un sistema software para ser transferido desde una plataforma a otra.

- Capacidad de instalación
- Capacidad de reemplazamiento
- Adaptabilidad
- Co-existencia

La sub característica Conformidad no está arriba ya se aplica a todas las características. Ejemplos son conformidad a la legislación referente a usabilidad y fiabilidad.

Cada sub característica (como adaptabilidad) está dividida en atributos. Un atributo es una entidad la cual puede ser verificado o medida en el producto software.

Los atributos no están definidos en el estándar ya que varían entre diferentes productos de software.

2.10.2. METODO PARA MEDIR EL TAMAÑO FUNCIONAL Y EVALUAR LA CALIDAD DE SITIOS WEB

La calidad del software para la Web es una preocupación actual de la comunidad científica y empresarial. En este artículo se discuten los principales problemas encontrados en el desarrollo de aplicaciones Web y se presenta un método con el objetivo de medir el tamaño funcional y evaluar la calidad de sitios Web. El método propuesto integra un modelo navegación elaborado en el proceso de desarrollo de estas aplicaciones una métrica que permite medir “puntos de función para la web” y un

modelo de calidad para identificar y cuantificar atributos Web. De esta forma nos centramos en aspectos fundamentales de la calidad de aplicaciones Web: Medición y evaluación del producto software y control del proceso de desarrollo.

2.10.2.1.EVALUACION DE LA CALIDAD

Actualmente se han publicado en la web una serie de guías y criterios que ayudan a mejorar el proceso de diseño y autoría de las aplicaciones Web con relación a aspectos de usabilidad, navegabilidad, accesibilidad entre otros. Esas guías son útiles en la documentación de características y criterios de calidad que deben tenerse en cuenta en un proceso de evaluación pero no constituyen una metodología de evaluación de artefactos Web.

En este paso se ha utilizado la metodología Web QEM (QualityEvaluationMethod) para evaluar la calidad de sitios web. Esta metodología parte de un modelo de calidad que proporciona un enfoque cuantitativo y sistemático para evaluar y comparar productos Web tanto en la fase operativa como en la fase de desarrollo del ciclo de vida de un producto. El principal objetivo de Web QEM es evaluar y determinar el nivel de cumplimiento de los siguientes factores de calidad descritos en el estándar ISO 9126:

- Usabilidad: capacidad del producto software ser entendido, aprendido y usado por los usuarios bajo condiciones especificas.
- Funcionalidad: capacidad del producto software proporcionar funciones que ejecuten las necesidades explicitas e implícitas de los usuarios cuando el software es usado bajo condiciones especificas.
- Confiabilidad capacidad del producto software en mantener un nivel especificado de rendimiento cuando es usado bajo condiciones especificas.
- Eficiencia: representa la relación entre el grado de rendimiento del sitio y la cantidad de recursos (tiempo, espacio, etc.) usados bajo ciertas condiciones.
- Mantenimiento: capacidad del producto software de ser modificado y probado.

- Portabilidad: capacidad del producto software de ser transferido de un ambiente a otro.

2.10.3. COCOMO

El Modelo Constructivo de Costos (o COCOMO), por su acrónimo del inglés (COnstructiveCOstMOdel) es un modelo matemático de base empírica utilizado para estimación de costes de software. Incluye tres sub modelos cada uno ofrece un nivel de detalle y aproximación cada vez mayor a medida que avanza el proceso de desarrollo del software: básico, intermedio y detallado.

Este modelo fue desarrollado pro Barry W. Boehm a finales de los años 70 y comienzos de los 80, exponiéndolo detalladamente es su libro "Software EngineeringEconomics"

2.10.3.1. CARACTERISTICAS

Pertenece a la categoría de modelos de sub estimaciones basados en estimaciones matemáticas. Está orientado a la magnitud del producto final, midiendo el "tamaño" del proyecto en líneas de código principalmente.

2.10.3.2. INCONVENIENTES

- Los resultados no son proporcionales a las tareas de gestión ya que no tiene en cuenta los recursos necesarios para realizarlas.
- Se puede desviar de la realidad si se indica mal el porcentaje de líneas de comentarios en el código fuente.
- Es un tanto subjetivo puesto que está basado en estimaciones y parámetros que pueden ser "vistos" de distinta manera pro distintos analistas que usen el método.
- Se miden los costes del producto de acuerdo a su tamaño y otras características pero no la productividad.
- La medición por líneas de código no es válida para orientación a objetos.

- Utilizar este modelo puede resultar un poco complicado en comparación con otros métodos (que también solo estiman).

2.10.3.3.MODELOS DE ESTIMACION

Las ecuaciones que se utilizan en los tres modelos son:

$$E = a(Kl)^b * m(X), en\ personas - mes$$

$$Tdev = c(E)^d, en\ meses$$

$$P = \frac{E}{Tdev}, en\ personas$$

Donde:

E: es el esfuerzo requerido por el proyecto en personas – mes.

Tdev: es el tiempo requerido por el proyecto en meses.

P: es el número de personas requerido por el proyecto.

a, b, c y d: son constantes con valores definidos en una tabla según cada sub modelo.

Kl: es la cantidad de líneas de código en miles.

m(X): es un multiplicador que depende de 15 atributos.

A la vez cada sub modelo también se divide en modos que representan el tipo de proyecto y puede ser:

- **Modo orgánico:** un pequeño grupo de programadores experimentados desarrollan software en un entorno familiar. El tamaño del software varía desde unos pocos miles de líneas (tamaño pequeño) a unas decenas de miles (medio).
- **Modo semilibre o semiencajado:** corresponde a un esquema intermedio entre el orgánico y el rígido; el grupo de desarrollo puede incluir una mezcla de personas experimentadas y no experimentadas.
- **Modo rígido o empotrado:** el proyecto tiene fuertes restricciones que pueden estar relacionadas con la funcionalidad y/o pueden se técnicas. El problema a resolver es único y es difícil basarse en la experiencia puesto que puede no haberla.

2.10.3.4.EL MODELO DE BASE DE ESTIMACION

Comenzaremos con el modelo "base" de COCOMO II, el cual corresponde al esfuerzo de desarrollo estimado una vez que se ha fijado la arquitectura del sistema (estimación del proceso post - arquitectura). Después de presentar tal modelo base veremos cómo se ajusta el modelo para:

- Estimaciones más tempranas correspondiente al diseño temprano (Pre - Arquitectura);
- Mantenimiento;
- Estimación de número de defectos esperados.

Paso 1: Estimación de puntos de función.

Debemos considerar que cada pantalla o reporte a generar equivale a 8 puntos de función (de ahora en adelante pf).

Paso 2: Modelo más detallado.

El modelo más detallado de N. Callaos (y consonó con la literatura sobre puntos de función técnicamente "featurepoints") es:

- Cada pantalla de consulta vale 4 pf.
- Cada pantalla que carga datos vale 5 pf.
- Cada reporte vale 5 pf.
- Cada tabla de una base de datos vale 10 pf.
- Cada interfaz con otro sistema vale 6 pf.

La simplificación de N. Callaos consiste en considerar que toda pantalla o reporte vale 5 pf y que solo el 60% de los puntos de función provienen de pantalla – el resto vienen den tablas o archivos lógicos.

Paso 3: aplicaciones de las formulas básicas de esfuerzo tiempo calendario y personal requerido.

La formula básica de esfuerzo (PM, person – months)

$$PM = 2.94 * Tamaño^{E} * \prod EMi$$

Donde:

Tamaño: se mide en KSLOC.

EMi: corresponde a los factores de ajuste de costo.

E: es el exponente no lineal calculado según:

$$E = 0.91 + 0.01 * \sum SFi$$

Donde: SFi son parámetros de costo (cost-drivers).

2.10.3.5. VALOR ACTUAL NETO

También conocido valor actualizado neto (en ingles Net PresentValue), cuyo acrónimo es VAN (en inglés NPV), es un procedimiento que permite calcular el valor presente de un determinado número de flujos de caja futuros, originados por una inversión. La metodología consiste en descontar al momento actual (es decir actualizar mediante un tasa) todos los flujos de caja futuros del proyecto. A este valor se le resta la inversión inicial de tal modo que el valor obtenido es el valor actual neto del proyecto.

El método de valor presente es uno de los criterios económicos más ampliamente utilizados en la evaluación de proyectos de inversión. Consiste en determinar la equivalencia en el tiempo 0 de los flujos de efectivo futuros que genera un proyecto y comparar esta equivalencia con el desembolso inicial. Cuando dicha equivalencia es mayor que el desembolso inicial entonces es recomendable que el proyecto sea aceptado.

La fórmula que nos permite calcular el Valor Actual Neto es:

$$VAN = \sum_{t=1}^{n} \frac{V_t}{(1+k)^t} - I_0$$

Vt: representa los flujos de caja en cada periodo t.
Io: es el valor del desembolso inicial de la inversión.
N: es el número de periodos considerado.

El tipo de interés es *k*si el proyecto no tiene riesgo, se tomara como referencia el tipo de la renta fija de tal manera que con el VAN se estimara se la inversión es mejor que invertir en algo seguro sin riesgo especifico. En otros casos se utilizara el coste de oportunidad.

Cuando el VAN toma en valor igual a 0, *k* pasa a llamarse TIR (tasa interna de retorno). La TIR es la rentabilidad que nos está proporcionando el proyecto.

CAPITULO III
MARCO APLICATIVO

3.1. INTRODUCCION

En este capítulo se efectuara el análisis y diseño correspondiente al sistema puesto que SCRUM es una metodología de desarrollo muy simple que requiere trabajo duro porque la gestión no se basa en el seguimiento de un plan sino en la adaptación continua a las circunstancias de la evolución del proyecto.

Al comenzar cada iteración (“sprint”) se determina que partes se van a construir tomando como criterios la prioridad para el negocio y la cantidad de trabajo que se podrá abordar durante la iteración.

Para el desarrollo del proyecto se utilizara la metodología Ágil SCRUM que utiliza un modelo de proceso incremental y se la complemento con la metodología UWE para las etapas de desarrollo de cada iteración. En la siguiente figura se puede apreciar gráficamente el modelo de procesos que utilizo en el presente Proyecto de Grado.

También mencionar que se utilizaran los agentes inteligentes de recuperación de información o agentes de base de datos. Los agentes serán de utilidad al momento de seleccionar datos de predios más viables o predios con mayor prioridad a esta acción la nombraremos despliegue de una lista de predios viables.

El modelado de Agentes (AUML) nos ayudara a formar el correcto reconocimiento del área en el que intervendrán los agentes y el proceso que seguirán. Para esto se utilizara diagramas que son: Diagramas de Casos de Uso, para la descripción del área o actividad donde intervendrán los agentes y diagramas de clases para describir los objetos con los que se relacionara el agente.

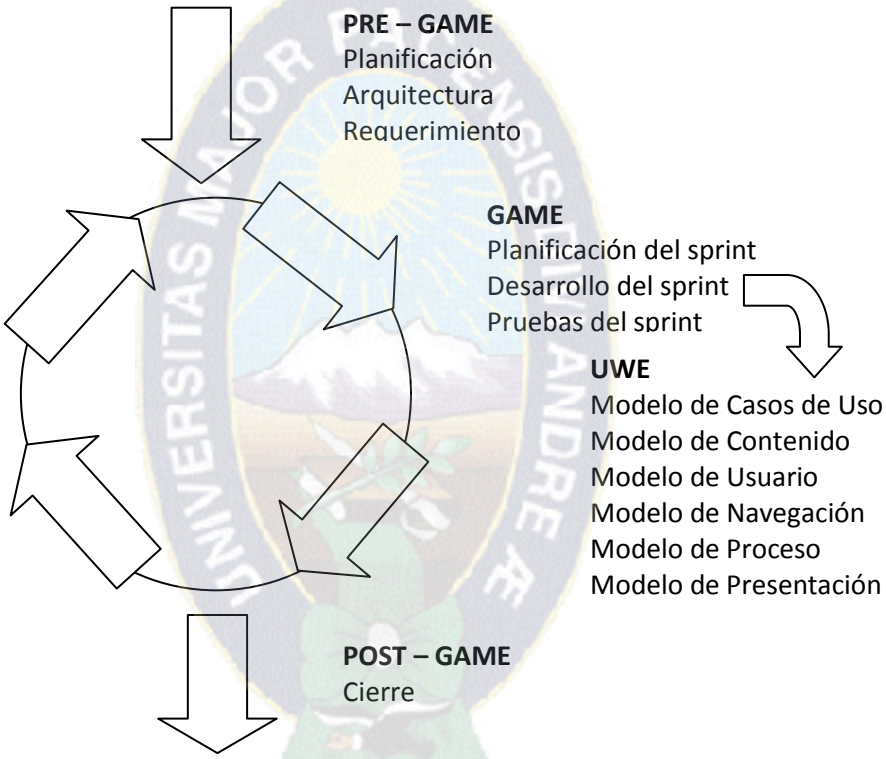


Figura 3.1 Etapas de Sprint

Fuente: [Elaboración propia]

3.2. PRE – GAME

3.2.1. RECOPILACION DE REQUERIMIENTOS

A continuación se presenta el Backlog del producto (Pila del producto), que contiene las historias de usuario. Mismas que fueron obtenidas de las reuniones realizadas con el personal de la Jefatura Regional Valles.

ID	Nombre	Imp.	Usuario
H1	Entrar al sistema	850	Administrador,Supervisor, Jurídico, Técnico, Sist
H2	Ver listado de comisiones	800	Administrador,Supervisor
H3	Añadir nueva comisión	750	Administrador
H4	Añadir nuevo predio a comisión	700	Administrador
H5	Editar predio	550	Administrador
H6	Eliminar predio de comisión	500	Administrador
H7	Ver detalle de comisión	650	Administrador,Supervisor
H8	Registrar recepción de predio	620	Administrador
H9	Ver detalle de predio	600	Administrador,Supervisor
H10	Buscar por nombre de predio	450	Administrador,Supervisor
H11	Ver listado de predios	520	Administrador,Supervisor
H12	Búsqueda avanzada de predios	100	Administrador,Supervisor
H13	Ver listado de personal	400	Administrador,Supervisor
H14	Añadir nuevo personal	350	Administrador
H15	Editar personal	300	Administrador
H16	Asignar predio a personal	610	Administrador
H17	Ver detalle de personal	340	Administrador,Supervisor
H18	Ver predios asignados a personal	330	Administrador,Supervisor
H19	Bloquear personal	110	Administrador
H20	Activar personal	110	Administrador
H21	Buscar por nota de recepción	150	Administrador,Supervisor
H22	Ver listado de comisiones realizadas	280	Jurídico, Técnico, Sist
H23	Ver detalle de comisión	270	Jurídico, Técnico, Sist
H24	Ver predios revisados en comisión	260	Jurídico, Técnico, Sist
H25	Ver detalle de predio revisado	250	Jurídico, Técnico, Sist
H26	Ver listado de predios revisados	250	Jurídico, Técnico, Sist
H27	Buscar por nombre de predio revisado	160	Jurídico, Técnico, Sist
H28	Búsqueda avanzada de predios revisados	100	Jurídico, Técnico, Sist
H29	Ver resumen de predios revisados	120	Jurídico, Técnico, Sist
H30	Ver listado de predios viables	230	Jurídico, Técnico, Sist
H31	Listar predios revisados por comisión	240	Jurídico, Técnico, Sist
H32	Registrar fecha de réplica de predio	250	Sist
H33	Actualizar estado de predio	250	Sist
H34	Registrar control topológico de predio	250	Técnico

Tabla 3.1 Cuadro de Historias de usuario

Fuente: [Elaboración propia]

3.2.2. DEFINICION DEL CRONOGRAMA DE TRABAJO

El cronograma de trabajo se definió en base al ciclo de vida de la metodología SCRUM, en el cual se identifican 3 etapas principales que son: el pre – game, game y post – game. La descripción detallada del cronograma de trabajo para el proyecto se la puede observar en el diagrama Gantt (Ver Anexo A).

3.2.3. ANALISIS DE RIESGO

Un riesgo es la probabilidad de que ocurra algo adverso, existen 3 tipos de riesgos:

- **Riesgo del proyecto:** que afecta a la calendarización o recursos del proyecto.
- **Riesgo del producto:** afectan a la calidad o rendimiento del software que se está desarrollando.
- **Riesgo de negocio:** afectan a la organización que desarrolla o suministra el software

Riesgo	Tipo	Descripción	Probabilidad	Efecto	Estrategia
No se cumplen las fechas establecidas en el cronograma	Proyecto	Es probable que en las fechas descritas del diagrama Gantt no se cumplan al pie la letra	Alta	Tolerable	Realizar un segundo cronograma que sea más flexible
Cambio en los requerimientos del cliente	Proyecto Producto	Riesgo de que haya cambios en los requerimientos de la Unidad de Auditoría Interna	Moderada	Tolerable	Realizar una revisión constante a los requerimientos Programar reuniones con los auditores
No se cumple con los plazos de entrega del producto	Producto	Los plazos de entrega del producto están determinados por el Jefe Regional Valles	Moderada	Serio	Agilizar los procesos de desarrollo del producto Realizar una correcta planificación, considerando el

					tiempo y los alcances del proyecto
No existe la infraestructura necesaria para la implementación del sistema	Proyecto	Es probable que no cuente con infraestructura necesaria para realizar el sistema	Moderada	Tolerable	Solicitar con anticipación o contar con un equipo portátil

Tabla 3.2 Análisis de riesgo

Fuente: [Elaboración propia]

3.3. GAME

Durante esta etapa del proyecto se desarrollaron 4 iteraciones, cada una de ellas corresponde a un modulo del sistema. A continuación se desglosan las actividades realizadas en cada una de estas etapas.

La estrategia que utilizo para el desarrollo de cada iteración fue construir en un principio los modelos de la metodología UWE y posteriormente implementarlos utilizando como elemento central la base de datos "MySQL". La solución por la que se opto para la persistencia de objetos fue el hacer corresponder cada clase del modelo conceptual con una tabla de la base de datos en donde las filas representan las instancias de los objetos y las columnas a los atributos de la clase. Las clases y sus métodos fueron implementados en lenguaje PHP. Finalmente se desarrollaron las páginas Web en base a los modelos de presentación y de navegación.

3.3.1. PRIMERA ITERACION

Durante la primera iteración se desarrollaron los elementos pertenecientes al modulo de comisiones y predios enviados por las departamentales. Las actividades realizadas durante esta iteración se observa en la siguiente tabla que constituye el backlog del sprint (Pila del sprint).

PILA DEL SPRINT 1			Inicio	Duración
			21-sept	25 días
ID	Nombre	Tareas / Actividades	Tipo	Estado
HT1	Crear y documentar el diseño general de la base de datos	Diseñar el modelo entidad – relación	Análisis/diseño	Terminado
		Crear la base de datos y tablas del sistema general	Análisis/diseño	Terminado
HT2	Crear y documentar el diseño del sprint	Analizar los requerimientos con casos de uso	Análisis/diseño	Terminado
		Diseñar el modelo de contenidos	Análisis/diseño	Terminado
		Diseñar el modelo de usuario	Análisis/diseño	Terminado
		Diseñar el modelo de navegación	Análisis/diseño	Terminado
		Diseñar el modelo de proceso	Análisis/diseño	Terminado
		Diseñar el modelo de presentación	Análisis/diseño	Terminado
H1	Entrar al sistema	Se crea la forma de ingreso al sistema	Diseño	Terminado
		Notificación de error en caso de que el usuario y/o contraseña sean incorrectos o no exista	Diseño	Terminado
		Query para iniciar sesión de cliente al sistema	Codificación	Terminado
		Creación de pruebas para testeo	Pruebas	Terminado
H2	Ver listado de comisiones	Se crea la pantalla que muestra las comisiones por depto.	Diseño	Terminado
		Query para obtener las comisiones por depto.	Codificación	Terminado
		Creación de pruebas y testeo	Pruebas	Terminado
H3	Añadir nueva comisión	Creación de formulario de entrada de comisión	Diseño	Terminado
		Creación de mensaje de notificación de éxito o fracaso	Diseño	Terminado
		Validación de datos de formulario de entrada	Codificación	Terminado
		Query de Inserción de datos en registro de tabla en BD	Codificación	Terminado
		Creación de pruebas y testeo	Pruebas	Terminado
H4	Añadir nuevo predio a comisión	Creación de formulario de entrada de predio a comisión	Diseño	Terminado

		Creación de mensaje de notificación de éxito o fracaso	Diseño	Terminado
		Validación de datos de formulario de entrada	Codificación	Terminado
		Query de Inserción de datos en registro de tabla en BD	Codificación	Terminado
		Creación de pruebas y testeo	Pruebas	Terminado
H7	Ver detalle de comisión	Creación de pantalla que despliegue datos de la comisión y sus predios asociados	Diseño	Terminado
		Query para obtener datos de comisión	Codificación	Terminado
		Query para obtener datos de predios de comisión	Codificación	Terminado
		Creación de pruebas y testeo	Pruebas	Terminado
H8	Registrar recepción de predio	Creación de formulario de recepción de predio	Diseño	Terminado
		Creación de mensaje de notificación de éxito o fracaso	Codificación	Terminado
		Validación de datos de formulario de entrada	Codificación	Terminado
		Query de inserción de datos del registro en la tabla de BD	Codificación	Terminado
		Creación de pruebas y testeo	Pruebas	Terminado
H16	Asignar predio a personal	Creación de formulario de asignación	Diseño	Terminado
		Creación de mensaje de notificación de éxito o fracaso	Diseño	Terminado
		Validación de datos de formulario de entrada	Diseño	Terminado
		Query de inserción de datos del registro en la tabla de BD	Codificación	Terminado
		Creación de pruebas y testeo	Pruebas	Terminado
H9	Ver detalle de predio	Creación de pantalla que despliegue datos de predio y revisores	Diseño	Terminado
		Query para obtener datos de predio	Codificación	Terminado
		Query para obtener datos de revisores del predio	Codificación	Terminado
		Creación de pruebas y testeo	Pruebas	Terminado

H5	Editar predio	Creación de formulario para editarpredio	Diseño	Terminado
		Creación de mensaje de notificación de éxito o fracaso	Diseño	Terminado
		Validación de datos de formulario de edición	Codificación	Terminado
		Query de Actualización de datos en registro de tabla en BD	Codificación	Terminado
		Creación de pruebas y testeo	Pruebas	Terminado

Tabla 3.3 Primera iteración

Fuente: [Elaboración propia]

Las funcionalidades correspondientes al incremento de la iteración son:

- Base de datos independiente
- Paginas de ingreso con control de acceso a los usuarios
- Modulo de comisiones con funcionalidades básicas
- Modulo de predios con funcionalidades básicas

3.3.2. SEGUNDA ITERACION

En la segunda iteración se complementaron funcionalidades al modulo de comisiones y modulo de predios, así también se desarrollo el modulo de personal con funcionalidades básicas.

PILA DEL SPRINT 2			Inicio	Duración
			14-oct	20 días
ID	Nombre	Tareas / Actividades	Tipo	Estado
HT2	Crear y documentar el diseño del sprint	Analizar los requerimientos con casos de uso	Análisis/diseño	Terminado
		Diseñar el modelo de contenidos	Análisis/diseño	Terminado
		Diseñar el modelo de usuario	Análisis/diseño	Terminado
		Diseñar el modelo de navegación	Análisis/diseño	Terminado

		Diseñar el modelo de proceso	Análisis/diseño	Terminado
		Diseñar el modelo de presentación	Análisis/diseño	Terminado
H11	Ver listado de predios	Creación de pantalla que muestre listado de predio por depto.	Diseño	Terminado
		Query para obtener los predios por depto.	Diseño	Terminado
		Creación de pruebas y testeo	Pruebas	Terminado
H6	Eliminar predio de comisión	Creación de pantalla para confirmar la baja de comisión	Diseño	Terminado
		Creación de pantalla emergente para notificar éxito o fracaso de baja de comisión	Diseño	Terminado
		Query de eliminación para baja de comisión en tabla de BD	Codificación	Terminado
		Creación de prueba de baja de comisión	Pruebas	Terminado
H10	Buscar por nombre de predio	Creación de formulario de búsqueda de predio	Diseño	Terminado
		Query de búsqueda de predio con parámetros	Codificación	Terminado
		Creación de pantalla de respuesta de query de búsqueda	Diseño	Terminado
		Creación de pruebas y testeo	Pruebas	Terminado
H13	Ver listado de personal	Creación de pantalla que muestre listado de personal por perfil.	Diseño	Terminado
		Query para obtener lista del personal por perfil.	Diseño	Terminado
		Creación de pruebas y testeo	Pruebas	Terminado
H14	Añadir nuevo personal	Creación de formulario de entrada de personal	Diseño	Terminado
		Creación de mensaje de notificación de éxito o fracaso	Diseño	Terminado
		Validación de datos de formulario de entrada	Codificación	Terminado
		Query de Inserción de datos en registro de tabla en BD	Codificación	Terminado
		Creación de pruebas y testeo	Pruebas	Terminado
H17	Ver detalle de personal	Creación de pantalla que despliegue datos de personal	Diseño	Terminado

		Query para obtener datos de personal	Codificación	Terminado
		Creación de pruebas y testeo	Pruebas	Terminado
H18	Ver predios asignados a personal	Creación de pantalla que muestre listado de predios asignados al personal	Diseño	Terminado
		Query para obtener lista del predios asignados a el personal	Diseño	Terminado
		Creación de pruebas y testeo	Pruebas	Terminado
H15	Editar personal	Creación de formulario para editarpersonal	Diseño	Terminado
		Creación de mensaje de notificación de éxito o fracaso	Diseño	Terminado
		Validación de datos de formulario de edición	Codificación	Terminado
		Query de Actualización de datos en registro de tabla en BD	Codificación	Terminado
		Creación de pruebas y testeo	Pruebas	Terminado
H22	Ver listado de comisiones realizadas	Creación de pantalla que muestre listado de comisiones realizadas por personal (Jurídico, Técnico o Sist)	Diseño	Terminado
		Query de autentificación de personal en la BD	Codificación	Terminado
		Query para obtener lista de comisiones realizadas por personal especifico	Codificación	Terminado
		Creación de pruebas y testeo	Pruebas	Terminado
H23	Ver detalle de comisión	Creación de pantalla que despliegue datos de personal	Diseño	Terminado
		Query de autentificación de personal en la BD	Codificación	Terminado
		Query para obtener datos de comisión	Codificación	Terminado
		Creación de pruebas y testeo	Pruebas	Terminado

Tabla 3.4Segunda iteración

Fuente: [Elaboración propia]

Las funcionalidades correspondientes al incremento de la iteración son:

- Modulo de comisiones con funcionalidades complementarias implementadas
- Modulo de predios con funcionalidades complementarias implementadas
- Modulo de personal con funcionalidad parcialmente desarrollada

3.3.3. TERCERA ITERACION

En la tercera iteración se complementaron funcionalidades al modulo de comisiones y modulo de predios, tomando en cuenta que el personal Jurídico, Técnico o Sist solo pueden visualizar los predios y comisiones con las que están vinculados. También se desarrollara el modulo en el cual el agente nos devolverá un listado con los predios más viables al personal.

PILA DEL SPRINT 3			Inicio	Duración
			02-nov	15 días
ID	Nombre	Tareas / Actividades	Tipo	Estado
HT2	Crear y documentar el diseño del sprint	Analizar los requerimientos con casos de uso	Análisis/diseño	Terminado
		Diseñar el modelo de contenidos	Análisis/diseño	Terminado
		Diseñar el modelo de usuario	Análisis/diseño	Terminado
		Diseñar el modelo de navegación	Análisis/diseño	Terminado
		Diseñar el modelo de proceso	Análisis/diseño	Terminado
		Diseñar el modelo de presentación	Análisis/diseño	Terminado
HT3	Crear documentar el diseño del agente	Diseñar modelos de casos de uso y secuencia despliegue de la metodología MaSE	Análisis/diseño	Terminado
H24	Ver predios revisados en comisión	Creación de pantalla que muestre listado de predios revisados por personal (Jurídico, Técnico o Sist)	Diseño	Terminado
		Query de autentificación de personal en la BD	Codificación	Terminado
		Query para obtener lista de predios revisados por personal especifico autentificado	Codificación	Terminado

		Creación de pruebas y testeo	Pruebas	Terminado
H25	Ver detalle de predio revisado	Creación de pantalla que despliegue datos de predio revisado por personal (Jurídico, Técnico o Sist)	Diseño	Terminado
		Query de autentificación de personal en la BD	Codificación	Terminado
		Query para obtener datos de predio revisado por personal autentificado	Codificación	Terminado
		Creación de pruebas y testeo	Pruebas	Terminado
H26	Ver listado de predios revisados	Creación de pantalla que despliegue listado de predios revisados por personal (Jurídico, Técnico o Sist) clasificado por depto.	Diseño	Terminado
		Query de autentificación de personal en la BD	Codificación	Terminado
		Query para obtener lista de predios revisados por personal especifico autentificado	Codificación	Terminado
		Creación de pruebas y testeo	Pruebas	Terminado
H32	Registrar fecha de réplica de predio	Creación de interfaz que registre un predio como replicado	Diseño	Terminado
		Query de autentificación de personal Sist en la BD	Codificación	Terminado
		Query para Actualizar un predio en la BD como replicado	Codificación	Terminado
		Creación de pruebas y testeo	Pruebas	Terminado
H33	Actualizar estado de predio	Creación de interfaz que registre el respaldo de cambio de estado de un predio	Diseño	Terminado
		Query de autentificación de personal Sist en la BD	Codificación	Terminado
		Query de inserción de cambio de estado de un predio en la BD	Codificación	Terminado
		Query de inserción de respaldo de cambio de estado en la BD	Codificación	Terminado
		Creación de pruebas y testeo	Pruebas	Terminado
H34	Registrar control topológico de predio	Creación de formulario que registre datos del control topológico de un predio	Diseño	Terminado

		Query de autentificación de personal Técnico en la BD	Codificación	Terminado
		Query para actualizar datos de control topológico de un predio en la BD	Codificación	Terminado
		Creación de pruebas y testeo	Pruebas	Terminado
H31	Listar predios revisados por comisión	Creación de pantalla que despliegue listado de predios revisados por comisión clasificada por depto.	Diseño	Terminado
		Query de autentificación de personal (Jurídico, Técnico o Sist) en la BD	Codificación	Terminado
		Query para obtener lista de predios revisados por comisión con parámetros	Codificación	Terminado
		Creación de pruebas y testeo	Pruebas	Terminado
H30	Ver listado de predios viables	Creación de pantalla que despliegue listado de predios revisados viables por depto.	Diseño	Terminado
		Script del agente inteligente con parámetros de entrada	Codificación	Terminado
		Creación de pruebas y testeo	Pruebas	Terminado

Tabla 3.5 Tercera iteración

Fuente: [Elaboración propia]

Las funcionalidades correspondientes al incremento de la iteración son:

- Se desarrollaron los módulos de comisión, predio para el personal Jurídico, Técnico y Sist
- Modulo de obtención de predios viables aplicando el script del agente inteligente desarrollado
- Modulo de comisiones con funcionalidades complementarias implementadas
- Modulo de predios con funcionalidades complementarias implementadas
- Modulo de personal con funcionalidad parcialmente desarrollada

3.3.4. CUARTA ITERACION

En la cuarta iteración se complementaron funcionalidades de los módulos de comisiones, predios, personal para el usuarios (personal Jurídico, Técnico, Sist, Administrador y Supervisor) de la jefatura regional Valles.

PILA DEL SPRINT 4			Inicio	Duración
			14-nov	8 días
ID	Nombre	Tareas / Actividades	Tipo	Estado
HT2	Crear y documentar el diseño del sprint	Analizar los requerimientos con casos de uso	Análisis/diseño	Terminado
		Diseñar el modelo de contenidos	Análisis/diseño	Terminado
		Diseñar el modelo de usuario	Análisis/diseño	Terminado
		Diseñar el modelo de navegación	Análisis/diseño	Terminado
		Diseñar el modelo de proceso	Análisis/diseño	Terminado
		Diseñar el modelo de presentación	Análisis/diseño	Terminado
H27	Buscar por nombre de predio revisado	Creación de formulario de búsqueda de predio	Diseño	Terminado
		Query de búsqueda de predio con parámetros	Codificación	Terminado
		Creación de pantalla de respuesta de query de búsqueda	Diseño	Terminado
		Creación de pruebas y testeo	Pruebas	Terminado
H21	Buscar por nota de recepción	Creación de formulario de búsqueda de nota de recepción	Diseño	Terminado
		Query de búsqueda de notas con parámetros	Codificación	Terminado
		Creación de pantalla de respuesta de query de búsqueda	Diseño	Terminado
		Creación de pruebas y testeo	Pruebas	Terminado
H29	Ver resumen de predios revisados	Creación de la pantalla de resumen de predios	Diseño	Terminado
		Query de obtención de totales con parámetros	Codificación	Terminado
		Creación de pruebas y testeo	Pruebas	Terminado

H28	Búsqueda avanzada de predios	Creación de formulario de búsqueda de predio	Diseño	Terminado
		Query de búsqueda de predio con parámetros	Codificación	Terminado
		Creación de pantalla de respuesta de query de búsqueda	Diseño	Terminado
		Creación de pruebas y testeo	Pruebas	Terminado
	Activar / bloquear personal	Creación de formulario	Diseño	Terminado
		Query de actualización tabla personal	Codificación	Terminado
		Creación de pruebas y testeo	Pruebas	Terminado

Tabla 3.6 Cuarta iteración

Fuente: [Elaboración propia]

En la cuarta iteración se desarrollaron las siguientes funcionalidades para el sistema:

- Modulo de búsqueda básica y avanzada de predios
- Modulo de búsqueda de notas de recepción
- Modulo para activar y bloquear al personal

3.4. GAME

3.4.1. MODELO DE CASOS DE USO

Un modelo de casos de uso es un modelo que representa la manera en que el sistema interactúa con los actores o usuarios.

A continuación se describen las características de los actores identificados en el manejo e implementación del sistema.

ACTORES	DESCRIPCION
Supervisor	Encargado de realizar seguimiento de predios, comisiones y verificar carga de trabajo al personal de la Jefatura Regional Valles
Administrador	Realiza el registro y control de predios, comisiones y personal de la Jefatura Regional Valles. También registra la recepción del predio.
Técnico	Encargado de realizar la aprobación del control topológico del predio Realizar seguimiento a los predios revisados en comisión
Técnico – Sist	Encargado de verificar si el predio ya esta replicado. Realiza el cambio de estado de un predio. Realizar seguimiento a los predios revisados en comisión
Jurídico	Encargado de realizar el seguimiento a los predios revisados en comisión.

Tabla 3.7 Descripción de Actores

Fuente: [Elaboración propia]

3.4.1.1. DIAGRAMA DE CASOS DE USO GENERAL

Los casos de uso describirán la secuencia de eventos de un actor, es decir es un documento narrativo de los actores del sistema.

El diagrama de casos de uso de la Figura 3.2 nos detalla los casos de uso generales y los actores involucrados en el sistema implementado en la Jefatura Regional Valles.

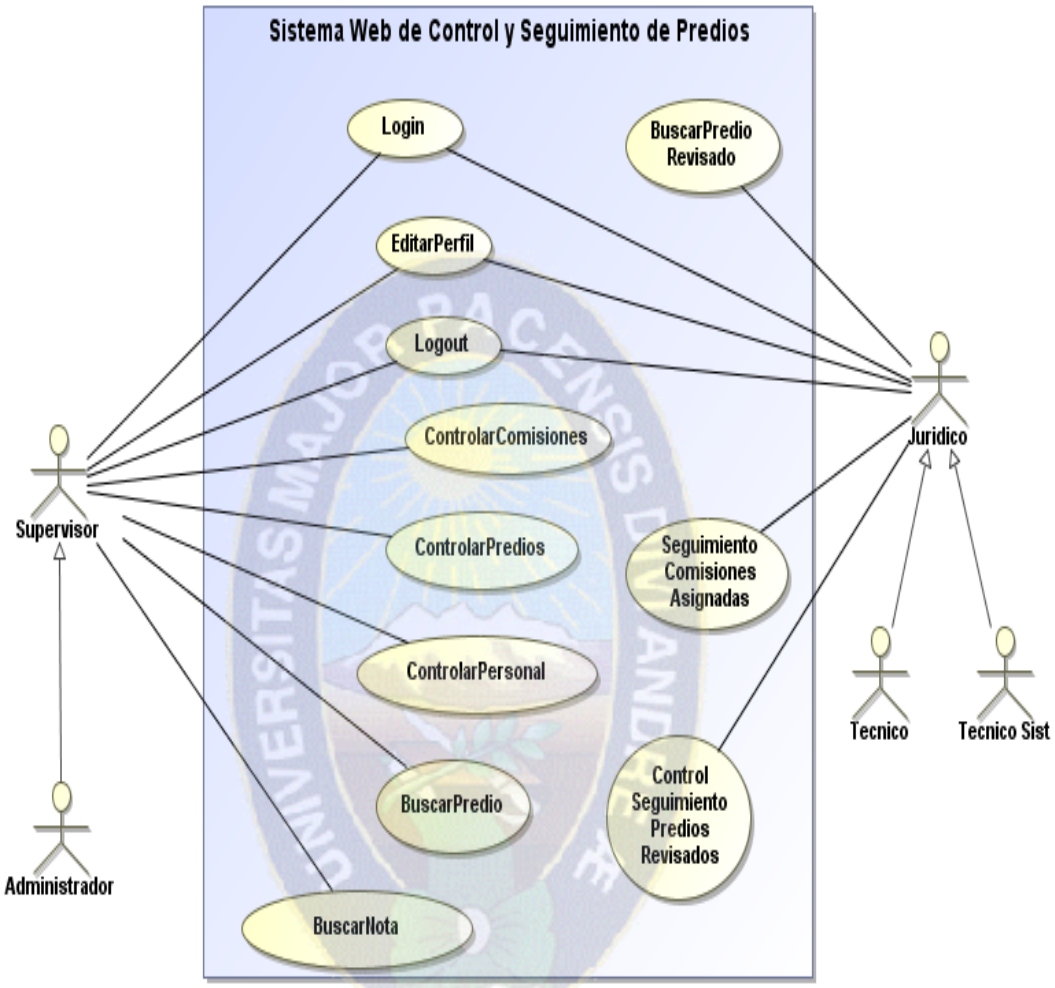


Figura 3.2 Diagrama de Casos de Uso General

Fuente: [Elaboración propia]

3.4.1.2. DIAGRAMA DE CASOS DE USO SUPERVISOR

Es el responsable directo del seguimiento de predios, comisiones y verificar carga de trabajo del personal de la Jefatura Regional. También le desplegara los predios que tengan mayor viabilidad.

Toma conocimiento de los predios enviados por las departamentales y realiza seguimiento a los predios pendientes de envió a la Jefatura Regional

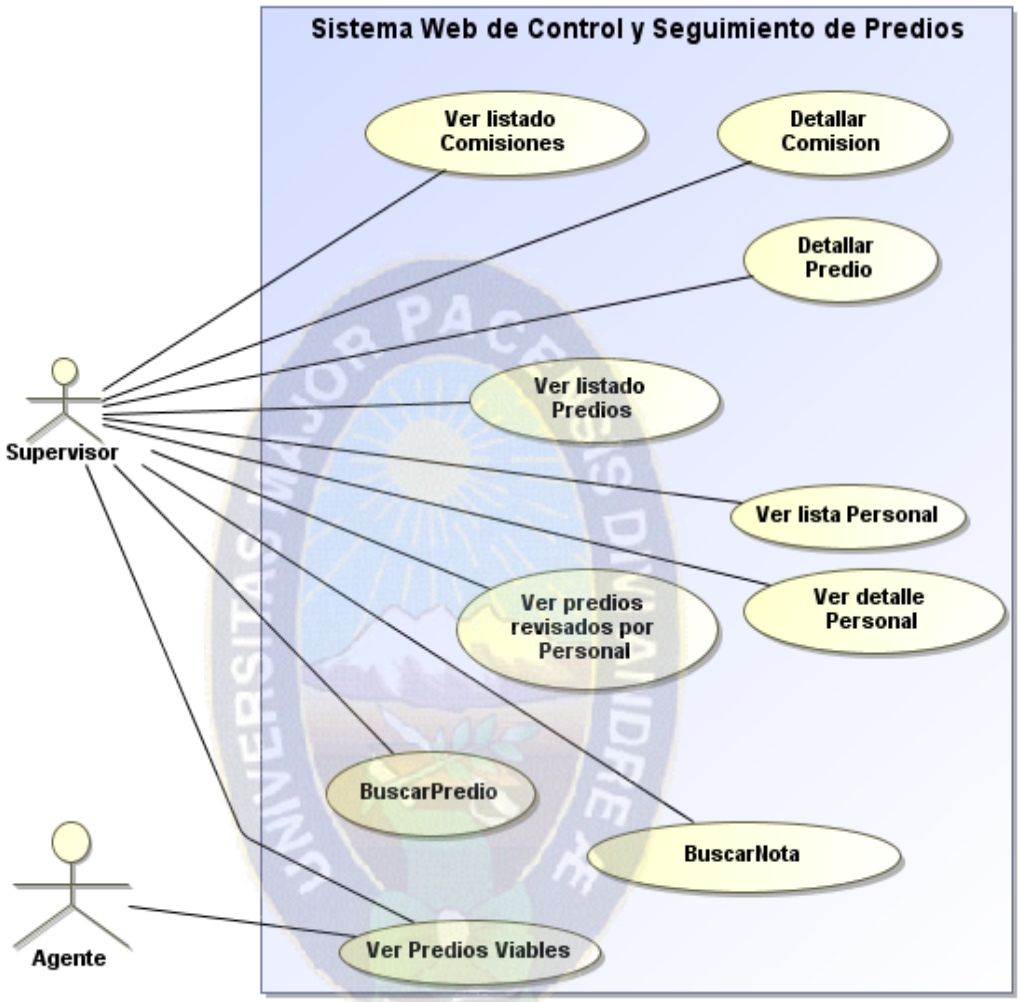


Figura 3.3 Diagrama de Casos de Uso Supervisor

Fuente: [Elaboración propia]

3.4.1.3. DIAGRAMA DE CASOS DE USO ADMINISTRADOR

Es el responsable del registro y control de predios, comisiones y personal, así como la recepción de predios enviados por las departamentales.

El actor administrador hereda casos de uso del actor Supervisor (Ver Figura 3.3)

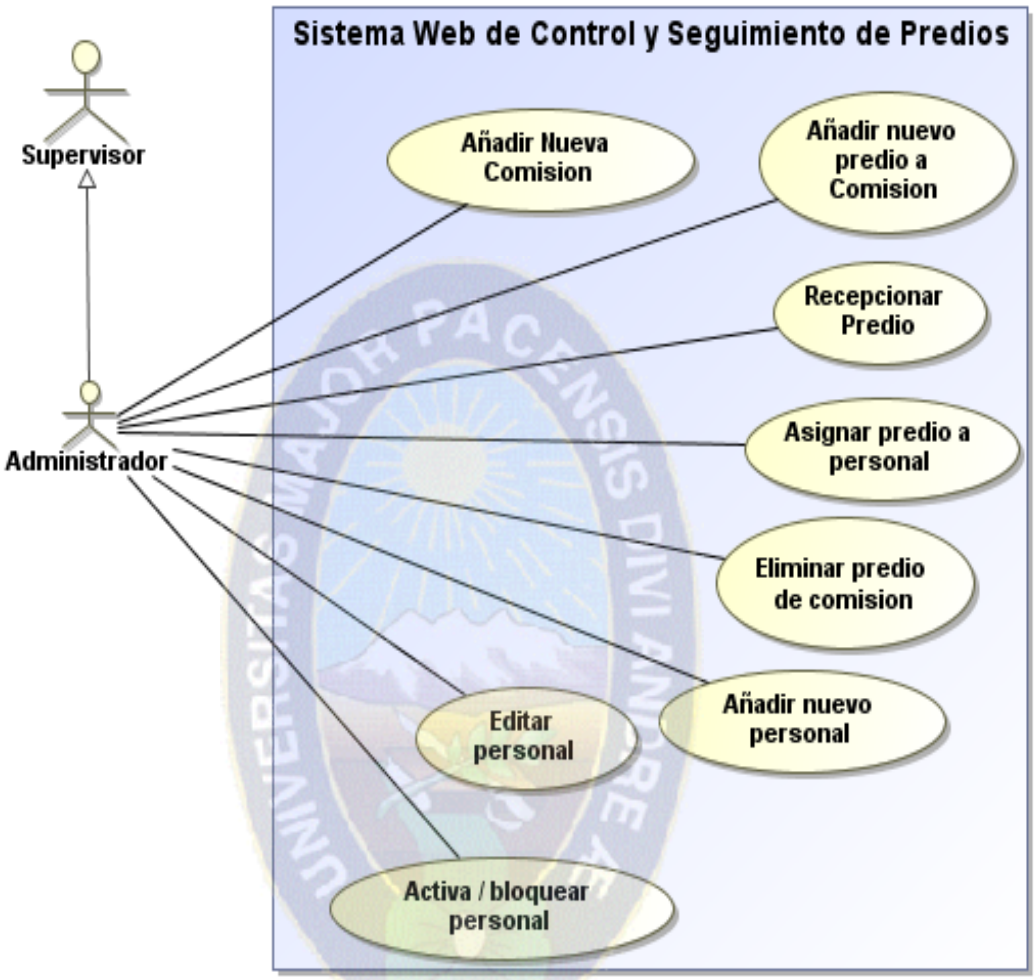


Figura 3.4 Diagrama de Casos de Uso Administrador

Fuente: [Elaboración propia]

3.4.1.4. DIAGRAMA DE CASOS DE USO JURIDICO

Es el responsable de realizar el seguimiento a las comisiones que realizo y predios revisados en comisión.

El sistema le desplegara un listado con los predios revisados viables los cuales serán obtenidos con la ayuda del agente implementado en el sistema.

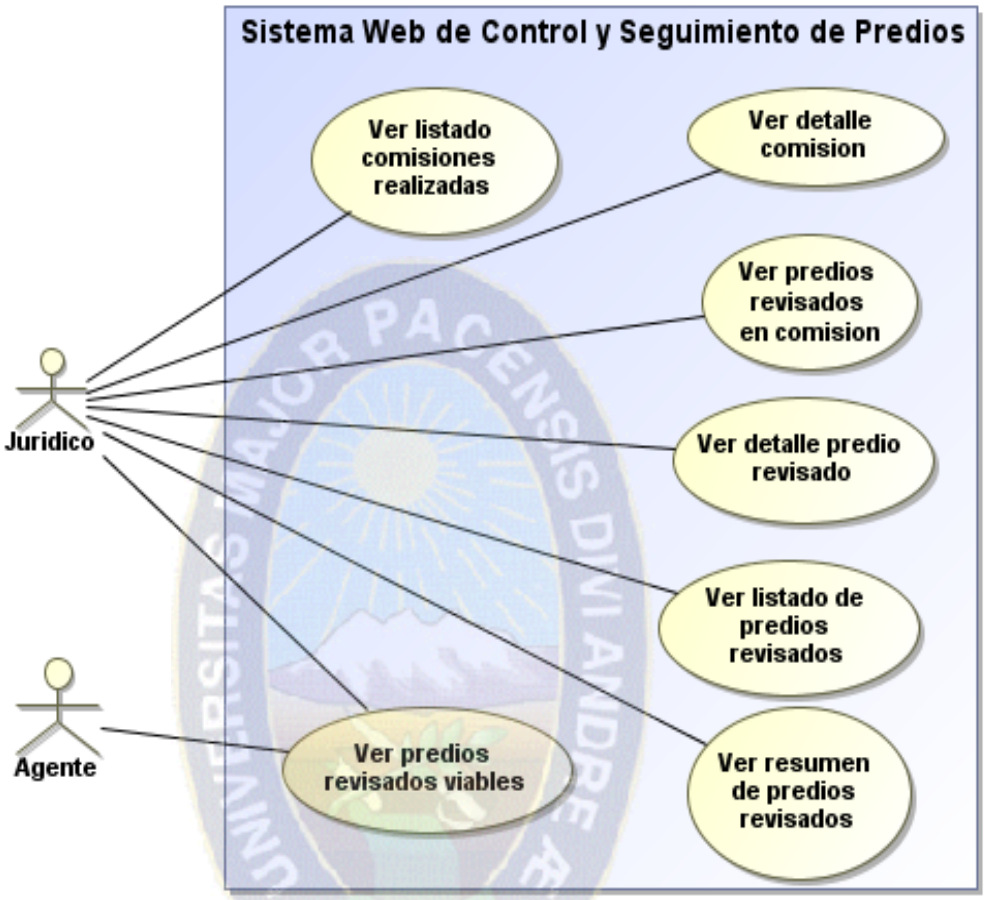


Figura 3.5 Diagrama de Casos de Uso Jurídico

Fuente: [Elaboración propia]

3.4.2. DIAGRAMA DE CASOS DE USO TECNICO Y TECNICO – SIST

El personal técnico realiza el registro del control topológico del predio previamente aprobado del cual es responsable el actor.

El personal Técnico – Sist es responsable de registrar la fecha en la cual se replico el predio así también registrar el respaldo y actualizar el estado del predio.

Como se puede observar en la Figura 3.5 ambos actores heredan del actor jurídico puesto que tienen casos de uso en común.

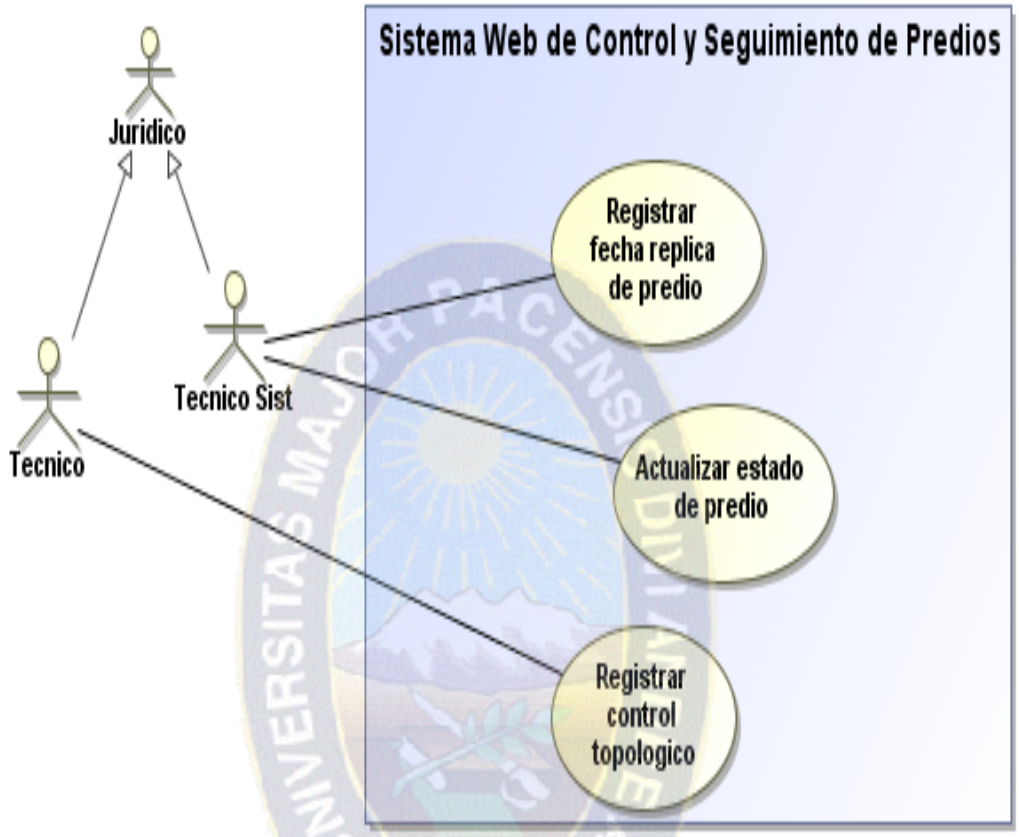


Figura 3.6 Diagrama de Casos de Uso Técnico y Técnico – Sist

Fuente: [Elaboración propia]

3.4.3. MODELO DE CONTENIDOS

Este modelo especifica cómo se encuentran relacionados los contenidos del sistema es decir define la estructura de los datos que se encuentran alojados en el sitio web el modelo de contenido obtiene la información relevante almacenada en el sistema, como se estructura y como se relaciona. Esto se representa mediante un diagrama de clases UML.

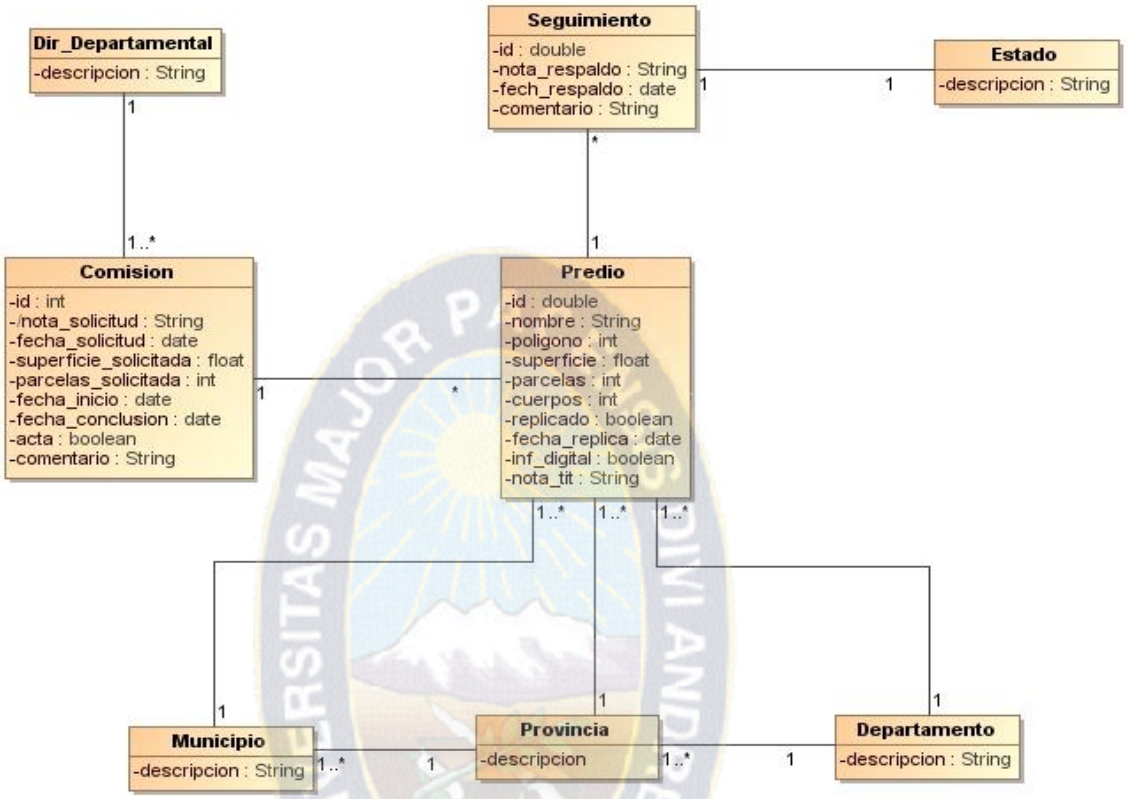


Figura 3.7 Modelo de Contenido

Fuente: [Elaboración propia]

3.4.4. MODELO DE USUARIO

El modelo de usuario tiene dos objetivos diferenciados:

Contiene las clases que define que información es almacenada en el contexto de una sesión. En este caso práctico una sesión está formada por el personal actual si su rol es Jurídico, Técnico o Técnico - Sist solo podrá tener manipular los predios que reviso en comisión.

Así también cabe mencionar si el rol del personal que ingrese al sistema es supervisor o administrador del sistema podrá manipular toda la información referente a los predios y comisiones.

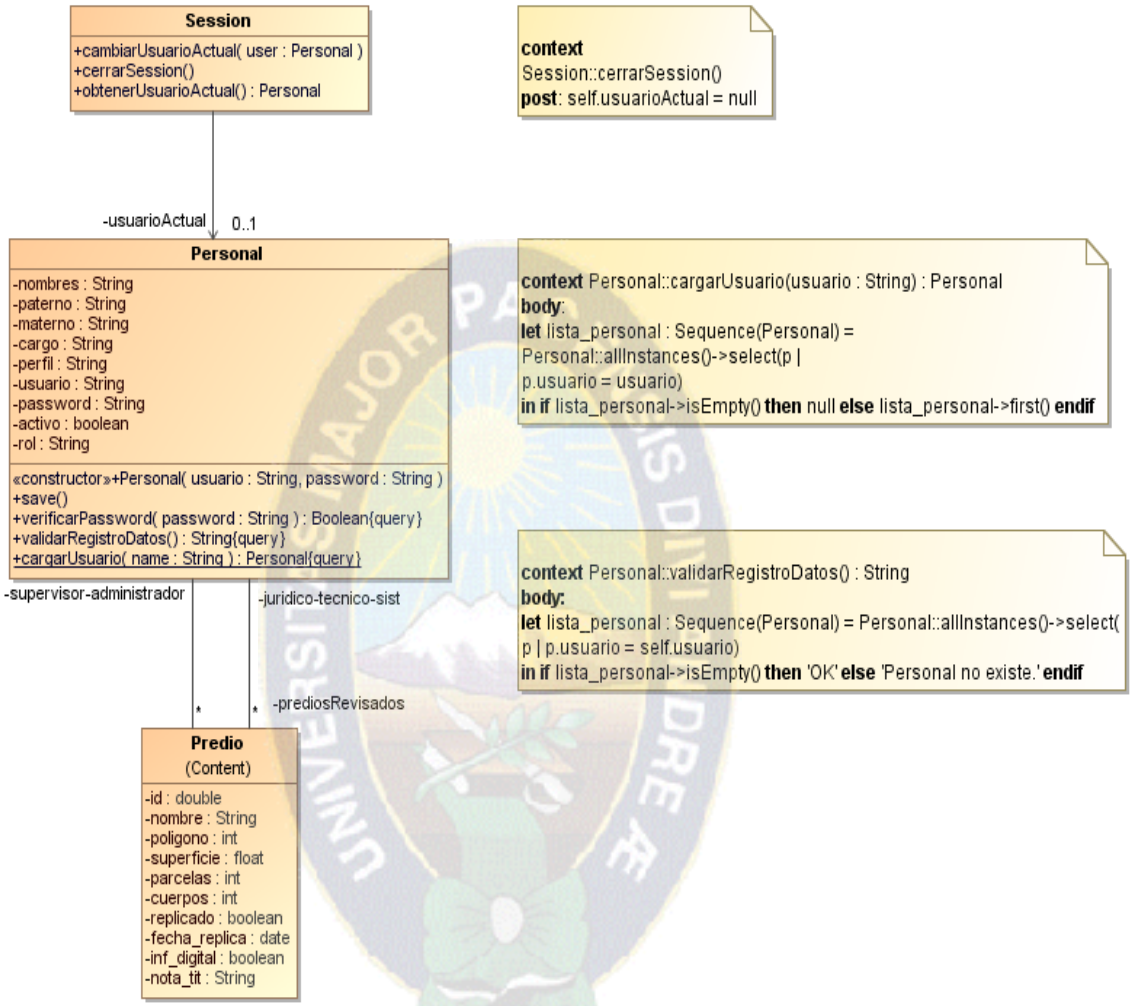


Figura 3.8 Modelo de Usuario

Fuente: [Elaboración propia]

3.4.5. MODELO DE NAVEGACION

En este modelo se especifica la relación interna del sitio web es decir cómo se relaciona cada página web con las demás con lo cual en definitiva es como se navega por el sitio web. El modelo que se presenta a continuación es el obtenido para el sistema desarrollado.

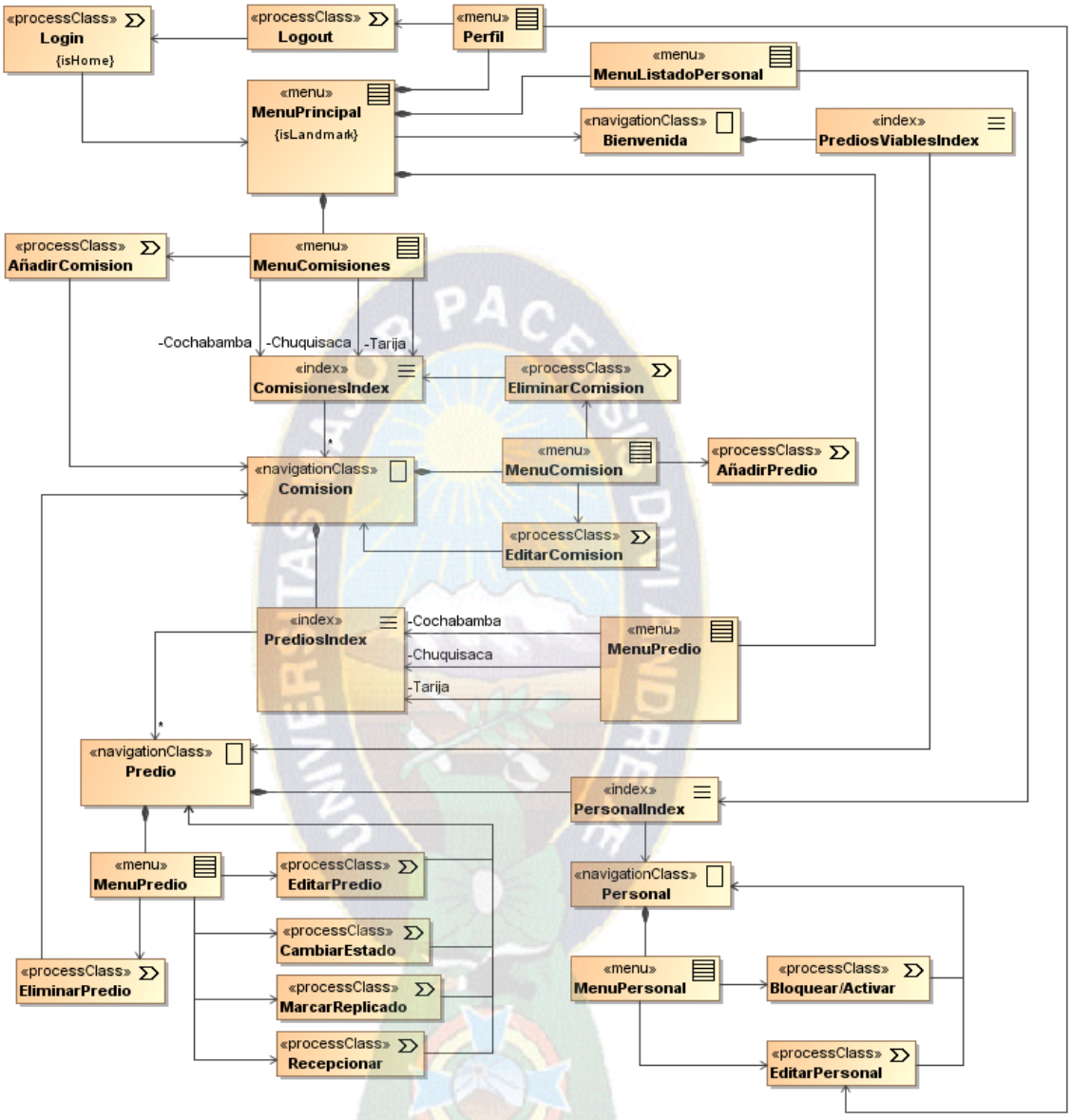


Figura 3.9 Modelo de Navegación

Fuente: [Elaboración propia]

En ese diagrama podemos ver como se relacionan las clases entre ellas llegando a un entendimiento superior que un diagrama UML con la extensión de UWE para sistemas web.

3.4.6. MODELO DE PROCESO

En este modelo se definen las acciones que realizan las clases de proceso especificadas en el modelo de navegación. El modelo de proceso se divide en dos partes la primera el modelo de Estructura de Procesos, en la cual se incluyen las relaciones entre las clases de proceso y la segunda es el modelo de Flujo de Procesos en el que se incluyen las actividades relacionadas con cada proceso describiendo el comportamiento interno de cada clase proceso.

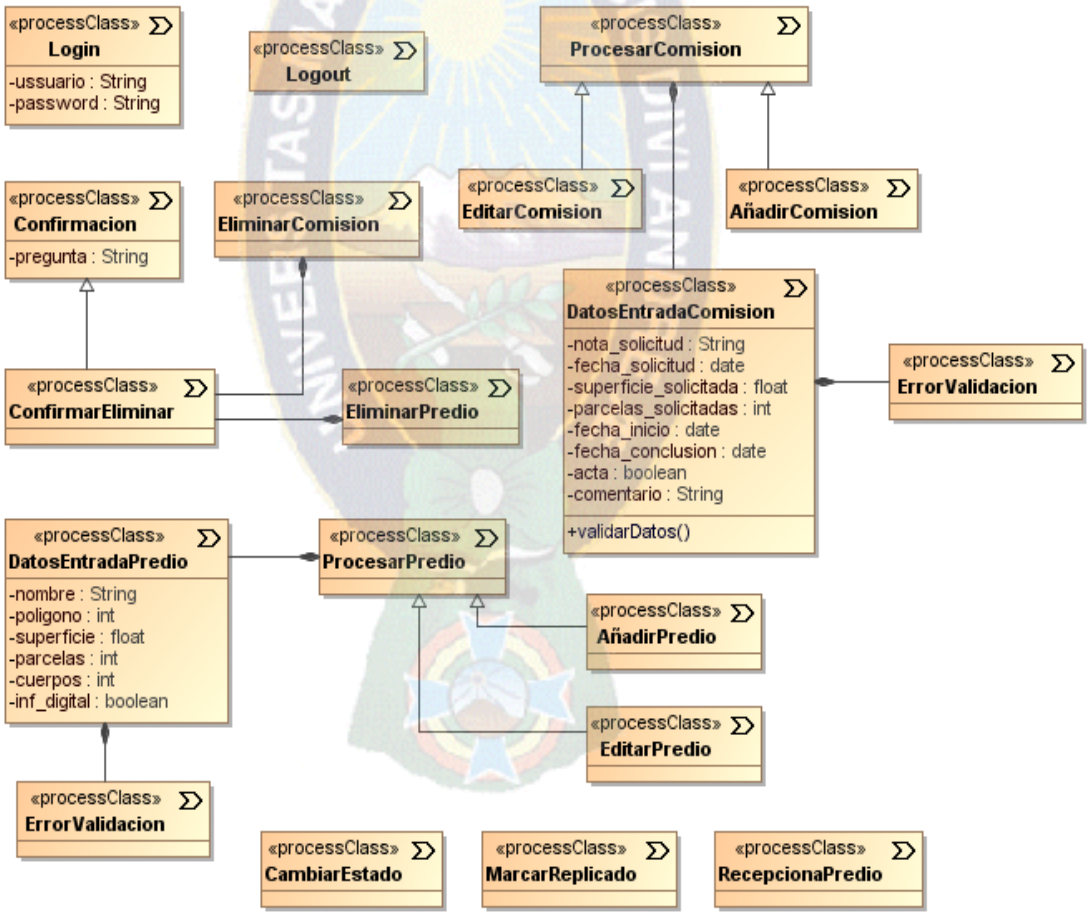


Figura 3.10 Modelo de Procesos

Fuente: [Elaboración propia]

3.4.7. MODELO DE PRESENTACION

Se contemplan las clases de navegación y de procesos que pertenecen a cada página web. Las propiedades que contiene por composición se representan como un rectángulo en el que un elemento queda contenido en otro

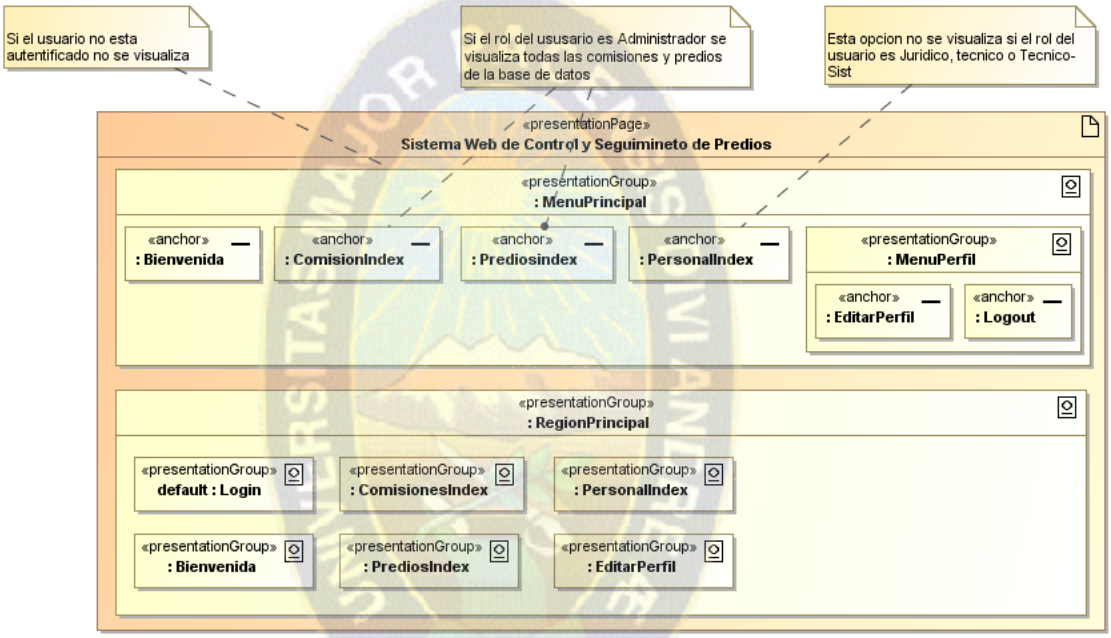


Figura 3.11Modelo de presentación pagina inicial

Fuente: [Elaboración propia]

La Figura 3.11 nos detalla el diseño de la pantalla principal del sistema implementado, a continuación se presentara los demás diseños como ser la pagina de Login, pagina de Bienvenida, Listar Comisiones, Listar Predios y en caso de que el usuario ingrese con el rol de administrador tendrá habilitado la opción de Listar Personal y de esta manera visualizar las opciones referente al Personal.

Figura 3.12 Modelo de presentación Ingreso al sistema

Fuente: [Elaboración propia]

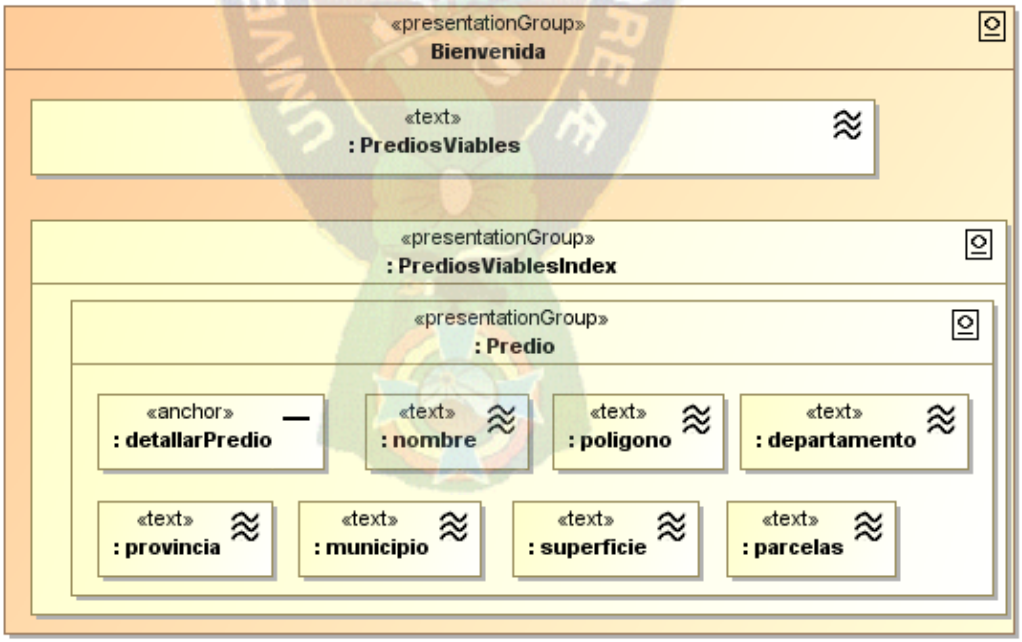


Figura 3.13 Modelo de presentación Bienvenida al sistema

Fuente: [Elaboración propia]

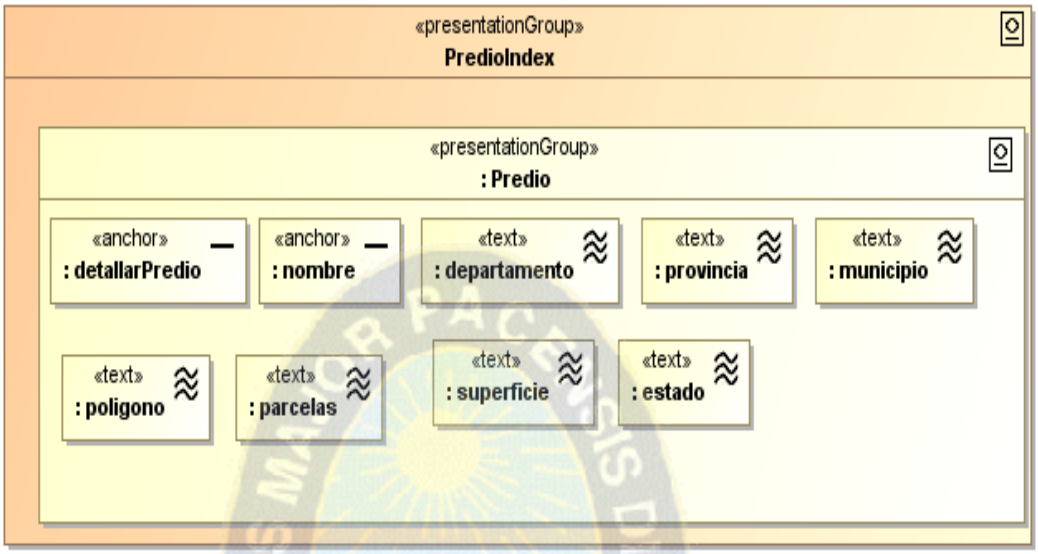


Figura 3.14 Modelo de presentación Listar Predios

Fuente: [Elaboración propia]

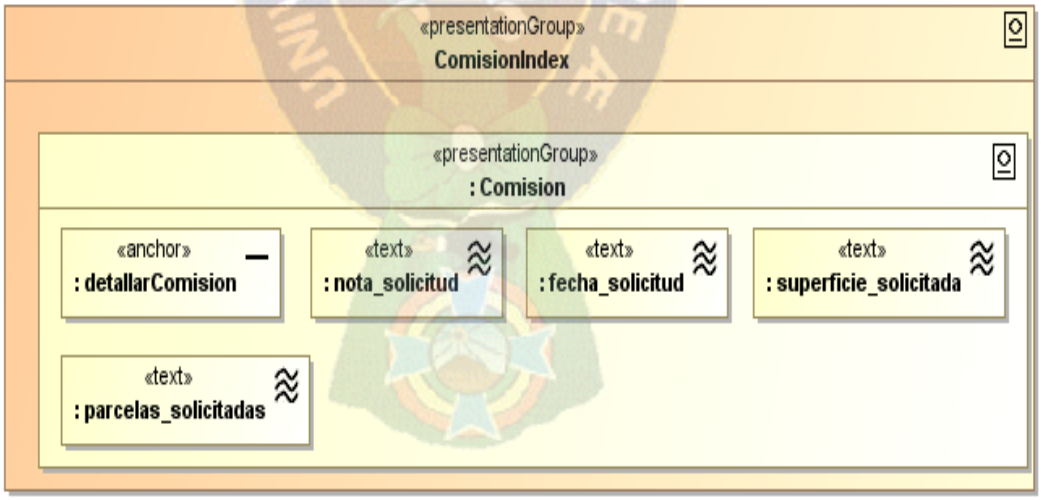


Figura 3.15 Modelo de presentación Listar Comisiones

Fuente: [elaboración propia]

Figura 3.16 Modelo de presentación Listado de Personal

Fuente: [Elaboración propia]

3.5. POST – GAME

Durante esta última etapa se realizaron las actividades de pruebas funcionales de la aplicación Web, se propusieron políticas de seguridad para el sistema se obtuvieron las métricas y se realizo el diseño de la ayuda para los usuarios.

CAPITULO IV
CALIDAD DE SOFTWARE

4.1. DEFINICION

La calidad de software es el conjunto de características de una entidad que le confieren su aptitud para satisfacer las necesidades expresadas y las implícitas, además indicando la dirección hacia donde debemos buscar soluciones.

4.1.1. CONFIABILIDAD

El análisis de confiabilidad se basara en que el sistema opera en puntos discretos de tiempo. Se debe considerar que:

- El sistema entero falla si uno de sus componente falla, esto es que el caso de que el sistema presente sus componentes en serie, para lo cual se tiene:

  $$P = P1\ P2\ P3 \ldots\ldots\ldots\ldots Pn$$

  Donde

  Pi = Probabilidad de que el componente i-esimo falle

  P = Probabilidad de que el sistema falle

- El sistema falla si y solo si todos sus componentes fallan. Esto en el caso de que el sistema presente todos sus componentes en paralelo para lo cual se tiene:

  $$R = 1 - (1-R1)\ (1-R2)\ (1-R3)\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(1-Rn)$$

  Donde:

  R1: Confiabilidad del modulo de registro proyectos

R2: Confiabilidad del modulo de registro entidades

R3: Confiabilidad del modulo de reportes

R4: Confiabilidad del modulo personal

Asignando valores de error a cada uno de estos módulos (posibles errores de proceso). Tenemos:

R1:95% con error de 0.05% = e-0.05

R2:98% con error de 0.02% = e-0.02

R1:99% con error de 0.01% = e-0.01

R1:95% con error de 0.05% = e-0.05

Agrupando por módulos se tiene:

Rs1 = 1 – (1-R1) (1-R2) = 1 – (1-0.95) (1-0.98)

= 1 – (0.05) (0.02)

= 0.99

Es decir que 0.01 es el grado de probabilidad de que cualquier de los componentes falle:

Rs2 = (R3) (R4) = (0.99) (0.95)

= 0.94

Es decir que el 0.06 es el grado de probabilidad de que cualquier de los componentes falle:

Analizando todo el sistema tenemos:

Rs1 =1 – (1-R1) (1-R2) (1-R3R4) = 1 – (1-095) (1-0.98) (1-0.94)

=0.99

Se tiene por lo tanto un 0.99 de confiabilidad, y una probabilidad de 0.01 de que cualquiera de los componentes falle.

4.1.2. MEDICION DEL SOFTWARE

El análisis de confiabilidad se basara en que el sistema opera en puntos discretos de tiempo. Se debe considerar que:

- El sistema entero falla si uno de sus componente falla, esto es que el caso de que el sistema presente sus componentes en serie, para lo cual se tiene:

  P = P1 P2 P3……Pn

  Donde

  Pi = Probabilidad de que el componente i-esimo falle

  P = Probabilidad de que el sistema falle

- El sistema falla si y solo si todos sus componentes fallan. Esto en el caso de que el sistema presente todos sus componentes en paralelo para lo cual se tiene:

  R = 1 – (1-R1) (1-R2) (1-R3) ……(1-Rn)

  Donde:

  R1: Confiabilidad del modulo de registro proyectos

  R2: Confiabilidad del modulo de registro entidades

  R3: Confiabilidad del modulo de reportes

  R4: Confiabilidad del modulo personal

Asignando valores de error a cada uno de estos módulos (posibles errores de proceso). Tenemos:

R1:95% con error de 0.05% = e-0.05

R2:98% con error de 0.02% = e-0.02

R1:99% con error de 0.01% = e-0.01

R1:95% con error de 0.05% = e-0.05

Agrupando por módulos se tiene:

Rs1 = 1 – (1-R1) (1-R2) = 1 – (1-0.95) (1-0.98)

= 1 – (0.05) (0.02)

= 0.99

Es decir que 0.01 es el grado de probabilidad de que cualquier de los componentes falle:

Rs2 = (R3) (R4) = (0.99) (0.95)

= 0.94

Es decir que el 0.06 es el grado de probabilidad de que cualquier de los componentes falle:

Analizando todo el sistema tenemos:

Rs1 =1 – (1-R1) (1-R2) (1-R3R4) = 1 – (1-095) (1-0.98) (1-0.94)

=0.99

Se tiene por lo tanto un 0.99 de confiabilidad, y una probabilidad de 0.01 de que cualquiera de los componentes falle.

4.1.3. FUNCIONALIDAD

El punto función es una métrica orientada a la función. Es una indirecta de software y del proceso por el cual se desarrolla.

Se centra en la funcionalidad o utilidad del programa. Los puntos de función se calculan llenando la tabla 3.14 para el cálculo se determinan cinco características de dominios de información.

- **Número de entrada usuario.** Se cuenta cada entrada de usuario que proporciona al software diferentes datos orientados a la aplicación. Las entradas deben ser restringidas de las peticiones que se contabilizan por separado.
- **Número de salidas de usuario.** La salida se refiere a informes, pantallas, mensajes de error.
- **Número de peticiones de usuarios.** Una petición está definida como una entrada interactiva que resulta de la generación de algún tipo de respuesta en forma de salida interactiva.
- **Número de archivos.** Se cuenta cada archivo maestro lógico.
- **Número de interfaces externas.** Se cuenta todas las interfaces legibles por el ordenador que son utilizados para transmitir información a otro sistema.

4.1.3.1. OBTENCION DEL PUNTO FUNCION

PF = Cuenta Total (X + Y * fi)

Donde:

PF = Medida de funcionalidad del sistema, dad la aplicación como valor de normalización.

CUENTA TOTAL = Suma de todas las entradas obtenidas en la tabla 3.14 para lo cual se considero los siguientes puntos:

N1 Número de Entradas de Usuario

N2 Número de Salidas de Usuario

N3 Número de Peticiones de Usuario

N4 Número de Archivos

N5 Número de Interfaces externas

Ni * factor = CT = Cuenta Total

X = Confiabilidad del Sistema

Y = Margen de error

fi = Valores de ajuste de la complejidad obtenidas en la (Tabla 3.14) muestran la computación del Punto de Función.

Estimaremos a continuación cada uno de los factores de ponderación de la complejidad, para luego poder hacer el ajuste de la complejidad.

PARAMETROS DE MEDICION	**CUENTA**	**COMPLEJIDAD**	**TOTAL**
Número de entradas de usuario	7	4	28
Número de salidas de usuario	10	5	50
Número de peticiones de usuario	5	7	35
Número de archivos o tablas	8	10	80
Número de interfaces de usuario	12	5	60
Cuenta Total			253

Tabla 4.1 Factor de ponderación de parámetros

Fuente: [Elaboración propia]

El factor complejidad se evalúa en una escala de 0 a 5 que se muestran a continuación en la siguiente figura:

0	1	2	3	4	5
Noinfluencial	Incidencial	Moderado	Medio	Significativo	Esencial

Tabla 4.2 Computación de punto función

Fuente: [Elaboración Propia]

Nº	Cuestionamiento	Factor
1	¿Requiere el sistema copias de seguridad y de recuperación?	5
2	¿Se requiere comunicaciones de datos?	5
3	¿Es criticado el rendimiento?	3
4	¿Sera ejecutado el sistema en un entorno operativo existente y fuertemente utilizado?	5
5	¿Requiere el sistema entrada de datos interactiva?	3
6	¿Requiere la entrada de datos interactiva que las transacciones de entrada se lleven a cabo sobre múltiples pantallas o variadas operaciones?	4
7	¿Se actualizan los archivos maestros de forma interactiva?	4
8	¿Son complejos las entradas, las salidas, los archivos o las peticiones?	3
9	¿Es complejo el procedimiento interno?	4
10	¿Se ha diseñado el código para ser utilizable?	4
11	¿Están incluidas en el diseño la conservación y la instalación?	4
12	¿Se ha diseñado el sistema para soportar múltiples instalaciones?	4
13	¿Se ha diseñado la aplicación para facilitar los cambios y para ser fácilmente por el usuario?	5
	TOTAL	53

Tabla 4.3 Calculo medida de complejidad

Fuente: [Elaboración Propia]

A continuación se realiza el cálculo de Punto de Función de la funcionalidad del sistema con el resultado de la sumatoria de la tabla 3.15 para lo cual se toma en cuenta la confiabilidad del sistema, un margen de error de 0.01 y se reemplaza a la ecuación que se muestra a continuación:

PF = Cuenta Total (X + Y * fi)

PF = 253 (0.67 + 0.1 (53))

PF = 303.06

Entonces podemos concluir que la funcionalidad del sistema es optima de acuerdo al resultado obtenido ya que es mayor a 300 de acuerdo a la siguiente tabla:

ESCALA	**OBSERVACION**
PF > 300	Optimo
200 < PF < 300	Bueno
100 < PF < 200	Suficiente
PF < 100	Deficiente

Tabla 4.4 Escala de Punto Función

Fuente: [Elaboración Propia]

4.1.4. FIABILIDAD

En vista de que la fiabilidad es fundamental para la reutilización del software, y de acuerdo con las sub-características que conforma la fiabilidad, el sistema cumple con lo que esta fiabilidad, ya que la frecuencia y gravedad de los fallos, es pequeña, esto porque el sistema fue desarrollado rescatando errores relativamente grandes de otros proyectos, de los cuales existía poca fiabilidad en cuanto a la reutilización del sistema.

Para la obtención de la fiabilidad se aplica la siguiente ecuación:

$$\text{Fiabilidad} = 1 - \left(\frac{\#Errores}{\#lienasdecodigo} \right)$$

Reemplazando valores, obtenemos:

$$\text{Fiabilidad} = 1 - \left(\frac{5}{1500}\right) = 0.9966$$

Fiabilidad = 0.99666 * 100% = 99.666

Fiabilidad = 99.7 %

Así, analizando el porcentaje de errores a presentarse del sistema se llega a la conclusión de que le sistema es un 99.7% fiable.

4.1.5. PORTABILIDAD

El sistema es adaptable a cualquier entorno del mismo propósito, por que el paquete de instalaciónno es muy complejo, tiene una facilidad de instalación, ya que el código la base de datos es portable y solo requiere ser copiado, y el ejecutable puede ser instalado o copiado el archivo exe del sistema.

Además de que el sistema puede compartir algunos recursos, el sistema reemplaza a otro sistema el cual tenía el mismo propósito en el mismo entorno.
Para tal hecho, se aplica de la siguiente ecuación:

$$\text{Portabilidad} = 1 - \left(\frac{\#diasparaportarelsistema}{\#diasparaimplementarelsistema}\right)$$

Reemplazando valores se tiene:

$$\text{Portabilidad} = 1 - \left(\frac{1}{5}\right) = 0.857142857$$

$$\text{Portabilidad} = 0.857142857 * 100\% = 85.7142857$$

$$\text{Portabilidad} = 85.7\%$$

4.1.6. FLEXIBILIDAD

El coste de modificación del producto de acuerdo a las especificaciones del usuario final, es relativamente pequeño, debido a que se trabaja sobre una base prácticamente estable. Además de que la Base de Datos del sistema está estructurado y proyectado para ser implementado vía Web.

Para la obtención de la flexibilidad, se aplica la siguiente ecuación:

$$\text{Flexibilidad} = 1 - 0.05\ (\#\text{medio de días – hombre por cambio})$$

Reemplazando valores se tiene:

$$\text{Flexibilidad} = 1 - 0.05\ (3\text{-}1) = 0.9$$

Flexibilidad = 0.9 * 100%

Flexibilidad = 90%

De esta manera se puede decir que el sistema en cuanto al coste de modificaciones es flexible en un 90%.

4.1.7. RESULTADO FINAL

CARACTERISTICAS	RESULTADO
Usabilidad	90
Fiabilidad	99.7
Portabilidad	85.7
Flexibilidad	90
Evaluación de calidad total	91.35

Tabla 4.5 Evaluación de calidad total

Fuente: [Elaboración Propia]

Interpretamos la anterior tabla como, si un usuario especifico visita la aplicación web tendrá un grado de satisfacción de 73.5% al utilizarla.

4.2. ANALISIS COSTO BENEFICIO DEL SISTEMA COCOMO

4.2.1. COSTO DE ANALISIS DE PROGRAMACION

Para el análisis de costos del sistema, haremos uso del modelo de estimación empíricade:

COCOMO

- El sistema es considerado como un proyecto sencillo y pequeño (menor a 50.000 líneas de código)
- Uso del modelo COCOMO de tipo Modo Orgánico, que es utilizado en proyectos pequeños y sencillos, en los que trabajan pequeños equipos.
- Para la obtención del Esfuerzo y el Tiempo Empleado, se usa ecuaciones matemáticas.

La ecuación que nos ayudara hallar el esfuerzo, viene dada de la siguiente manera:

$$E = a * (KLCD)^b \text{ (personas/mes)}$$

Donde:

E es el esfuerzo expresado en personas por mes

a,b son constantes empíricas

KLDC es un número estimado de código fuente en miles distribuidas.

Reemplazando los datos en la ecuación, se tiene:

$$E = 2.4 * (0.9)^{1.05} = 2.148650$$

$$E = 2.1 \text{ (personas/mes)}$$

De esta manera el esfuerzo aplicado para la realización del sistema, es de 2 personas por mes.

De igual manera la obtención del tiempo empleado para el desarrollo del sistema se hace uso de la siguiente ecuación:

$$T = c * E^d \text{ (meses)}$$

Donde

T es el tiempo de desarrollo expresado en meses

c.d son constantes empíricas

E es el esfuerzo expresado en personas por mes

Reemplazando los datos en la ecuación, se tiene:

$$T = 2.5 * (2.1)^{0.38} = 3.3142353$$

$$T = 3.3 \text{ (meses)}$$

Así, el tiempo aproximado de desarrollo del sistema, es de 3 meses.

Los costos de este sistema se distribuyen de la siguiente manera:

Descripción	Cantidad	Costo mensual	Costo	Costo total
Programación	1	n/a	$ 12000,00	$ 12000,00
Software	n/a	n/a	$ 0	$ 0
Equipo Nuevo (computadoras)	n/a	n/a	$ 0	$ 0
Actualización (hardware)	1	n/a	$ 370,00	$ 370,00
Adiestramiento	300hrs / 3 empleados		$ 5,00	$ 1500,00
TOTAL				$ 13870,00

Tabla 4.6 Análisis de costos

Fuente: [Elaboración Propia]

4.2.2. VALOR ACTUAL NETO

También conocido valor actualizado neto (en ingles Net PresentValue), cuyo acrónimo es VAN (en inglés NPV), es un procedimiento que permite calcular el valor presente de un determinado nemeo de flujos de caja futuros, originado por una inversión. La metodología consiste en descontar al momento actual (es decir, actualizar mediante una tasa) todos los flujos de caja futuros del proyecto. A este valor se le resta la inversión inicial, de tal modo que el valor obtenido es el valor actual neto del proyecto.

Aplicando la formula obtendremos el Valor Actual Neto:

$$\text{VAN} = \sum \frac{ganancias}{(1+12\%)^n} - \sum \frac{costos}{(1+22\%)^n}$$

$$\text{VAN} = \sum \frac{2000}{(1+12\%)^5} - \sum \frac{13870}{(1+22\%)^5}$$

$$\text{VAN} = \$\ 17500$$

4.2.3. INTERPRETACION DEL RESULTADO VAN

Valor	Significado	Decisión a tomar
VAN > inversión	La inversión producirá ganancias por encima de la rentabilidad exigida	El proyecto puede aceptarse.

VAN < inversión	La inversión producirá ganancias por debajo de la rentabilidad exigida.	El proyecto debería rechazarse
VAN = inversión	La inversión no producirá ni ganancias ni perdidas	Dado que el proyecto no agrega valor monetario por encima de la rentabilidad exigida, la decisión debería basarse en otros criterios, como la obtención de un mercado mejor posicionamiento en el mercado u otros factores.

Tabla 4.7 Intervalo de decisión de ganancias

Fuente: [Elaboración Propia]

Entonces, como el valor del VAN es mayor a la inversión inicial, podemos decir que es recomendable que el proyecto sea aceptado.

Luego:

$$C/B = \sum \frac{ganancias}{costos}$$

$$C/B = \sum \frac{20000}{13870}$$

$$C/B = \$1.45$$

De este resultado podemos concluir que, en la realización de este proyecto, por cada dólar invertido ganamos 45 dólares.

Luego:

$$TIR = \frac{\sum ganancias - costos}{(1+3\%)^n}$$

$$TIR = \frac{\sum 20000 - 13870}{(1+3\%)^5}$$

$$TIR = 2919$$

Por lo tanto la rentabilidad que nos está proporcionando el proyecto es de $ 2919.

CAPITULO V
CONCLUSIONES Y RECOMENDACIONES

5.1. CONCLUSIONES

Con la finalización del presente proyecto y la implementación del sistema se cumplió con el objetivo general propuesto. Dicho sistema optimiza la obtención de la información de los predios y comisiones realizadas por el personal de la Jefatura Regional Valles, mejorando el control y manejo de la información haciendo posible el seguimiento a los predios y control de carga de trabajo al personal de la jefatura.

Así mismo los objetivos específicos planteados también se desarrollaron a cabalidad cumpliendo cada uno de ellos:

- Se implemento agentes inteligentes aplicando la metodología MaSe el mismo permitió modelar la forma adecuada los agentes de análisis de datos de los predios y el de generar un listado de predios con mayor viabilidad.
- La implementación de agentes inteligentes los cuales nos ayudan en la obtención de predios viables asignados al personal.
- Los procesos de registro de predios y comisiones se automatizaron para brindar un mejor servicio a la Jefatura.

- El sistema alberga cuatro tipos de usuarios cada cual con un perfil diferente: Supervisor, Administrador, Jurídico, Técnico y Técnico – Sist, satisfaciendo con sus necesidades de información.
- Se realizo el análisis e implementación de la base de datos que organiza toda la información administrada por la Jefatura Regional Valles.
- Se logro optimizar la obtención de informes y reportes por medio de interfaces diseñadas a partir de la experiencia de los usuarios de la Jefatura Regional Valles
- El control y seguimiento de predios y comisiones es más eficiente gracias a la base de datos centralizada que coadyuva a la obtención de información de predios.

Con respecto al desarrollo del sistema de gestión se utilizo la metodología Scrum que permitió el progreso de la misma, con la organización y planificación de las tareas captadas a través de las historias de usuario que fueron útiles para la ingeniería de requerimientos, además se incorporo UWE en el modelado del sistema para el mejor entendimiento de los procesos.

Es por ello que la metodología Scrum con el apoyo de UWE, permitieron la conclusión del sistema y la obtención del software de calidad, con las funcionalidades requeridas por la institución en este caso la Jefatura Regional Valles.

Así mismo el lenguaje de programación PHP permitió el desarrollo de un software con una interface amigable con los diferentes usuarios involucrados en el sistema.

5.2. RECOMENDACIONES

El trabajo desarrollado denominado “Sistema Web de Control y Seguimiento de Predios Usando Agentes Inteligentes” fue implementado en un entorno WAN en las instalaciones de la Jefatura Regional Valles.

Sin embargo las recomendaciones para el buen manejo del sistema son:

- Capacitaciones semestrales al personal de la Jefatura Regional Valles, con el motivo de formar y verificar el uso adecuado del sistema.
- Implementar el sistema en las demás Jefaturas Regionales.
- Realizar copias de seguridad de la información.
- Desarrollar un modulo el cual realice copias de seguridad automáticamente.

BIBLIOGRAFIA

[Sampiere2002] Sampiere, Fernandez y Baptista – 2002, "Metodología de la Investigación", tercera edición, Mc Graw Hill, Mexico.

[I. Jacobsob200] I. Jacobsob, G. Boch, J. Rumbaugh – 2000, "El Proceso Unificado de Desarrollo de Software", Addison Wesley, Mexico.

[Pressman2010] Pressman – 2010, "Ingeniería del Software: Un enfoque practico", sexta edición, Mc Graw Hill.

[Sommerville2005] IanSommerville - 2005, "Ingeniería del Software", Séptima Edición, Pearson Addison Wesley, Pearson Education S.A. Madrid.

[Senn2001] Sebb – 1990, "Ingeniería de Sistemas de Información para la Administración", sexta edición, Mc Graw Hill, Mexico.

[Nerur2005] Nerur, Mahapatra, Mangalaraj – 2005, "Los desafíos de la migración a las metodologías agiles", ACM.

[Ramirez2004] Ramírez – 2004, "Editor Inteligente Basado en Agentes", Benemérita Universidad Autónoma de Puebla.

[Berney1996] Berney – 1996, "Agentes de Software", Universidad Metropolitana de Manchester.

[Rodriguez2008] Rodríguez – 2008, Tesis de Maestría "Estudio de la Aplicación de Metodologías Agiles para la Evolución de Productos Software", Facultad de Informática Universidad Politécnica de Madrid.

[Ramos2011] Ramos – 2011, tesis Doctoral "Ingeniería de software orientada a agentes den el modelado de sistemas multimedia", Escuela Superior de Ingeniería Informática Universidad de Vigo.

[Graig1999] Graig L. – 1999, "Uml y Patrones"

[Kniberg2007] Kniberg H. – 2007, "SCRUM y XP desde las trincheras".

[Brenner1998] W. Brenner, R. Zarnekow y H. Witting – 1998, “Intelligent Software Agents”, Springer-Verlag.

[Koch2002] Nora Parcus de Koch – 2002, “Software Enginieering for Adaptive Hypermedia Systems”, Ludwig MaximiliansUniversitätMünchen, Alemania.

[Baresi2001] L- Baresi, F. Garzotto, P. Paolini – 2001, “Extending UML for Modelling Web Applications”, Annual International Conference on System Sciences Maui.

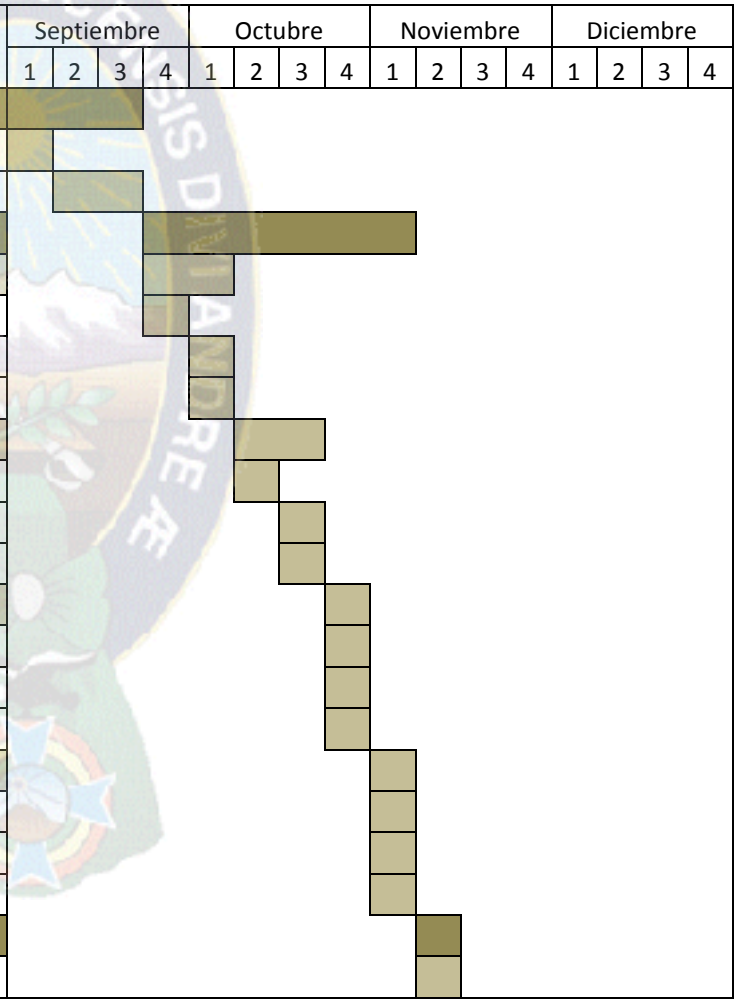
[Gallego2004] Gallego Fernando, LLorensFaraon, Rizo Ramon – 2004, “Breve Analisis de algunas metodologías de diseño de SMA”, Departamento Ciencias de la Computación e Inteligencia Artificial Universidad de Alicante, España

ANEXOS

ANEXO A

DIAGRAMAGANTT DEL PROYECTO

Id	Nombre de tarea	Comienzo	Fin	Duracion (dias)
1	PRE - GAME	01/09/2013	21/09/2013	20
2	Planeacion	15/09/2013	17/09/2013	2
3	Arquitectura	18/09/2013	21/09/2013	3
4	GAME	22/09/2013	03/11/2013	42
5	Primer Sprint	22/09/2013	13/10/2013	21
6	Planeacion del Sprint	22/09/2013	25/09/2013	3
7	Desarrollo del Sprint	25/09/2013	04/10/2013	9
8	Revision Sprint	06/10/2013	13/10/2013	7
9	Segundo Sprint	14/10/2013	22/10/2013	8
10	Planeacion del Sprint	14/10/2013	16/10/2013	2
11	Desarrollo del Sprint	16/10/2013	20/10/2013	4
12	Revision Sprint	20/10/2013	22/10/2013	2
13	Tercer Sprint	22/10/2013	27/10/2013	6
14	Planeacion del Sprint	22/10/2013	23/10/2013	1
15	Desarrollo del Sprint	23/10/2013	27/10/2013	4
16	Revision Sprint	27/10/2013	28/10/2013	1
17	Cuarto Sprint	28/10/2013	03/11/2013	6
18	Planeacion del Sprint	28/10/2013	30/10/2013	2
19	Desarrollo del Sprint	30/10/2013	01/11/2013	2
20	Revision Sprint	01/11/2013	03/11/2013	2
21	POST - GAME	04/11/2013	07/11/2013	3
22	Cierre	04/11/2013	07/11/2013	3



ANEXO B

SECCION IV

DEL INSTITUTO NACIONAL DE REFORMA AGRARIA

ARTICULO 17º (Instituto Nacional de Reforma Agraria).

I. Créase el Instituto Nacional de Reforma Agraria (INRA), como entidad pública descentralizada del Ministerio de Desarrollo Sostenible y Medio Ambiente, con jurisdicción nacional, personalidad jurídica y patrimonio propio.

II. El Instituto Nacional de Reforma Agraria (INRA) es el órgano técnico-ejecutivo encargado de dirigir, coordinar y ejecutar las políticas establecidas por el Servicio Nacional de Reforma Agraria.

ARTICULO 18º (Atribuciones).

El Instituto Nacional de Reforma Agraria tiene las siguientes atribuciones:

1. Dirigir, coordinar y ejecutar políticas, planes y programas de distribución, reagrupamiento y redistribución de tierras, priorizando a los pueblos y comunidades indígenas, campesinas y originarias que no las posean o las posean insuficientemente, de acuerdo a la capacidad de uso mayor de la tierra;
2. Proponer, dirigir, coordinar y ejecutar las políticas y los programas de asentamientos humanos comunarios, con pobladores nacionales;
3. Emitir y distribuir títulos, en nombre de la autoridad máxima del Servicio Nacional de Reforma Agraria, sobre tierras fiscales incluyendo las expropiadas o revertidas a dominio de la Nación, tomando en cuenta la vocación de uso del suelo establecida en normas legales correspondientes;
4. Emitir disposiciones técnicas para la ejecución del catastro rústico legal de la propiedad agraria, coordinar su ejecución con los municipios y otras entidades públicas o privadas;
5. Determinar la ubicación y extensión de las tierras fiscales disponibles, de las tierras comunitarias de origen, de las áreas clasificadas por normas legales y de la propiedad agraria en general;

6. Expropiar fundos agrarios, de oficio por la causal de reagrupamiento y redistribución, o a denuncia de la Superintendencia Agraria, por incumplimiento de la función económico-social, en los términos establecidos en esta ley;
7. Revertir tierras de oficio o a denuncia de las entidades recaudadoras o beneficiarias de impuestos, de las comisiones agrarias departamentales y de la Comisión Agraria Nacional, por la causal de abandono establecida en esta ley;
8. Determinar y aprobar las áreas y superficies a distribuir por dotación o adjudicación de tierras, de acuerdo a la capacidad de uso mayor de la tierra y a las necesidades socio-económicas del país, previo dictamen de las comisiones agrarias departamentales;
9. Promover la conciliación de conflictos emergentes de la posesión y del derecho de propiedad agraria;
10. Actualizar y mantener un registro sobre tierras distribuidas, sus beneficiarios y la disponibilidad de tierras fiscales. Esta información tendrá carácter público;
11. Coordinar sus actividades con las entidades públicas y privadas encargadas de dotar de infraestructura, de servicios básicos y de asistencia técnica a zonas de asentamientos humanos;
12. Certificar derechos existentes en tierras fiscales destinadas a la conservación, investigación, ecoturismo y aprovechamiento forestal; y
13. Otras que le asigne esta ley y su reglamento.

ARTICULO 19º (Estructura Orgánica).

El Instituto Nacional de Reforma Agraria tiene la siguiente estructura orgánica:

1. La Dirección Nacional;
2. Las Direcciones Departamentales; y,
3. Las Jefaturas Regionales.

ARTICULO 20º (Dirección Nacional).

I. La Dirección Nacional, como máximo nivel de autoridad institucional, es el órgano ejecutivo encargado de dirigir, coordinar y supervisar el cumplimiento de las atribuciones conferidas al Instituto Nacional de Reforma Agraria (INRA).

II. El Director Nacional del Instituto Nacional de Reforma Agraria será designado por el Presidente de la República, de terna aprobada por la Honorable Cámara de Diputados por dos tercios de votos de sus miembros presentes. Desempeñará sus funciones por un período personal e improrrogable de cinco (5) años, no pudiendo ser reelegido sino después de un período igual al ejercido. Sus atribuciones serán establecidas en el reglamento a esta ley.

III.Para ser Director Nacional se requiere:

1. Ser boliviano y ciudadano en ejercicio;
2. Tener grado académico a nivel de licenciatura con título en provisión nacional, haber ejercido su profesión con idoneidad durante cinco (5) años y tener experiencia en materia agraria; y,
3. No estar comprendido en las causales de incompatibilidad que señala la ley.

IV. Las resoluciones del Director Nacional admiten recurso de revocatoria ante la misma autoridad y recurso jerárquico ante el Ministro de Desarrollo Sostenible y Medio Ambiente.

ARTICULO 21º (Direcciones Departamentales).

I. Las direcciones departamentales son unidades desconcentradas del Instituto Nacional de Reforma Agraria (INRA) y realizarán sus actividades en coordinación con el órgano central. Sus atribuciones serán establecidas en el reglamento de esta ley.

II. Los directores departamentales serán designados por el Director Nacional, de ternas propuestas por las comisiones agrarias departamentales.

III. Para ser Director Departamental se requiere cumplir los requisitos establecidos para el Director Nacional.

IV. Las resoluciones de los directores departamentales podrán ser impugnadas en sede administrativa. Agotada la sede administrativa podrán ser impugnadas mediante proceso contencioso-administrativo ante el Tribunal Agrario Nacional, en el plazo de cuarenta y cinco (45) días perentorios computables desde la notificación con la resolución que agote la sede administrativa.

ARTICULO 22º (Jefaturas Regionales).

I. Conforme a las necesidades, en una o en varias provincias agrupadas en regiones, funcionarán jefaturas regionales, dependientes de las direcciones departamentales correspondientes. Sus atribuciones serán establecidas en el reglamento de esta ley.

II. Los Jefes Regionales serán designados por el Director Departamental.

III. Para ser designado Jefe Regional se requiere:

1. Ser boliviano y ciudadano en ejercicio;
2. Tener formación técnica y experiencia en materia agraria; y,
3. No estar comprendido en las causales de incompatibilidad que señala la ley.

ARTICULO 23º (Régimen Financiero).

Son fuentes de financiamiento del Instituto Nacional de Reforma Agraria:

1. Asignación presupuestaria del Tesoro General de la Nación;
2. Ingresos propios; y,
3. Otros que obtenga por donaciones, legados o empréstitos.

ANEXO C

SECCION III

ELABORACION DE PROYECTOS DE RESOLUCIONES FINALES DE SANEAMIENTO

ARTÍCULO 325° (Proyectos de Resoluciones).

I. Concluida la actividad del informe en conclusiones y con base en las sugerencias expuestas, en un plazo no mayor a quince (15) días calendario por polígono, los responsables de esta actividad elaboraran proyectos de resoluciones por cada proceso agrario titulado, en trámite o posesión, por organización social o por predio cuando corresponda, conjuntamente los planos prediales.

II. Los proyectos de resoluciones y las etapas precedentes de saneamiento serán objeto de aprobación por el Director Departamental competente, previa a su remisión a la Dirección Nacional.

CAPITULO V

ETAPA DE RESOLUCIONES Y TITULACION DEL PROCEDIMIENTO COMUN DE SANEAMIENTO

ARTÍCULO 326° (Actividades).

I. Esta etapa consiste en el desarrollo de un conjunto de actividades que se realizan en gabinete a partir de la recepción de los proyectos de resoluciones finales de saneamiento a la Dirección Nacional del Instituto nacional de reforma Agraria.

II. Comprende las siguientes actividades:

1. Firma de resoluciones y plazo de impugnación,
2. Titulación,
3. Registro en Derechos Reales y transferencia de información a las municipalidades.

ARTÍCULO 327° (Firma de Resoluciones y Notificación)

I. La firma de resoluciones administrativas no deberá exceder quince (15) días calendario de recepcionado el proyecto de resolución y sus antecedentes en la Dirección Nacional.

II. En el caso de resoluciones supremas serán remitidas en un plazo no mayor de tres (3) días calendario a la unidad competente de la Presidencia de la República, para la respectiva firma, computable desde la recepción del proyecto de resolución y sus antecedentes en la Dirección Nacional.

III. Firmadas que fueran las resoluciones finales de saneamiento, se procederá a la remisión de estas a las Direcciones Departamentales respectivas, para fines de notificación a las personas interesadas, a cuyo efecto se consideraran los domicilios individuales o comunes acreditados por las mismas.

ARTICULO 328° (Verificación de Interposición de Acciones Contencioso Administrativas).

Una vez sea notificada la resolución final de saneamiento, salvando el caso de acreditarse renuncias expresas al término de impugnación por las personas interesadas, se solicitara certificación o informe del Tribunal Agrario Nacional sobre la interposición de acciones contenciosas administrativas contra la resolución emitida.

ARTICULO 329° (Titulación).

I. Ejecutoriadas que fueran las resoluciones finales de saneamiento o si existiesen renuncias al término de impugnación, se remitirán antecedentes a la Unidad de Titulación de la Dirección Nacional del Instituto Nacional de Reforma Agraria para la emisión de Títulos Ejecutoriales.

II. El trámite de firma y refrenda, así como los alcances, reglas, forma y contenido para el otorgamiento, registro y entrega de Títulos Ejecutoriales, se sujetaran a lo previsto en este Reglamento.

ARTICULO 330° (Registros en Derechos Reales y Transferencia de Información a Municipalidades).

Consolidada la información en el Sistema Catastral se procederá al registro en Derechos Reales.

DOCUMENTOS