UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

SISTEMA INFORMÁTICO PARA LA GESTIÓN DE PROYECTOS

T E S I S
QUE PARA OBTENER EL TÍTULO DE:
INGENIERO EN COMPUTACIÓN

PRESENTA:
MARIO ALBERTO RAMÍREZ ORIHUELA

DIRECTOR:
ING. ORLANDO ZALDÍVAR ZAMORATEGUI

ÍNDICE

Agradecimientos

En primera instancia quiero agradecer a mis padres, por el amor y apoyo que siempre me han dado y por ello me siento muy afortunado. De igual manera por ser excelentes seres humanos y una fuente de admiración constante para mi, esperando retribuirles siguiendo su ejemplo.

A Ernesto, quien ha realizado un gran trabajo como mi hermano mayor, pues siempre me comparte sus experiencias de las cuales he aprendido mucho y sus consejos son siempre bien recibidos. Muchas gracias hermano.

A Sergio, quien a pesar de las situaciones no pudimos coincidir, pero constantemente tengo presente también quiero dedicar este logro. Gracias donde quiera que estés.

Asimismo, quiero agradecer a Carmen, primeramente por ser parte de mi vida y acompañarme todos estos años brindándome su amor y apoyo constante en todo lo que hago.

Agradezco enormemente a la Universidad Nacional Autónoma de México por haberme abierto sus puertas y mostrado un universo de conocimiento, el cual ha enriquecido mi espíritu y mente, por lo que me siento orgulloso de ser parte de ella.

De igual manera, quiero expresar mi agradecimiento a la Facultad de Ingeniería ya que gracias a la excelencia de sus profesores y programas de estudio pude llegar a ser el profesionista que soy ahora.

Mi más sincero agradecimiento al profesor Orlando Zaldívar Zamorategui por su invaluable ayuda, asesoría, recomendaciones y tiempo en la realización de este trabajo de tesis.

Así también un agradecimiento muy especial a mis sinodales, por el tiempo dedicado y valiosos consejos para el enriquecimiento de este trabajo de tesis.

A mis amigos por el ánimo y amistad que me han brindado les doy las gracias, ya que son una parte importante en mi vida.

Introducción

En los últimos años la tecnología computacional ha avanzado rápidamente con grandes progresos tanto en hardware como en software. Asimismo, las necesidades de las organizaciones en cuanto a sistemas de información demandan productos de software cada vez más grandes y complejos sin comprometer la calidad. En estos tiempos, donde la disponibilidad e integridad de la información es un factor crítico, es necesario contar con metodologías para tener sistemas de software confiables que sean entregados a tiempo y con costo reducido. El hecho de tener proyectos de software robustos implica hacer uso de la ingeniería de software para que proporcione procesos bien establecidos que controlen el desarrollo de los sistemas.

Dicho lo anterior, es claro que existe una brecha entre las necesidades de las organizaciones y el tiempo de respuesta y la calidad de los sistemas de información, ya que a lo largo de los años las organizaciones tienden a ser más integrales y globales, y dada la competitividad creciente actual en los mercados las compañías necesitan mecanismos que les ayuden a lograr sus objetivos y a la vez cuidar sus costos de operación. Debido a estas razones, es importante conocer y elegir la metodología de ingeniería de software adecuada que atienda dichas necesidades de manera oportuna y eficiente.

Actualmente, los sistemas de información son un factor clave para el cumplimiento de estos objetivos específicos y es evidente la evolución que han tenido en los últimos años. Es por ello que la implementación de las metodologías establece la base para que un proyecto de software sea exitoso.

Dentro de las metodologías de software se encuentra el modelo en espiral, el cual se estudiará en este trabajo, aplicándolo específicamente al desarrollo de aplicaciones web.

A partir de la delimitación del objeto de estudio, resulta interesante analizar ¿en qué consiste el modelo en espiral?, ¿cuándo es oportuno utilizarlo? y en función a esto ¿qué tan bien responde esta metodología para proyectos grandes o pequeños?

Otra pregunta importante sería el saber ¿cuál es la relevancia o beneficio de incorporar la gestión de riesgos en un proyecto de software? y ¿cómo se utilizaría para un proceso de mantenimiento?, así como también conocer ¿cuál es el beneficio para el cliente y el desarrollador al utilizar esta metodología? Finalmente ¿esta metodología responde a los requerimientos hechos por el cliente?

En el presente trabajo de tesis se pretende mostrar que a partir de los conocimientos académicos adquiridos ha sido posible tener un crecimiento profesional en el campo laboral, lo cual ha permitido una aportación relevante en proyectos de ingeniería de software tanto en el aspecto técnico como en el de planeación de proyectos de tecnologías de información. Asimismo, a partir de las preguntas anteriormente formuladas, surge el interés por analizar la problemática que se desarrollará en este trabajo utilizando la metodología en espiral. Se estudiará la forma en que el modelo en espiral se comporta en proyectos de desarrollo de aplicaciones web reales para obtener las respuestas a nuestras preguntas y generar las conclusiones pertinentes respecto a la metodología.

Una vez enunciados los objetivos de este trabajo, se plantearán las hipótesis correspondientes.

El modelo en espiral es un acercamiento más conveniente y eficiente para proyectos de desarrollo de software grandes y complejos, pues la incorporación de gestión de riesgos y el desarrollo incremental genera productos mayormente apegados a la especificación de las necesidades a atender. Asimismo, el modelo provee de respuestas más rápidas y visibles

para el cliente que otros modelos, dado que las recurrentes iteraciones promueven constante comunicación y revisión (principalmente en la fase de evaluación), lo que genera confianza de que lo que se está haciendo es lo que se ha pedido.

Por otro lado, dentro del equipo de desarrollo es necesario contar con recursos altamente calificados, tanto en conocimiento técnico como en habilidades de planeación, análisis de riesgos y mitigación, así como también en manejo de la relación con el cliente.

Esta metodología no está totalmente centrada en los documentos y especificaciones de requerimientos como el modelo en cascada, el cual tiende a ser un modelo un tanto rígido para el desarrollador y el cliente, pues no fácilmente permite realizar cambios sobre la marcha respecto al alcance y actividades del plan de trabajo del proyecto. En cambio, el modelo en espiral combina las mejores prácticas de la estructura del modelo en cascada agregando flexibilidad a cambios generados sobre la marcha e incorporando los beneficios de los prototipos para acelerar el proceso de desarrollo. Estos acercamientos hacen que el cliente esté más involucrado en el proceso desarrollo y sea más transparente, logrando tener un producto lo más apegado a sus necesidades.

La metodología de este trabajo de tesis fue de gran relevancia para el desarrollo del Sistema Informático para la Gestión de Proyectos (SIGP), pues de esta manera el componente de software estuvo apegado a los requerimientos del cliente, para así demostrar las hipótesis antes enunciadas.

La metodología del modelo en espiral de desarrollo de software, bajo la cual se apegó el proyecto de desarrollo, en conjunto con los conocimientos técnicos, gestión de requerimientos, administración de riesgos, utilización de UML y gestión de calidad, formaron las bases necesarias para llevar a cabo un proyecto de esta magnitud y el análisis

de la situación previa y posterior nos brinda la oportunidad de comparar los datos y generar las conclusiones pertinentes después de que el sistema ha sido instalado y puesto en marcha.

Para plantear el presente trabajo, en el primer capítulo, denominado Marco teórico, se presentan inicialmente conceptos y fundamentos relacionados a la ingeniería de software y sus paradigmas, haciendo especial énfasis en el modelo en espiral, así como también conceptos de programación orientada a objetos y herramientas utilizadas para el desarrollo de software como UML. También se abordan temas de mantenimiento de sistemas y la metodología de calidad Six Sigma.

Terminada la formulación del marco teórico se continúa con el segundo capítulo, titulado como Planteamiento del problema, donde se abordan los antecedentes y problemática de un proyecto especifico en el que se participó activamente y que hizo uso de los conceptos y metodologías mencionados en el primer capítulo, con el objetivo de sentar las bases del proyecto antes de su desarrollo.

En el tercer capítulo, llamado Desarrollo del SIGP, se explica el propósito y objetivos del proyecto tomando en consideración las expectativas que el cliente tiene y se describe de manera detallada el desarrollo bajo la metodología en espiral de ingeniería de software.

El cuarto capítulo, referido a Desarrollo basado en casos de uso, muestra algunos ejemplos de casos de uso del proyecto, presentándolos desde su concepción general y preámbulo funcional hasta la especificación detallada del requerimiento y diagramas UML del mismo. Al final del capítulo se describen los entregables generados por estos casos de uso.

En el quinto capítulo, titulado como Mantenimiento del sistema, se describe la propuesta de implementación del plan de mantenimiento del sistema ya que ha sido terminado e

instalado. Se inicia el capítulo con los preparativos previos a la aplicación del plan de mantenimiento, pasando por su estructuración para, posteriormente, explicar el modelo de soporte y sus correspondientes áreas de servicio.

Finalmente, en el sexto capítulo, denominado como Resultados, impacto y conclusiones, se documentan los resultados después de la implementación del proyecto con un comparativo entre la situación previa y posterior al sistema para, finalmente, proporcionar las conclusiones derivadas de este trabajo de tesis.

Capítulo 1

Marco teórico

1.1 Ingeniería de software

La ingeniería de software es una disciplina de la ingeniería que comprende todos los aspectos de la producción de software, desde las etapas iniciales de la especificación del sistema, hasta el mantenimiento de éste después de que se utiliza (Sommerville, 2005:6).

Actualmente, existen varias metodologías para abordar un proyecto de software que han sido generadas a partir de mejoras continuas a lo largo de los años para que el proceso de desarrollo de software se adapte a las necesidades actuales.

La finalidad de apegarse a una metodología o proceso de desarrollo de software es determinar el orden de las etapas involucradas en el desarrollo y evolución del software y establecer los criterios de transición para el progreso de una etapa a otra. Incluye también un criterio de culminación de la etapa actual más los criterios de elección y entrada para la siguiente etapa.

Las metodologías de desarrollo de software son importantes, principalmente porque proporcionan una guía del orden (fases, incrementos, prototipos, validación de tareas, etc.) que un proyecto debe seguir para realizar sus tareas más importantes. Muchos proyectos de software han enfrentado dificultades porque abordan varias fases del desarrollo y evolución de la manera equivocada.

1.2 Evolución de las metodologías de software

A continuación se muestra una breve descripción de algunas metodologías de software existentes que forman parte de la evolución de las mismas.

1.2.1 Codificar- Corregir

Esta metodología básicamente consiste de dos pasos:

- Escribir porciones de código
- Resolver los problemas en el código.

Las etapas de diseño, pruebas y mantenimiento eran consideradas después. A lo largo de su utilización se identificaron varios problemas que dificultaban un desarrollo fluido y exitoso. En realidad, era prácticamente inevitable tener errores en el producto desarrollado que debían ser corregidos antes de su entrega. Algunos ejemplos se enuncian a continuación:

- Después de varias correcciones el código perdía su estructura
- Las expectativas del cliente no se satisfacían, por lo que se rechazaba el producto y se aplicaba mayor esfuerzo del necesario
- Los costos de corregir el código eran muy altos, por la poca preparación de las pruebas

1.2.2 Modelo en cascada

Para resolver los problemas que se presentaban en desarrollos de sistemas en los años 50´s se dio reconocimiento a los problemas y se desarrolló un modelo de etapas. La idea detrás de este modelo estipulaba que el software debía ser desarrollado en varias etapas sucesivas (plan operacional, especificaciones operacionales, especificaciones de código, codificación, parámetros de pruebas, pruebas de ensamblado y evaluación del sistema). Finalmente, en los años 70´s se tenía ya establecido el modelo de cascada que ofrecía las siguientes mejoras:

- Reconocimiento de los ciclos de retroalimentación entre las etapas, y una guía para limitar los mismos entre etapas sucesivas para minimizar trabajo duplicado o innecesario y costoso involucrado en la retroalimentación en diversas etapas.

- Mejor organización al momento de desarrollar un producto de software, pues la definición de sus etapas ayuda a generar planes de trabajo más específicos.

- Resulta sencillo para los usuarios comprender cada una de las fases del modelo, lo cual mejora la comunicación con el equipo de desarrollo del sistema.

El modelo en cascada se convirtió en la base de varios estándares de software en el sector público y privado. Algunas de sus dificultades iniciales han sido atendidas extendiendo el modelo a enfoques ligeramente mas flexibles en cuanto a las transiciones de las etapas que define. Sin embargo, ciertas dificultades no se han podido resolver mediante este modelo. La principal dificultad ha sido su énfasis en documentos completamente elaborados para las etapas iniciales de requerimientos y diseño. Para ciertos tipos de software tales como compiladores o sistemas operativos este modelo funciona muy bien. No obstante, para otras clases de software, particularmente para aplicaciones que ofrecen constante interacción con el usuario final, esta rigidez es poco viable ya que los requerimientos del producto tienden a cambiar constantemente.

1.2.3 Modelo evolutivo

Las etapas del modelo evolutivo consisten en expandir incrementos de un producto operacional de software, con las directivas de evolución siendo determinadas por la experiencia operacional.

El modelo evolutivo está idealmente relacionado a una aplicación de lenguaje de cuarta generación y se apega a situaciones donde el usuario menciona 'no puedo decirte qué es lo que quiero, pero lo sabré cuando lo vea'. Provee una ágil capacidad operacional inicial y establece las bases para determinar mejoras subsecuentes al producto. Pero este modelo también sufre ciertas dificultades. Generalmente es difícil distinguirlo contra el modelo de codificar y corregir en él que la falta de planeación fue la motivación inicial del surgimiento del modelo en cascada. Regularmente se basa en la suposición poco realista de que el sistema será lo suficientemente flexible para agregar rutas evolutivas no planeadas. Este supuesto es inválido en las siguientes condiciones:

- Circunstancias en las que varias aplicaciones evolutivas independientes deben ser al final integradas estrechamente

- Casos en que correcciones de software temporales se materializan y transforman en restricciones que ya no se pueden modificar en el proceso evolutivo.

- Cuando el nuevo software está incrementalmente reemplazando a un software existente que carece de organización y modularidad que permita hacer que cada parte del sistema evolucione.

1.2.4 El modelo de transformación

Este modelo supone la existencia de la capacidad de convertir automáticamente una especificación formal de un producto de software en un programa que satisfaga esa especificación. Por tanto, los pasos que se establecen en este modelo son los siguientes:

- Tener una especificación formal derivada del mejor entendimiento del producto deseado.

- Transformación automática de la especificación a código
- Un ciclo iterativo, de ser necesario, para mejorar el desempeño del código resultante mediante la optimización del sistema en transformación.
- Ejercicio del código resultante.
- Un ciclo iterativo independiente para ajustar la especificación basada en la experiencia operativa y optimizar el sistema de software.

1.2.5 El modelo en espiral

El modelo en espiral, originalmente propuesto por Bohem, es un modelo evolutivo de proceso de software que combina la naturaleza iterativa de los prototipos con los aspectos controlados y sistemáticos del modelo en cascada. Principalmente está intencionado para su aplicación en proyectos largos, complicados y costosos. Provee el potencial de un desarrollo rápido de versiones incrementales de software. Al utilizar el modelo en espiral, el software es desarrollado en series de liberaciones incrementales. Durante las iteraciones iniciales, la liberación incremental puede ser un modelo en papel o un prototipo y va creciendo hasta cubrir las funcionalidades requeridas.

Asimismo, el modelo en espiral puede reunir prácticas de modelos anteriores y proveer una guía mediante la combinación de las mejores prácticas de las mismas.

Este modelo ha evolucionado a través de los años, basado en experiencias para perfeccionar el modelo en cascada en casos tales como grandes proyectos de software. La Figura 1.1 representa un diagrama del modelo en espiral. Como se muestra en la figura, la dimensión radial de la espiral representa el costo acumulado para terminar los pasos del desarrollo. La dimensión angular representa el progreso para terminar cada ciclo de la espiral.

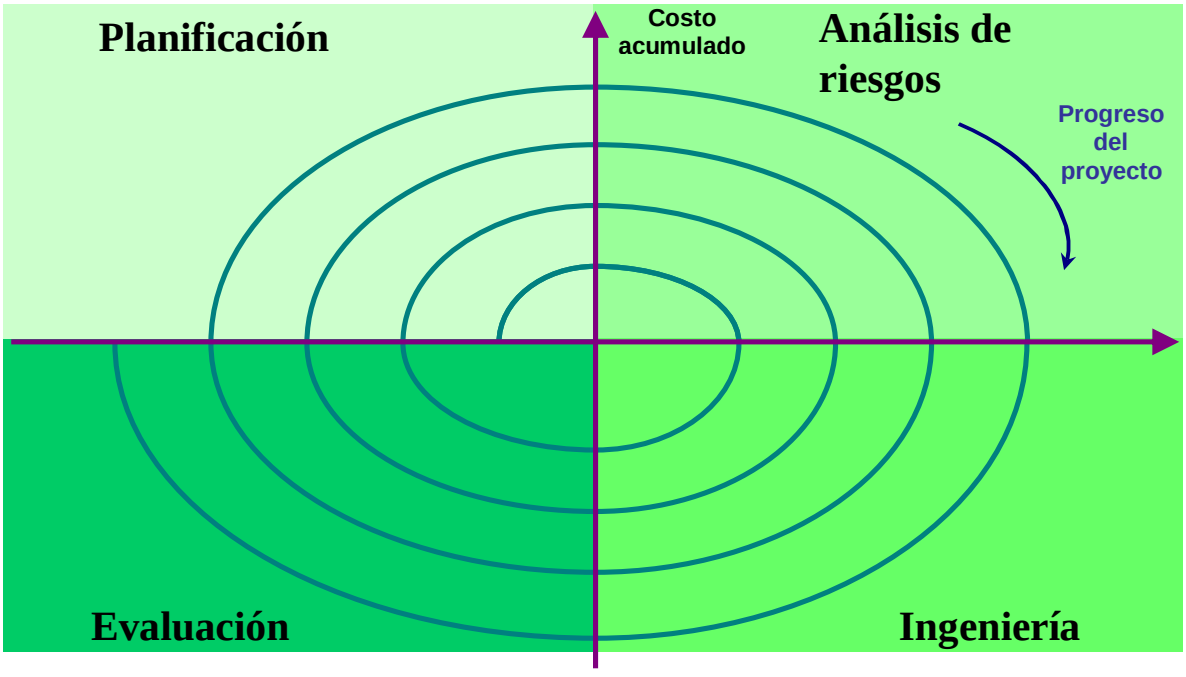


Figura 1.1 Diagrama del modelo en espiral

Cada una de las regiones está compuesta por un conjunto de tareas, las cuales son adaptadas a las características del proyecto en cuestión. Para proyectos pequeños el número de tareas es menor y por lo tanto no requieren demasiada formalidad. En contraste, refiriéndonos a proyectos grandes y críticos, cada región del modelo contiene más tareas que son definidas para lograr un mayor grado de formalidad y detalle.

Conforme el proceso evolutivo inicia, el equipo involucrado en la ingeniería de software se mueve de manera espiral en sentido a las manecillas del reloj, iniciando cerca del centro. El primer circuito de la espiral puede resultar en el desarrollo de la especificación del producto, los circuitos subsecuentes pueden ser usados para desarrollar un prototipo, para entonces tener versiones más sofisticadas del producto de software de manera progresiva. Cada paso a través de la región de planeación para determinar los objetivos, alternativas y restricciones resulta en ajustes en el plan del proyecto, pues las acciones a tomar para atender los riesgos o situaciones particulares repercuten en el plan inicial y es necesario

revisar constantemente el plan del proyecto. Tanto el calendario como el costo son modificados, dependiendo de la retroalimentación obtenida con el cliente. Adicionalmente el líder de proyecto ajusta el número planeado de iteraciones requeridas para que el producto esté terminado.

1.2.6 Un ciclo típico de la espiral

Cada ciclo de la espiral inicia con la identificación de los siguientes elementos:

- Los objetivos de la porción del producto que está siendo elaborada (desempeño, funcionalidad, alcance, etc.)
- Las alternativas que existen para la implementación de esta porción del producto
- Las restricciones que tiene cada alternativa para que sea aplicada (costo, calendario, recursos, etc.)

El siguiente paso consiste en evaluar las alternativas en cuanto a los objetivos y restricciones. Frecuentemente, este proceso identificará áreas de incertidumbre que son fuentes potenciales de riesgo.

Posteriormente, la siguiente etapa consistirá en la formulación de una estrategia efectiva en costo para implementar o desarrollar los objetivos definidos. Asimismo, se tiene que prestar especial atención a las actividades que atiendan las fuentes de riesgo identificadas en el paso anterior. Algunas técnicas que pueden ser utilizadas incluyen simulaciones, prototipos, benchmarking, cuestionarios, etc.

Una vez que se evaluaron los riesgos y desarrollaron los elementos necesarios, el siguiente paso está determinado por los riesgos que no se hayan podido atender en el paso anterior.

Para ello, se analizan y planean las siguientes fases que serán parte de una nueva iteración en la espiral.

Durante las iteraciones posteriores, versiones más completas del sistema son producidas. El modelo en espiral está dividido en un determinado número de actividades del esquema, dependiendo el detalle que se quiera dar a la espiral. Algunas de las actividades más comunes son:

- Planificación: Actividades requeridas para definir recursos, fechas de entrega y otra información relacionada con el proyecto.
- Análisis de riesgo: Actividades necesarias para evaluar riesgos tanto técnicos como administrativos del proyecto
- Ingeniería: Actividades necesarias para construir una o más representaciones de la aplicación
- Evaluación del cliente: Actividades necesarias para obtener la retroalimentación del cliente basada en la evaluación de las representaciones de software creadas e implementadas durante la etapa de ingeniería.

Hay 4 preguntas que surgen a causa del modelo en espiral.

1) ¿Cómo se empieza la espiral?
2) ¿Cómo salir de la espiral cuando es apropiado terminar tempranamente el proyecto?
3) ¿Por qué la espiral termina tan abruptamente?
4) ¿Qué sucede con la mejora del software (o mantenimiento)?

Las respuestas a estas preguntas involucran hacer la observación de que el modelo en espiral es aplicable tanto a desarrollos como a mejoras de software. En cualquiera de los

casos, la espiral es iniciada bajo la hipótesis de que una misión operacional en particular (o grupo de misiones operacionales) pueden ser mejoradas mediante un esfuerzo orientado al software. Entonces, el modelo en espiral involucra probar esta hipótesis: en cualquier momento, si la hipótesis falla la prueba (por ejemplo, si los retrasos causan que un producto de software pierda su periodo útil, o si un producto comercial superior es lanzado), el modelo en espiral es terminado. Por otro lado, termina con la instalación de un producto de software, ya sea nuevo o modificado, y la hipótesis es probada mediante la observación del efecto producido en la misión operacional. Usualmente, la experiencia con la misión operacional resulta en otra hipótesis relacionada a mejoras del software y una nueva espiral de mantenimiento es iniciada para probar la hipótesis. La iniciación, terminación e iteración de las tareas y productos de ciclos previos son implícitamente definidos en el modelo en espiral.

A diferencia de los modelos de proceso clásicos que terminan cuando el software es entregado, el modelo en espiral puede ser adaptado para su aplicación a lo largo del ciclo de vida del producto de software. El centro del modelo en espiral puede ser usado para representar el punto inicial para diferentes tipos de proyecto. Un “proyecto de desarrollo conceptual” inicia cerca del centro de la espiral y continuará hasta que el proyecto conceptual sea completado. Si el concepto debe ser desarrollado en un producto real, el proceso continúa al siguiente ciclo (nuevo punto de entrada para el proyecto de desarrollo del producto) y un “nuevo proyecto de desarrollo” es iniciado. El nuevo producto evolucionará a través de un número de iteraciones alrededor de la espiral, siguiendo la ruta que limita la región que de alguna manera es más clara que el centro. En esencia, la espiral, cuando es caracterizada de esta manera, permanece en operación hasta que el producto es

retirado. Hay veces que el proceso permanece inactivo, pero en el momento en que un cambio es iniciado, el proceso es iniciado en el punto de entrada apropiado (por ejemplo, cuando se necesita una mejora).

El modelo en espiral es un enfoque realista para desarrollar sistemas de gran escala. Dado que el software evoluciona conforme el proceso avanza, el desarrollador y el cliente tienen un mejor entendimiento del proyecto y toman acciones ante riesgos en cada nivel de la evolución del sistema. El modelo en espiral utiliza los prototipos como un mecanismo para reducir riesgos pero, más importante, para permitir al desarrollador aplicar el acercamiento prototipado en cualquier etapa de la evaluación del producto. Mantiene el enfoque sistemático sugerido en el ciclo de vida clásico, pero lo incorpora en un esquema iterativo que refleja de mejor manera el mundo real. El modelo en espiral demanda consideraciones de riesgos técnicos en todas las etapas del proyecto, que al ser aplicadas apropiadamente, deberá reducir riesgos antes de que se vuelvan problemas.

Pero como en otros paradigmas, el modelo en espiral no es una panacea. Puede ser complicado convencer al cliente (particularmente en situaciones de contratos) que el enfoque evolutivo es controlable. Demanda amplia experiencia en la evaluación de riesgos y depende de la misma para lograr el éxito. Finalmente, el modelo no ha sido utilizado tanto como los modelos lineales secuenciales o de prototipos. Tomará varios años antes de que la eficacia de este importante paradigma pueda ser determinada con absoluta certeza.

1.3 Mantenimiento de software

1.3.1 Definición y relevancia del mantenimiento de software

Una vez que el sistema se ha instalado se necesitan acciones para que el funcionamiento del mismo sea como se espera. El mantenimiento involucra tanto procesos de mejora y

optimización, como acciones correctivas para eliminar defectos del sistema. Es una de las consideraciones más relevantes de un proyecto de software, pues una vez que el mismo ha sido entregado se deben contemplar las modificaciones necesarias para corregir errores, mejorar el desempeño u otros atributos o adaptar el producto a nuevas necesidades y demandas del usuario.

Mantenimiento correctivo. Incluso cuando las mejores metodologías de calidad son utilizadas en el desarrollo de software no estamos exentos de tener algún error después de la instalación del sistema. En el mantenimiento correctivo se hacen modificaciones al software para corregir defectos que no fueron identificados durante el desarrollo del sistema y que el funcionamiento sea el que se espera.

Mantenimiento adaptativo. Durante el ciclo de vida del sistema es común que elementos del entorno (sistema operativo, reglas de negocio, hardware, etc.) cambien. El mantenimiento adaptativo se refiere a las actividades necesarias para que el sistema se adapte a esos cambios.

Mantenimiento perfectivo (mejoras). Conforme el sistema es utilizado por los usuarios es probable que se identifiquen nuevas funcionalidades que pueden ser incorporadas al sistema para que los beneficios del producto de software sean mayores. El mantenimiento perfectivo se encarga de extender el sistema a nuevas capacidades de acuerdo a la especificación del cliente.

Mantenimiento preventivo**.** Los productos de software tienden a deteriorarse después de que se hacen varias modificaciones probablemente derivadas por los tipos de mantenimiento anteriormente mencionados. Por lo tanto, se necesitan tomar acciones para reducir riesgos de falla en el futuro. El mantenimiento preventivo se encarga de modificar el sistema para que sea más fácil corregirlo, adaptarlo o mejorarlo.

1.3.2 Los procesos de mantenimiento de software

El mantenimiento de software involucra varias actividades a lo largo de la etapa operativa del producto de software, las cuales se agrupan en 6 procesos descritos a continuación.

1.3.2.1 Implementación del proceso

Para realizar la implementación del proceso de mantenimiento de software, el equipo que brindará el mantenimiento del producto genera el plan y los procedimientos a ser ejecutados durante el proceso de mantenimiento de software. Es importante resaltar que el plan de mantenimiento y el plan de desarrollo de software son generados de manera paralela. Asimismo, se establecen las formas de comunicación que se utilizarán con el usuario cuando el mantenimiento sea requerido. Por otra parte, el equipo de mantenimiento debe contar con toda la documentación asociada al sistema, la cual fungirá como la base de conocimiento técnico y funcional, necesario para brindar un mantenimiento apropiado.

La organización y el equipo de mantenimiento en conjunto definen el concepto de mantenimiento, para así tener un claro alcance del mismo.

Entre las tareas que el equipo de mantenimiento debe realizar se encuentran:

- Desarrollar los planes y procedimientos de mantenimiento
- Establecer procedimientos para peticiones de cambio del sistema

- Establecer procedimientos para reportar problemas
- Implementar la gestión de la configuración del sistema
- Estimar costos de mantenimiento

1.3.2.2 Análisis del problema y modificaciones

Este análisis se orienta a entender el problema, desarrollar una solución y obtener la aprobación para poder desarrollarla. Para ello se llevan a cabo las siguientes actividades:

- Analizar el problema encontrado o modificación solicitada para conocer el impacto, tanto en la organización, como en el sistema existente. En esta actividad se definen características, tales como el tipo de cambio, alcance y criticidad.
- Replicar el problema, ya sea con ayuda del usuario o por cuenta propia del equipo de mantenimiento. Implica contar con una estrategia de pruebas para replicar el problema, tener un ambiente con la misma versión de software y documentar los resultados.
- Generar y documentar opciones para corregir el problema o implementar la modificación solicitada, incluyendo estimaciones y riesgos por cada una de las opciones.
- Obtener la aprobación para la opción elegida por el cliente. De esta manera se tendrá definida la solución específica elegida por el cliente.

1.3.2.3 Implementación de la modificación

Durante este proceso, el equipo de mantenimiento y soporte de la aplicación desarrolla y prueba la corrección o modificación siguiendo la opción aprobada por el cliente apoyándose en la documentación de la misma. Se realizan las siguientes actividades:

- Actualización de los planes y procedimientos de pruebas
- Actualización de la documentación del sistema
- Modificación del código fuente
- Se genera un informe de resultados

1.3.2.4 Revisión y aceptación de la modificación

La revisión y aceptación de la modificación se utiliza para asegurar que las modificaciones se hicieron de forma correcta y cubrieron la necesidad o problema reportados. Entre las actividades para la revisión del mantenimiento se encuentran:

- Verificar que se cumplen los estándares de codificación
- Verificar que sólo se modificaron los componentes necesarios
- Verificar que la integración de los componentes modificados con el resto de los componentes sea correcta
- Comprobar que la documentación fue actualizada
- Ejecutar pruebas y documentar los resultados de las mismas

Una vez que la revisión ha terminado el mantenedor deberá obtener la aprobación para terminar satisfactoriamente el mantenimiento.

1.3.2.5 Migración

El proceso de migración es aplicado cuando se necesita que el sistema sea ejecutado en entornos diferentes, por tanto, el mantenedor necesita determinar las acciones necesarias para llevarla a cabo y generar la documentación relacionada. De esta manera se crea un plan de migración con las siguientes actividades:

- Análisis de requerimientos y definición de la migración

- Identificación de impacto, riesgos y esfuerzo de la migración
- Desarrollo de herramientas necesarias para efectuar la migración
- Conversión de datos y productos de software
- Ejecución de la migración
- Verificación de la migración por medio de pruebas
- Soporte para el entorno previo

1.3.2.6 Retiro del producto

El retiro del producto de software se lleva a cabo una vez que su periodo útil ha finalizado y conlleva a un nuevo análisis con enfoque costo-beneficio para elegir alguna de las siguientes opciones:

- Conservar el software obsoleto
- Cambiar de tecnología a través de un nuevo producto de software
- Desarrollar un nuevo producto de software

Después de tomar la decisión, se genera un plan de retiro con las siguientes actividades:

- Cese total o parcial del soporte del software
- Archivar el software y datos
- Definir la transición al nuevo producto de software
- Determinar el impacto del retiro del software
- Retirar el software

En la figura 1.2 se muestra la representación gráfica de los procesos de mantenimiento de software anteriormente descritos.

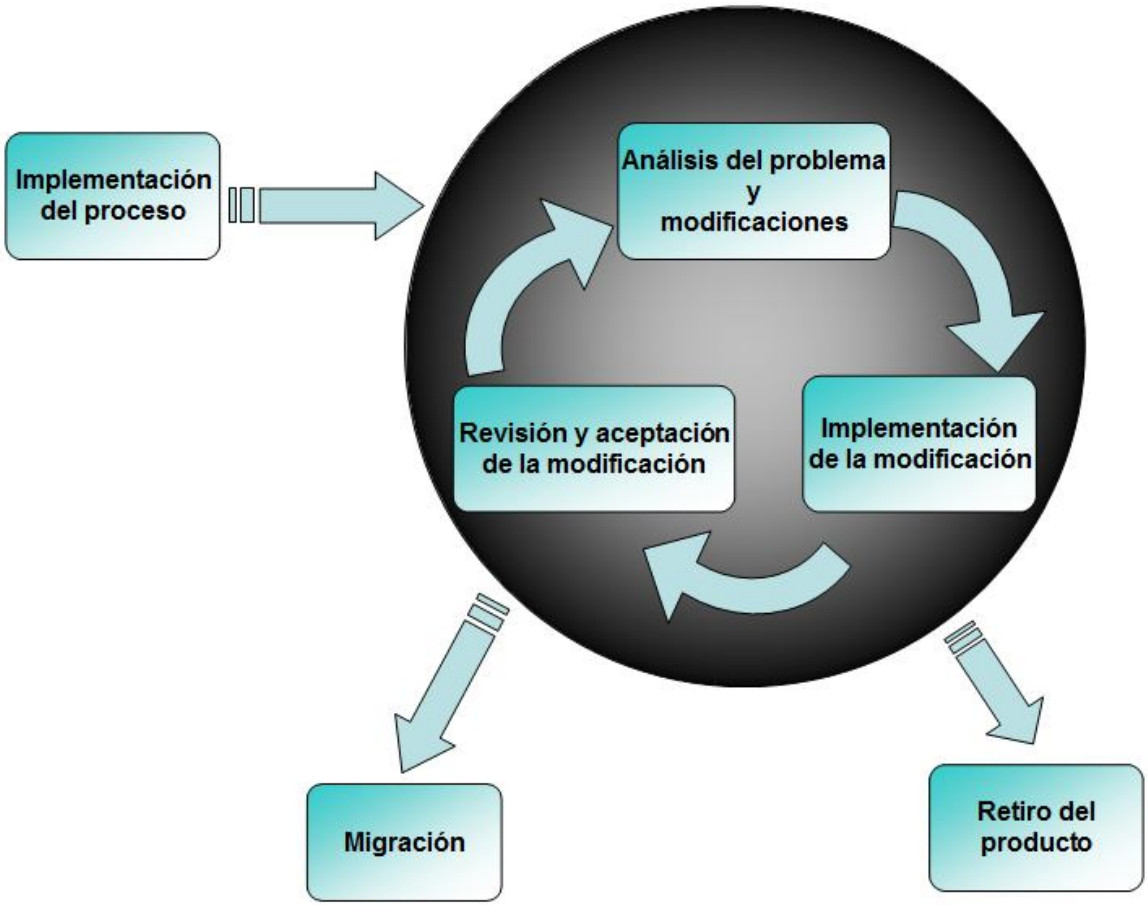


Figura 1.2 Procesos del mantenimiento de software

1.4 UML

Describimos de manera general el término UML (Unified Modeling Language) como un lenguaje utilizado para visualizar, especificar, construir y documentar los artefactos de un sistema de software. Se trata de un lenguaje gráfico orientado a apoyar todas las fases de desarrollo de software, desde la especificación y análisis de requerimientos hasta la construcción e instalación de los mismos.

La idea central detrás de la utilización de UML es capturar los detalles significativos de un sistema, de manera que el problema sea claramente comprendido, se desarrolle una solución al respecto y su implementación se identifique y construya de manera clara.

El lenguaje UML no sólo proporciona una notación para los elementos de construcción del sistema, también permite expresar relaciones complejas entre dichos elementos. Una relación puede ser estática, si se trata de conceptos de herencia entre un par de clases, interfaces implementadas o dependencias con otra clase. Por otro lado, existen relaciones dinámicas cuando describen el comportamiento del sistema, por ejemplo, cuando representan los intercambios de mensajes entre clases o el flujo de control del sistema.

A continuación se enlistan los tipos de diagramas proporcionados por el lenguaje UML.

- Diagrama de casos de uso. Son utilizados para representar gráficamente un comportamiento, mediante una secuencia de interacciones entre los actores y las funciones de un sistema. Debe capturar de manera precisa los requerimientos desde la perspectiva del usuario. En la figura 1.3 se muestra un ejemplo correspondiente a este tipo de diagramas.

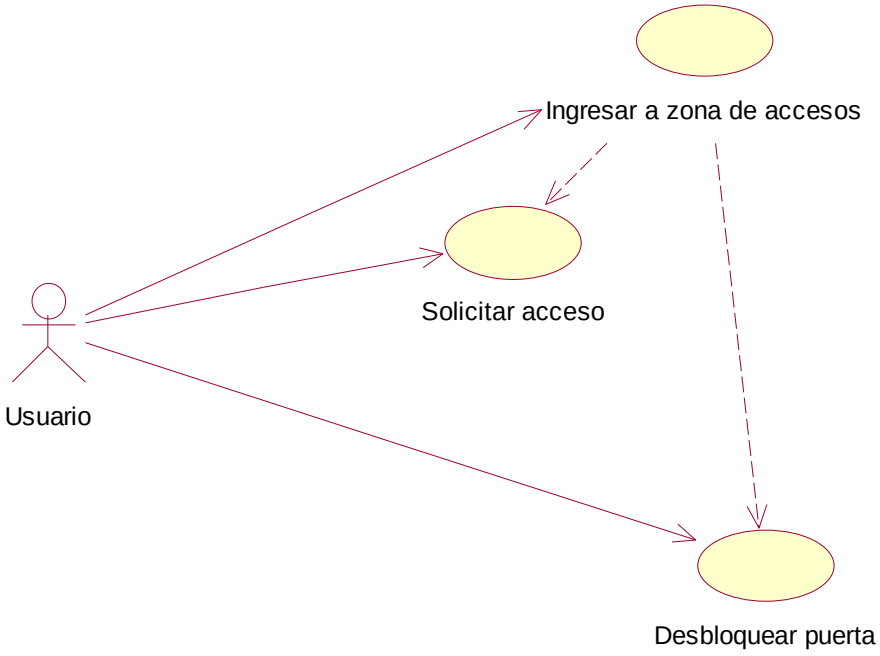


Figura 1.3 Ejemplo de diagrama de caso de uso

- Diagrama de clases. Muestra las relaciones estáticas existentes entre un grupo dado de clases e interfaces del sistema. Las relaciones más comunes representadas con este diagrama son herencia y dependencia de clases, tal y como se muestra en la figura 1.4.

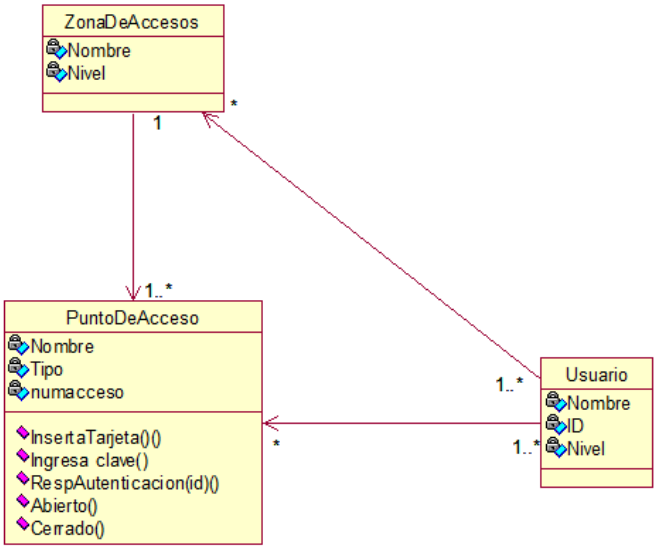


Figura 1.4 Ejemplo de diagrama de clases

- Diagrama de objetos. Brinda una vista instantánea de las relaciones que existen entre las instancias de clase en un determinado momento. Es de utilidad para capturar e ilustrar, de forma estática, relaciones complejas y dinámicas dentro del sistema como se aprecia en la figura 1.5.

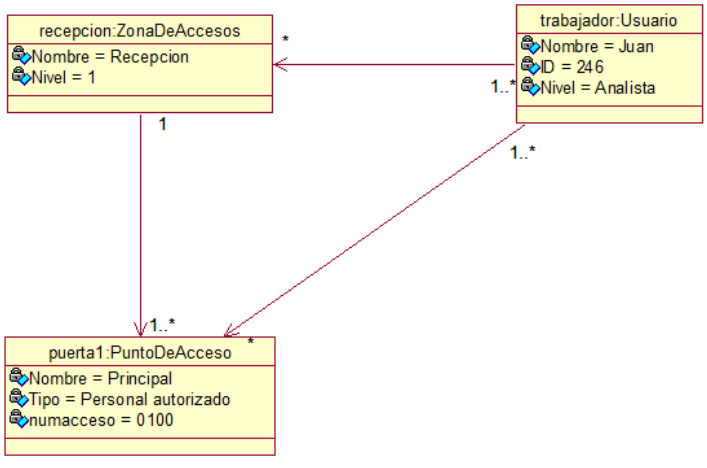


Figura 1.5 Ejemplo de diagrama de objetos

- Diagrama de estados. Excelente para ejemplificar el comportamiento dinámico de un sistema. Son aplicables a sistemas u objetos reactivos u orientados a eventos donde el orden de ejecución es importante como se ilustra en la figura 1.6.

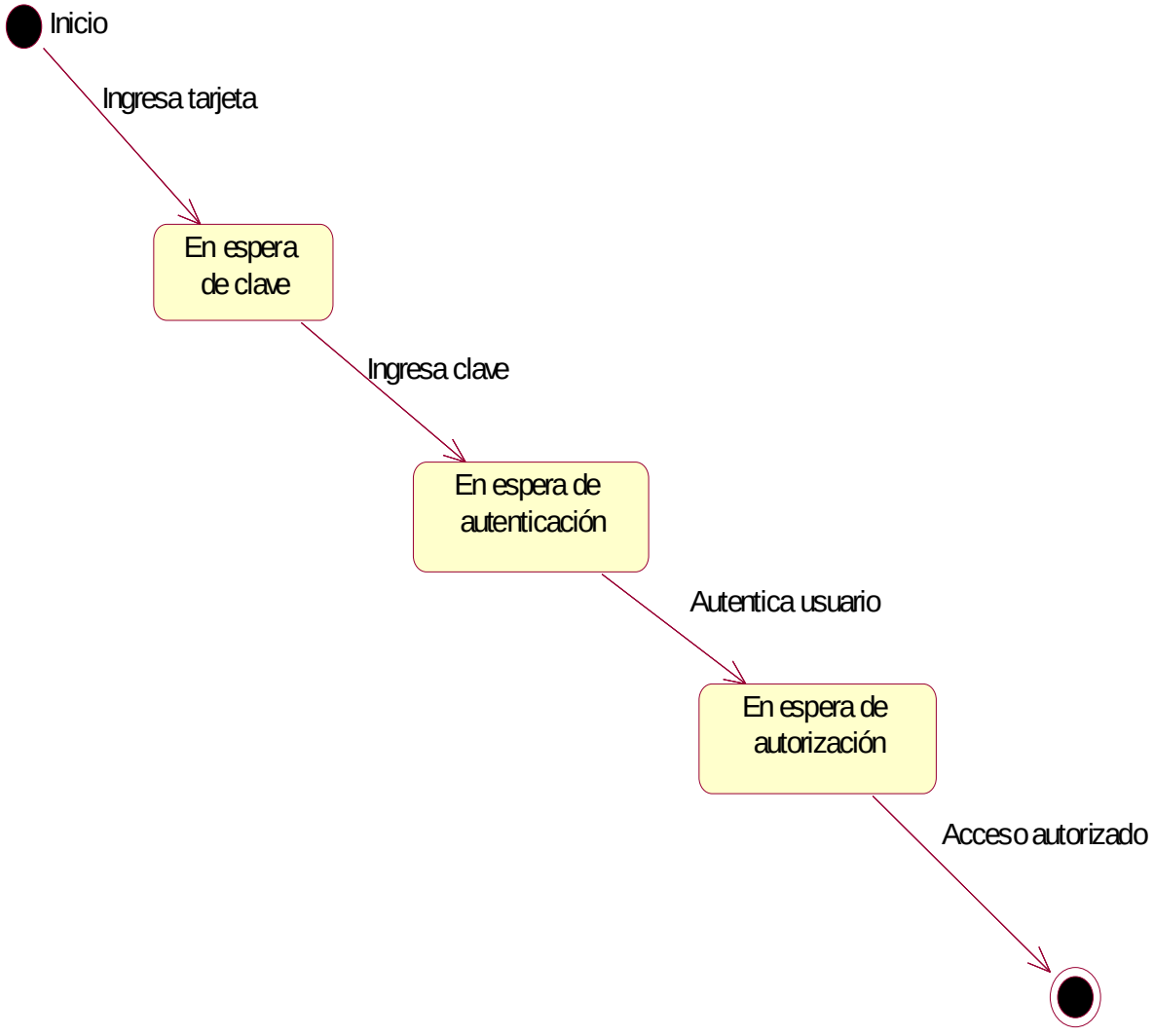


Figura 1.6 Ejemplo de diagrama de estados

- Diagrama de actividades. Son una extensión del diagrama de estados, pero orientado a flujos de trabajo dentro del sistema desde el punto de vista funcional. La figura 1.7 ejemplifica este tipo de diagramas.

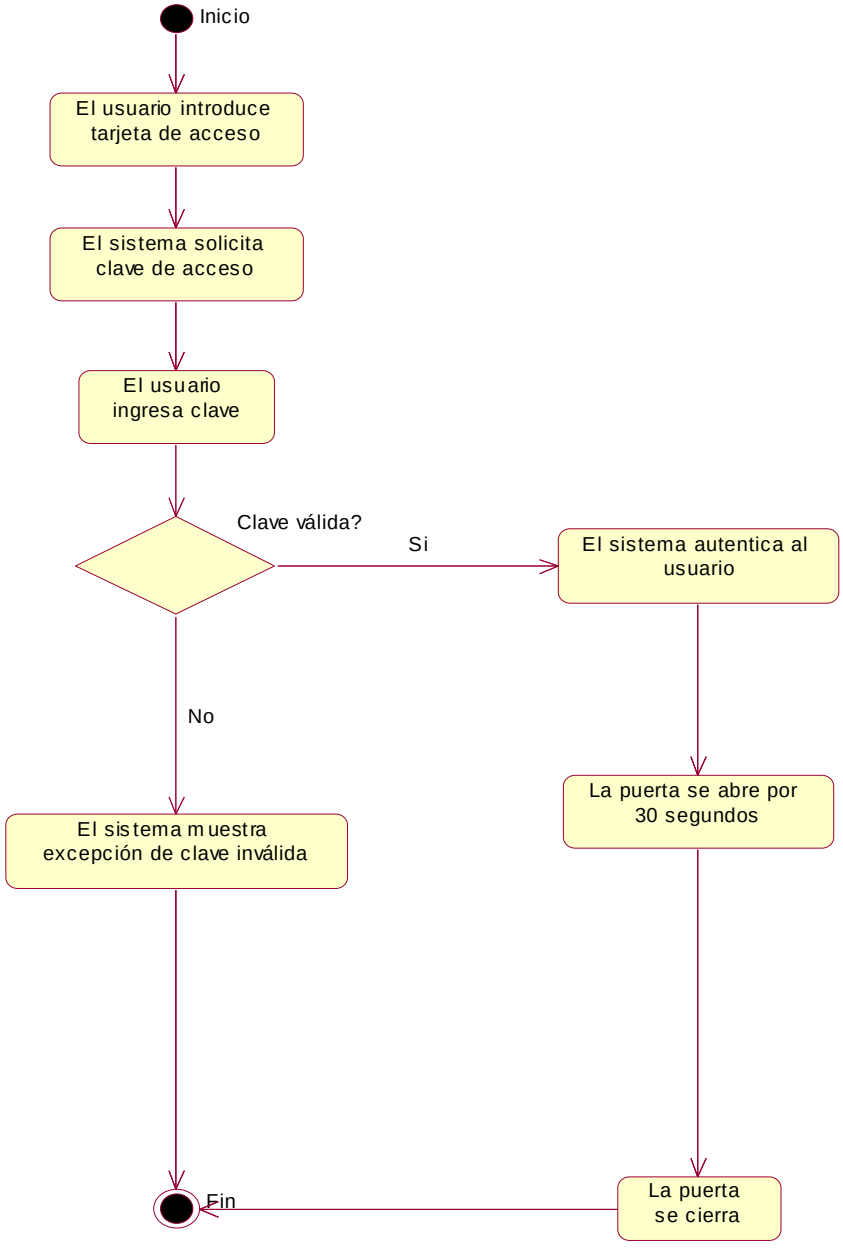


Figura 1.7 Ejemplo de diagrama de actividades

- Diagrama de secuencias. Son utilizados para modelar el intercambio de mensajes entre los objetos del sistema. También clarifican el tiempo relativo de los mensajes. La figura 1.8 ejemplifica un diagrama de secuencias para un sistema de control de accesos.

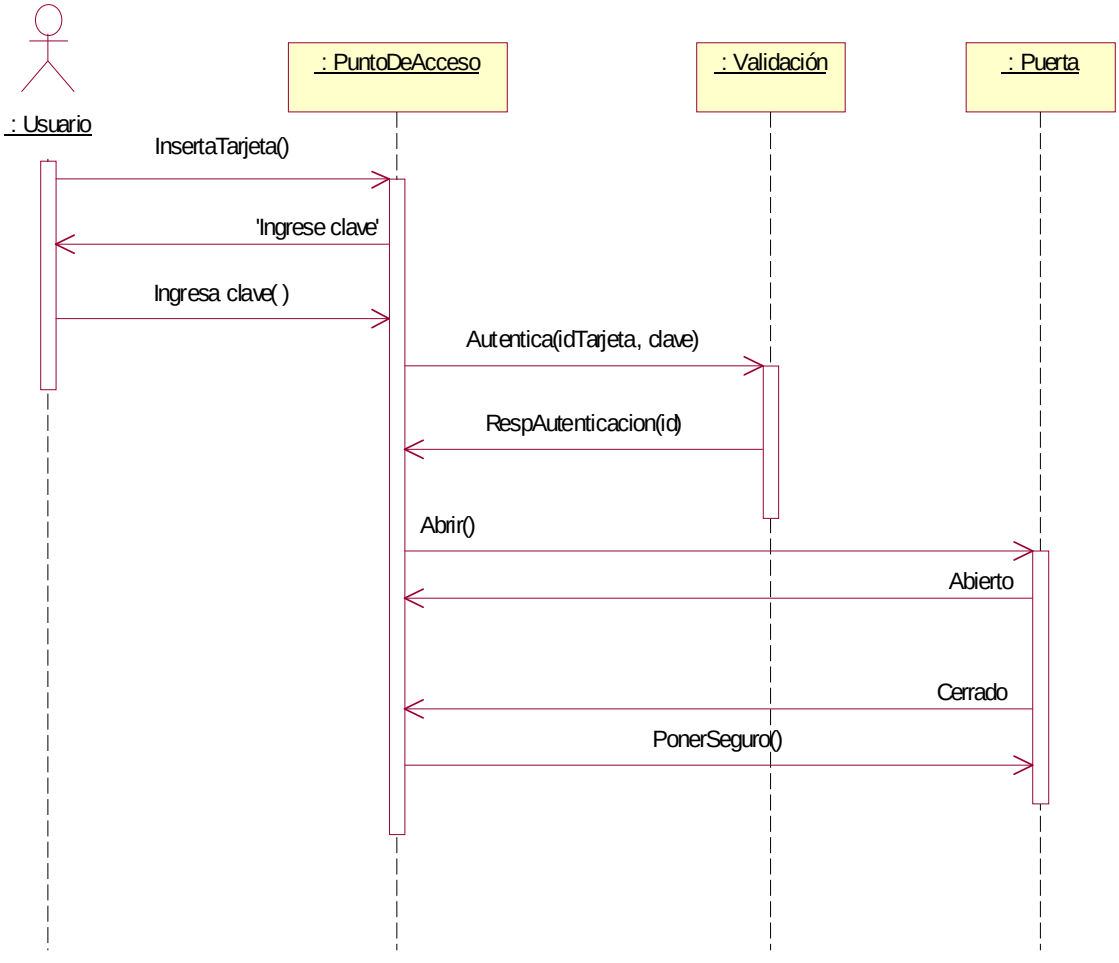


Figura 1.8 Ejemplo de diagrama de secuencias

- Diagrama de colaboración. Registran el intercambio de mensajes en el contexto de las relaciones estructurales generales de los objetos como se muestra en la figura 1.9.

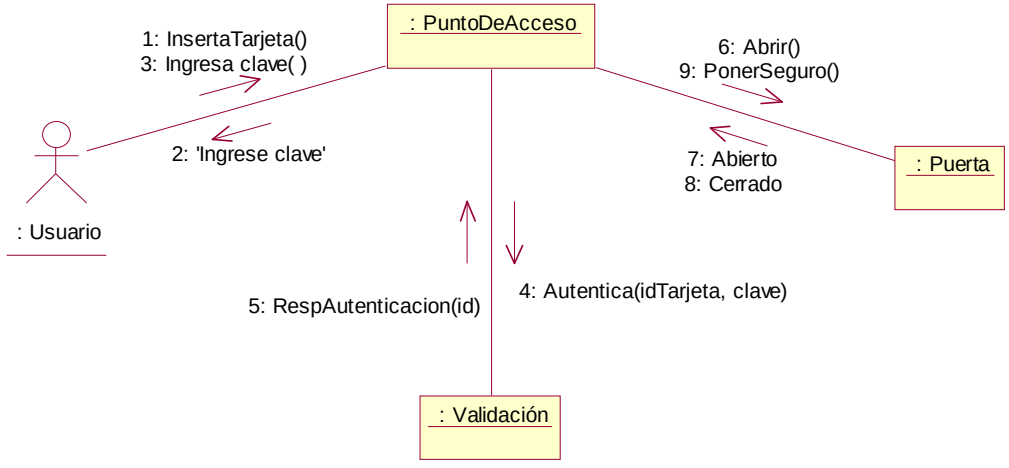


Figura 1.9 Ejemplo de diagrama de colaboración

- Diagrama de componentes. Representan la manifestación física de una parte del sistema, como algún archivo o ejecutable. Típicamente un componente se mapea a una o más clases o subsistemas como se ilustra en la figura 1.10.

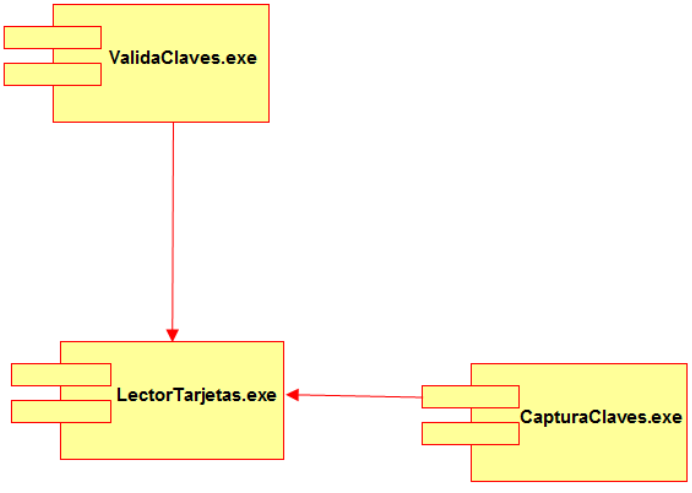


Figura 1.10 Ejemplo de diagrama de componentes

- Diagrama de despliegue. Muestra la arquitectura del sistema desde la perspectiva de los nodos y procesadores, así como también las relaciones entre ellos. Normalmente uno o más componentes se asocian a un nodo de despliegue. Son útiles para modelar y desarrollar arquitecturas de sistemas distribuidos. La figura 1.11 muestra un ejemplo de este tipo de diagramas.

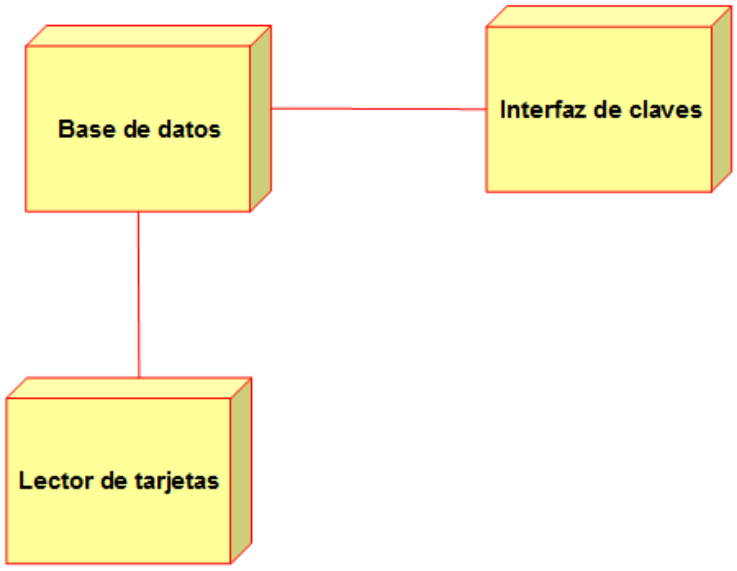


Figura 1.11 Ejemplo de diagrama de despliegue

1.5 ITIL

La Biblioteca de Infraestructura de Tecnologías de Información, abreviada como ITIL, por sus siglas en inglés, proporciona un marco de trabajo para la gestión de servicios de tecnologías de información, mediante una lista de las mejores prácticas adquiridas con el tiempo, relacionadas al control, operación y administración de los recursos disponibles y así, promover un acercamiento constante hacia la gestión de calidad y mejora continua de las tecnologías de información. De esta manera, se atienden necesidades tales como:

- Planeación estratégica para el negocio y las tecnologías de información
- Integración y alineación entre las metas del negocio y las tecnologías de información
- Implementación del concepto de mejora continua
- Medición de la efectividad y eficiencia de las tecnologías de información en la organización
- Optimización de costos
- Utilización de tecnologías de información para aumentar la ventaja competitiva
- Administración del cambio tanto en el negocio como en las tecnologías de información

El objetivo principal de la administración de servicios es asegurar que las tecnologías de información estén alineadas a las necesidades del negocio y soportarlo de manera activa. Es de gran importancia que las tecnologías de información actúen como un agente de cambio para facilitar la transformación del negocio. Para ello, se definen las etapas del marco de trabajo de ITIL, mostradas en la figura 1.12, que van desde la definición y análisis iniciales de los requerimientos del negocio en la estrategia y diseño del servicio, a través de la migración al ambiente de producción dentro de la etapa de transición del servicio, para llegar a la operación diaria y mejoramiento mediante las etapas de operación del servicio y mejora continua.

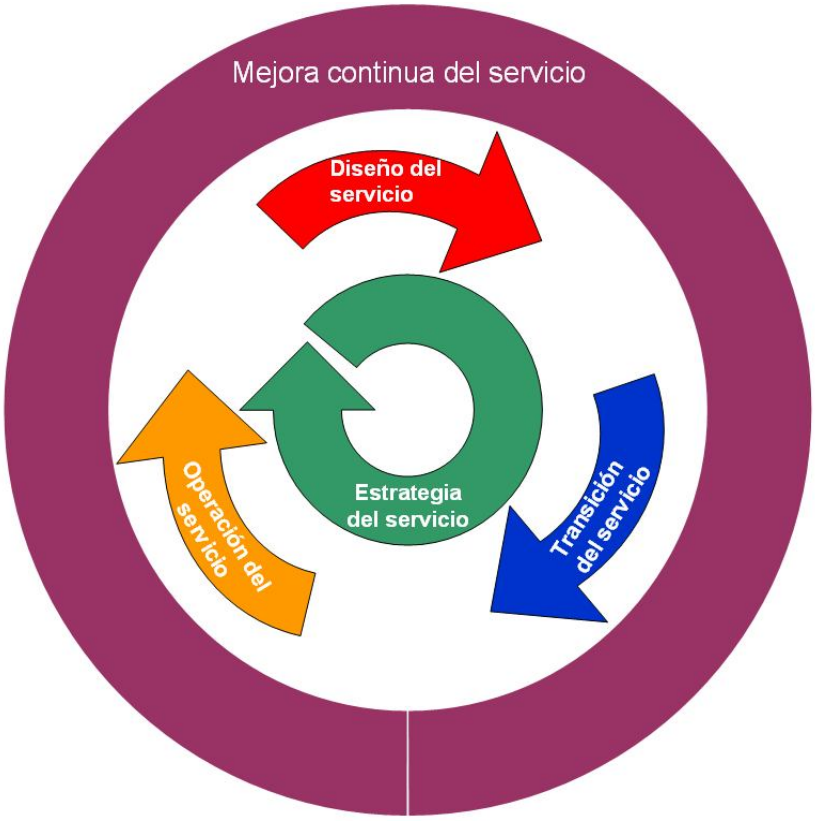


Figura 1.12 Marco de trabajo de ITIL

En cuanto a la etapa de estrategia del servicio de ITIL, se establece la forma de diseñar, desarrollar e implantar la gestión de la cartera de servicios como un activo estratégico. Posteriormente, en la etapa de diseño del servicio, se define un plan de administración de elementos, tales como capacidad, disponibilidad y continuidad del servicio, proveedores, niveles de respuesta, así como también la seguridad de la información.

Después, en la etapa de transición del servicio, se gestiona la configuración, la adaptación al cambio, pruebas, validación e implantación del servicio, sin descartar el apropiado manejo del conocimiento.

Referente a la etapa de operación del servicio, se proponen guías para obtener eficiencia y efectividad en la entrega y soporte del servicio mediante la gestión de accesos, eventos, incidentes y problemas para cumplir con las expectativas en cada solicitud de servicio.

Finalmente, la etapa de mejora continua del servicio se encarga de identificar posibles áreas de oportunidad para las etapas de diseño, transición y operación del servicio, con el objetivo de proporcionar más valor al cliente, mediante informes y mediciones de variables representativas del comportamiento del servicio.

1.6 Conclusiones del capítulo

Una vez terminada la revisión de diversas metodologías de ingeniería de software, es evidente el cambio que han tenido en el transcurso de los años para adaptarse a las necesidades del mundo actual. De igual manera, se aprecia que la incorporación de gestión de riesgos y enfoques de desarrollo incremental tienen cada vez mayor importancia al momento de generar un producto de software nuevo. A su vez, la participación del cliente en un proyecto de este tipo, es cada vez mayor, pues de esa manera se asegura el cumplimiento de sus expectativas respecto a los requerimientos que ha proporcionado. Asimismo, para facilitar el entendimiento de los requerimientos, herramientas como UML han sido utilizadas de manera exitosa, causando una mejor comunicación entre el cliente y los desarrolladores de software

En lo que respecta al mantenimiento de software, podemos observar que las mejores prácticas y lecciones aprendidas en el pasado han dado lugar a estándares destinados a la apropiada gestión del mantenimiento y soporte de sistemas. Por ello es que librerías como ITIL son comúnmente usadas en la actualidad.

Capítulo 2

Planteamiento del problema

2.1 Antecedentes y contexto del proyecto

En lo que respecta a los antecedentes del proyecto, se describe inicialmente el contexto donde se utiliza el producto de software. Dentro de la empresa del cliente para quien se desarrolla este proyecto, se manejan varios segmentos de negocio, entre los que se encuentran:

- Servicios financieros para consumidores y negocios pequeños
- Servicios de salud relacionados a investigación y desarrollo de equipos de diagnóstico
- Producción de aparatos electrónicos para el consumidor
- Servicios energéticos dedicados a la elaboración de tecnologías y productos comerciales de generación de energía eléctrica

En este último segmento existe un área dedicada a la optimización del desempeño de sus productos llamada Performance Services, misma que provee a sus clientes de paquetes de mejoras y valor agregado para acrecentar la eficiencia, resultados, confiabilidad y disponibilidad de turbinas de generación de energía. Estos paquetes, conocidos internamente como proyectos de venta, cubren las tecnologías de gas, vapor y generación, así como también una amplia gama de necesidades.

Por otro lado, la empresa cuenta con un gran número de administradores de proyectos, cuya labor es asegurar que toda la organización esté involucrada en la entrega de los paquetes de valor agregado y exista una adecuada coordinación para así entregar un producto de calidad y servicio dentro o antes de lo planeado en tiempo y costo. Existe una constante necesidad para mejorar y expandir las herramientas digitales usadas por el equipo de Performance Services en un esfuerzo continuo orientado a cumplir con las expectativas de sus clientes apoyándose fuertemente en las tecnologías de información. El *Sistema informático para la*

gestión de proyectos (SIGP) es la aplicación web en la que se enfoca este proyecto. Esta aplicación es utilizada para brindar visibilidad y administrar elementos como el alcance, monitorear el calendario de eventos, así como también interactuar con los aspectos financieros para el cumplimiento de los paquetes de valor agregado. Aunque Performance Services es el área primordial donde se localizan los usuarios de esta aplicación web, también interactúa con varios procesos de ventas, ingeniería y manufactura. La misión es mejorar la productividad de los usuarios mediante la integración de estos procesos a través del SIGP, mismo que está actualmente en ambiente de producción siendo utilizado por el área de Performance Services. La versión anterior de esta aplicación fue desarrollada bajo una arquitectura propia del cliente y una base de datos con un modelo de datos empresarial para una administración de datos más robusta y apegada a la organización.

2.2 Descripción del sistema

Específicamente para las aplicaciones web, se cuenta con una plataforma comprendida por el lenguaje de programación Java involucrando componentes XML, HTML, Java Script y hojas de estilo en cascada CSS bajo un Framework o marco de trabajo propio del cliente el cual está basado en Struts y el modelo MVC. De esta manera es como la implementación de servlets ha sido manejada en todas las versiones del SIGP, incluyendo a la que se refiere este trabajo. A su vez, las aplicaciones interactúan con bases de datos bajo la tecnología Oracle, en donde se realiza el manejo y la organización de la información para mantener la integridad y disponibilidad de los datos. Otros componentes relevantes son manipulados a través de integraciones con diferentes sistemas las cuales siguen el modelo ETL (por sus siglas en ingles Extract, Transform and Load).

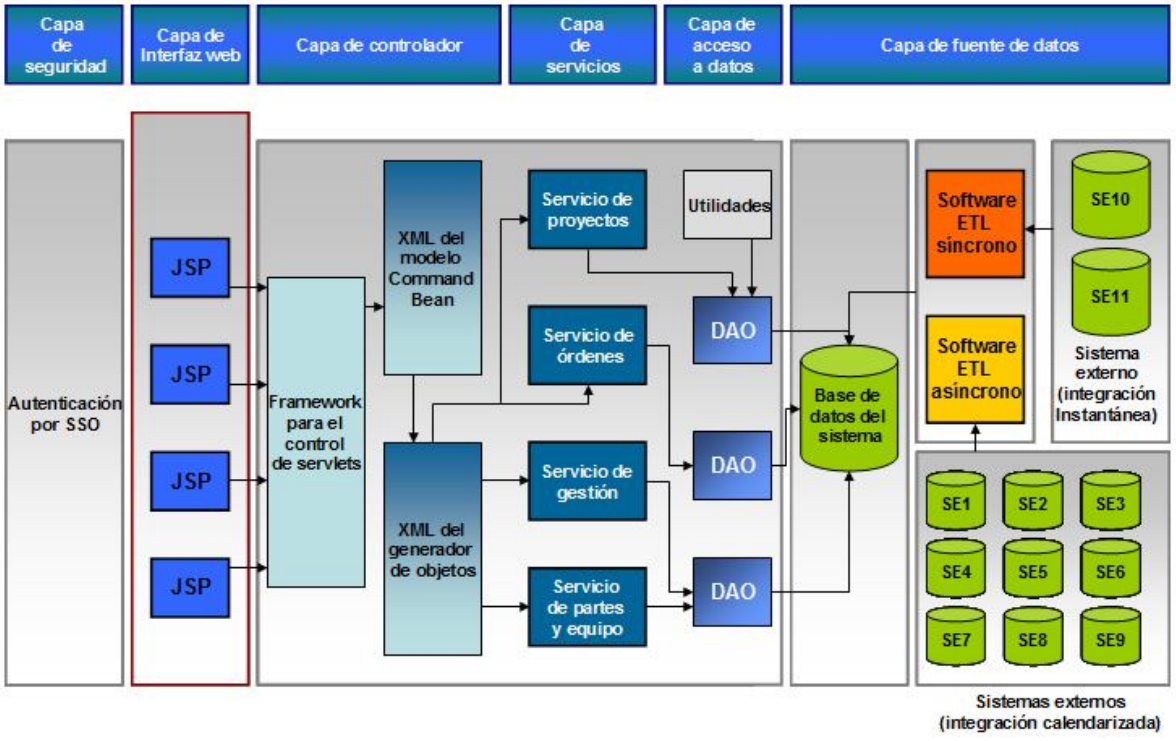


Figura 2.1 Plataforma del Sistema informático para la gestión de proyectos

Como se muestra en la figura 2.1, la plataforma del Sistema informático para la gestión de proyectos está compuesta por las siguientes capas:

- Capa de seguridad. El acceso a la aplicación está gestionado a través de single sign on (SSO) aplicado a las tecnologías web. Este procedimiento proporciona un método seguro y práctico para los usuarios, pues brinda la posibilidad de utilizar múltiples sistemas mediante una sola instancia de identificación.
- Capa de interfaz web. Compuesta generalmente por páginas JSP (Java Server Pages) que conforman la interfaz mediante la cual se mantiene la interacción entre los usuarios y el sistema. Después de que todos los Java beans han terminado de cargar los objetos de datos con la información necesaria para ser presentada al

usuario, la capa del controlador utiliza el framework de control de servlets para distribuir las peticiones a la capa de presentación.

- Capa de controlador. Está construida por el framework propio del cliente utilizado para el control de servlets. Asimismo, incluye el modelo de los comandos, también referidos como Java beans, con su correspondiente archivo XML descriptivo para definir las interacciones de la aplicación, pues representan la implementación de la lógica de negocio del sistema. Finalmente, cuenta con un componente generador de objetos para crear instancias conforme sean necesarias.
- Capa de servicios. Es el conjunto de clases Java que agrupan los métodos y dividen la lógica de negocio de acuerdo categorías conceptuales relacionadas a las funcionalidades del sistema. Por ejemplo, se puede definir un servicio de partes y equipos como el conjunto de métodos Java dedicados al procesamiento de información relacionada a los conceptos del mencionado servicio.
- Capa de acceso a datos. Proporciona al sistema una forma sencilla de acceso a bases de datos y archivos XML en formato renglón/columna. Define el enlace entre los elementos de código Java y las operaciones de inserción, recuperación y borrado de datos.
- Capa de fuente de datos. Es el conjunto de fuentes externas con los que el SIGP interactúa y de las que necesita alimentarse para actuar como un elemento integrador de la información necesaria para ejecutar un proyecto de venta. Se compone principalmente de las bases de datos externas y los componentes de software ETL (Extract, Transform and Load) que, como su nombre lo indica,

extraen los datos de otros sistemas y los transforman de manera que puedan ser cargados al SIGP. Dicho proceso puede ejecutarse de las dos maneras siguientes:

- Integración instantánea o síncrona. Ocurre cuando se necesita que en el momento en que la fuente de datos es modificada, inmediatamente el sistema destino sea actualizado.
- Integración calendarizada o asíncrona. Sucede cuando se define un momento específico en el tiempo, en el que se actualizarán los datos en el sistema destino con la información del sistema fuente, utilizando los datos nuevos que hayan sido actualizados en un periodo de tiempo anterior al de la carga de información en el sistema destino. Por lo regular, se calendariza en horas no laborables para que la carga de datos no tenga impactos de integridad de datos.

Precisamente el SIGP está construido bajo esta arquitectura, en la que a grandes rasgos se hace uso del lenguaje de programación Java orientado a aplicaciones web, mediante servlets, compuestos por vistas del tipo JSP (Java Server Pages) y comandos implementados en Java. A su vez, la información está almacenada en una base de datos con tecnología Oracle en un modelo de datos empresarial.

2.3 Análisis de la información previa al desarrollo

Debido a fallas generales recurrentes de la aplicación web, surgió la necesidad de analizar el comportamiento de las mismas para obtener la situación del sistema previa al desarrollo. Se entiende como falla general, al evento en el cual la aplicación no es accesible en su

totalidad por los usuarios por cierto periodo de tiempo. En la figura 2.2 se muestran las fallas por mes que se tuvieron desde enero de 2007 hasta noviembre del 2008.

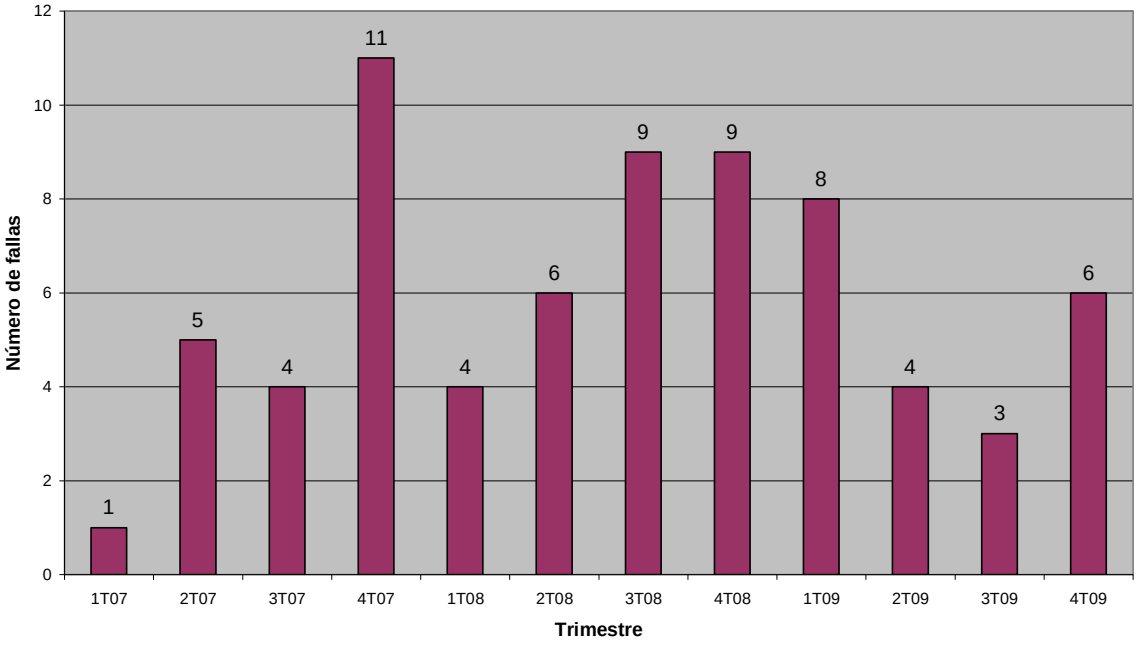


Figura 2.2 Fallas generales por trimestre (2007 a 2009)

Por otro lado, después de juntas internas con los usuarios, asi como retroalimentaciones con el equipo de soporte de aplicaciones, se concluyó que el SIGP requería de puntos de control críticos para mantener la consistencia de datos y lograr que los procesos internos del cliente pudieran hacer uso de la aplicación web confiando en la informacion disponible. A su vez, nuevos requerimientos derivados de la reingenieria de procesos de negocio fueron incluidos en el proyecto. De esta manera, se pretende asegurar que la aplicación evolucione junto con los procesos de negocio.

Capítulo 3
Desarrollo del SIGP

3.1 Objetivo del sistema informático para la gestión de proyectos

Como ya se ha mencionado, el SIGP es una aplicación web usada para administrar todos los proyectos del área de Performance Services. Dentro de la empresa este elemento de software es considerado de carácter altamente crítico para sus procesos de mejora continua y ventas, pues en ella los clientes visualizan y manejan información relacionada a las finanzas, seguimiento de partes, gestión de entregables, entre otros. A continuación, se enlista de manera general las áreas que la aplicación comprende:

- Personalización de proyectos
- Almacenamiento de direcciones físicas relacionadas a la entrega del producto
- Calendarización y seguimiento de la facturación de partes
- Gestión de la documentación relacionada a los proyectos
- Registro de los requerimientos comerciales de proyectos de turbinas
- Administración de riesgos
- Manejo de inventarios de componentes físicos
- Registro de base de conocimiento para las mejores prácticas de los proyectos
- Módulo de reportes específicos de cada proyecto
- Sección de reportes del negocio
- Conjunto de reportes generales
- Seguimiento del estado de pérdidas y ganancias de cada proyecto
- Registro de documentos de créditos
- Seguimiento del flujo de aprobaciones para liberación de contrato final del proyecto
- Integración con diversos almacenes de datos y ERPs

Debido a lo anterior, se aprecia que el alcance del SIGP dentro de la empresa es muy amplio. Por otro lado, es de los sistemas más utilizados por los clientes y que agregan de manera importante valor a la compañía.

Con el objetivo de incrementar las ventas y reducir gastos y tiempos de entrega, surge la necesidad de efectuar cambios de relevancia en los procesos internos y externos, así como reestructuraciones organizacionales. Por estas razones, la aplicación necesita adaptarse a los procesos y objetivos de productividad de manera completa.

Otro de los factores claves que dieron pie a la generación de un nuevo producto de software, fue el relacionado a las fallas del sistema. La primera versión del SIGP fue desarrollada en el año 2003. A partir de entonces se actualizó el sistema de manera recurrente mediante mantenimiento adaptativo cada año, hasta el 2006, cuando la versión 4 se hizo disponible. Estas actualizaciones anuales atendían constantemente los requerimientos emergentes del negocio y corregían fallas que el sistema pudiera tener. No así, en el periodo comprendido entre 2006 y 2009, ninguna ventana importante de mantenimiento o mejora fue implementada. El efecto de 3 años sin actualizaciones o correcciones impactó de manera alarmante las operaciones directamente dependientes del funcionamiento apropiado de la aplicación, provocando inconsistencias entre los procesos de negocios en curso y la lógica aplicativa del sistema. Asimismo, la infraestructura mostró evidencia de fallas después del largo tiempo sin el adecuado mantenimiento. En la figura 3.1 se muestra la representación gráfica de los datos históricos relacionados a las fallas generales del sistema, por trimestre en el periodo comprendido entre 2007 y 2009, previas a la instalación de la versión desarrollada en este trabajo de tesis. Posteriormente, se

mostrarán las estadísticas después de la implementación del nuevo sistema para analizar los efectos y resultados del proyecto.

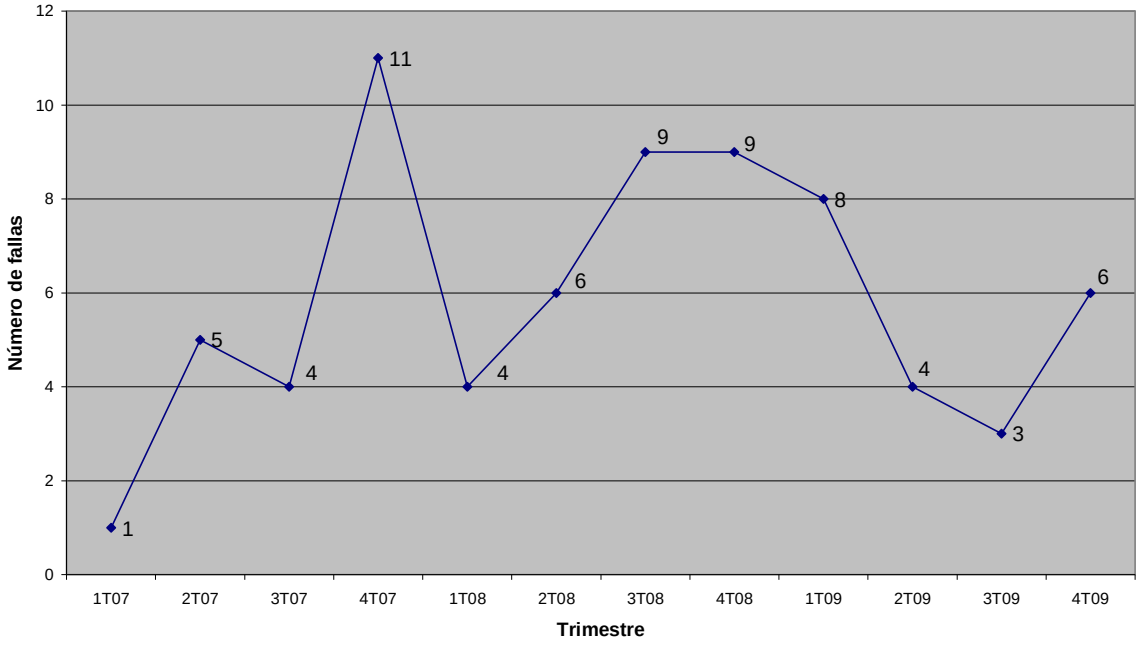


Figura 3.1 Fallas generales trimestrales antes del desarrollo

Se puede apreciar en la figura 3.1 que la disponibilidad del sistema sufrió numerosas interrupciones durante el periodo en el que hubo ausencia de mantenimiento y mejoras. La figura 3.2 representa las fallas por año, antes de que el nuevo sistema fuera liberado. Los niveles que se presentan son muy altos considerando la relevancia de esta aplicación para la compañía. En consecuencia, los usuarios del sistema externaron su alta inconformidad, pues el sistema perdía rápidamente confiabilidad y las ventas de proyectos tendían a verse afectados negativamente.

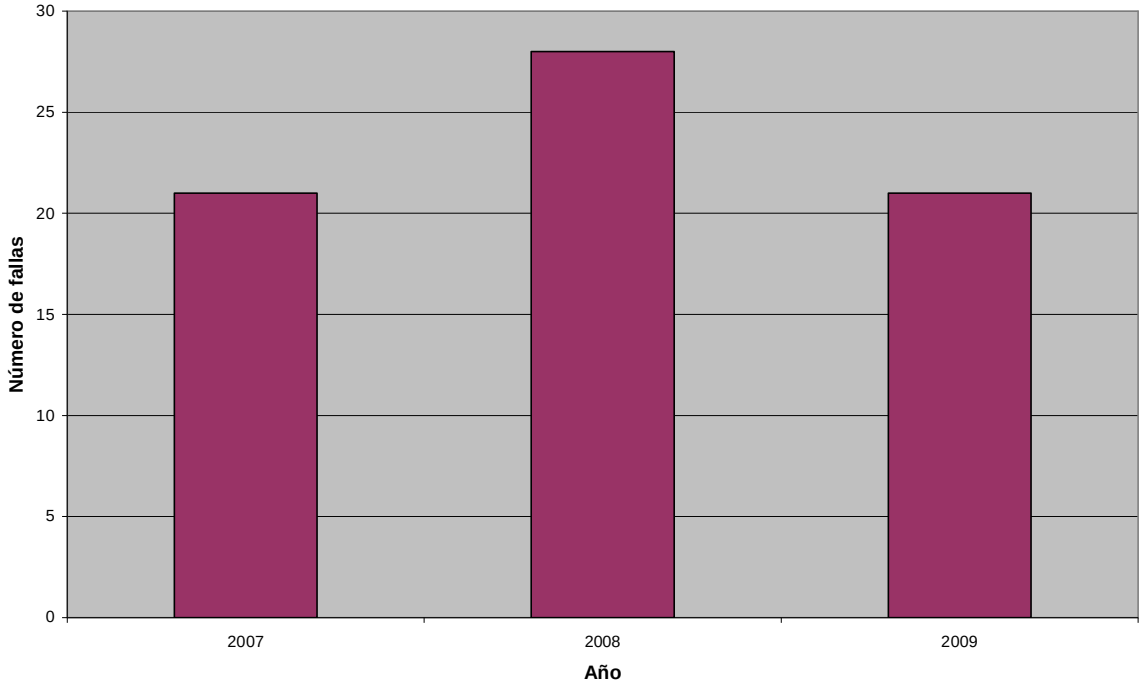


Figura 3.2 Fallas generales por año

Por los factores anteriormente mencionados, el sistema necesita ser actualizado para evitar fallas generales en la aplicación que interrumpan el servicio de la misma. De igual manera el SIGP debe apegarse a los procesos actuales y volver a ser una herramienta de alto nivel de productividad que responda a los requerimientos del cliente. Por lo tanto, para cumplir con estos objetivos, un nuevo grupo de requerimientos, mejoras y correcciones fueron identificados e implementados como parte del proyecto del SIGP versión 5. En resumen, los requerimientos solicitados deben cubrir los siguientes objetivos:

- Mejorar la seguridad de la aplicación
- Asegurar la precisión de los datos en el sistema
- Tener interfaces sencillas para usuarios finales
- Agregar controles apegados a la lógica de negocio para tener un flujo adecuado de los proyectos manipulados en el SIGP

- Sincronizar los datos entre cada fase de los proyectos registrados en el SGIP
- Corregir y mejorar las integraciones que el SIGP tiene con los sistemas externos
- Aumentar la disponibilidad del SIGP

El propósito del proyecto fue desarrollar una nueva versión de la aplicación que atendiera estas necesidades de adaptación a los nuevos procesos de la organización, así como también mejorar el funcionamiento, disponibilidad y desempeño de la misma.

3.2 Expectativas y entregables del proyecto

Como parte de un proyecto de desarrollo como éste, se consideran como entregables los siguientes elementos:

- Diseño de la nueva versión. Una vez recibidos los requerimientos, el equipo de desarrollo discute y propone el diseño para la solución y futura implementación de los requerimientos, generando así la documentación asociada.

- Desarrollo de la nueva versión. Después de contar con un diseño de solución de los requerimientos se codifica su implementación. Los archivos de código generados forman parte de los entregables que serán otorgados al cliente.

- Pruebas unitarias y de aceptación de usuario. Al terminar la codificación se hacen pruebas a cada elemento modificado o creado para hacer las correcciones pertinentes y reducir los defectos. Los resultados de las pruebas deben consolidarse y validarse con el cliente para resguardarlos como evidencia de su ejecución.

- Cambios en configuración. Si así lo amerita, también se considera cualquier cambio de configuración de la aplicación, tales como actualizaciones del servidor de aplicaciones, cambios en el framework de programación, movimientos de la aplicación a otro servidor,

etc. De igual manera debe generarse documentación al respecto para que, de ser necesario, sirva de referencia futura.

- Soporte en la entrega al ambiente productivo. Una vez que la fecha de entrega sea definida y la aplicación esté lista y validada, se procede a reflejar los cambios en contenedor de aplicaciones. Para ello se interactúa con equipos de servidores Unix, Servicios Web, Base de Datos y administración de accesos. El equipo ayuda a que se instale correctamente la aplicación, así como también se involucra en la ejecución de scripts para adaptar la base de datos a los requerimientos nuevos.

- Soporte de post-producción. Para mantener los niveles de calidad, después de la entrega de la aplicación se tiene un periodo de soporte a post-producción o garantía en el que el equipo debe estar al pendiente, por cualquier comportamiento inusual en la aplicación, para poder analizarlo al momento y revisar si es consecuencia del desarrollo de la nueva versión o si es un factor ajeno.

- Transferencia de conocimiento al equipo de soporte y mantenimiento del sistema. Parte de los acuerdos generados en el proyecto consiste en transferir el conocimiento técnico y funcional de los cambios implementados para que el equipo encargado de mantener el sistema pueda realizar sus actividades, de acuerdo a la nueva versión y evitar perder tiempo en investigaciones innecesarias.

- Asimismo, se modifica o genera la documentación de la aplicación para que al momento de necesitar alguna referencia, se estén revisando los documentos actualizados acordes con la nueva versión. Se incluyen documentos tales como:

  - Documento de especificaciones funcionales
  - Documento de interfaz de usuario

- Documento de arquitectura de la aplicación
- Documento de especificaciones técnicas del diseño
- Plan de pruebas
- Documento de los casos de prueba
- Guía de soporte a producción
- Guía de configuración e instalación

3.3 Aplicación del modelo en espiral

A continuación se presenta la aplicación del modelo en espiral al desarrollo del nuevo sistema, organizado por los ciclos o vueltas por los que pasó desde sus etapas iniciales, donde se analizan elementos como la factibilidad del proyecto (considerada como vuelta 0, pues es preliminar al desarrollo del sistema), la generación del plan de trabajo, hasta llegar a la entrega final del producto listo para continuar con su etapa de mantenimiento.

3.3.1 Vuelta 0: Estudio de factibilidad

3.3.1.1 Fase de planificación

Para la primera iteración del modelo se definió como objetivo lo siguiente:

Incrementar la productividad y reducir los problemas recurrentes considerando que el número de usuarios de la aplicación se incrementaría considerablemente.

Las principales restricciones identificadas durante esta fase fueron el elevado costo que involucraba cumplir con dicho objetivo mediante la generación de un nuevo sistema capaz de cubrir las necesidades actuales y nuevas, así como también modificar la infraestructura actual para alojar al nuevo sistema. Por otro lado, de acuerdo a la experiencia del negocio y a la antigüedad tanto del sistema actual como de los usuarios, se recibieron muestras

importantes de resistencia al cambio por medio de comunicados para conservar el sistema actual y no modificar ese esquema.

Considerando estas restricciones se formularon las siguientes alternativas:

- Realizar mejoras menores para las principales necesidades y así evitar un cambio tan grande
- Generar una nueva versión del sistema para lograr un cambio relevante e incrementar la productividad del área de Performance Services de manera substancial.
- No realizar ningún cambio y permanecer con el sistema actual, evitando costos y problemas de resistencia al cambio.

3.3.1.2 Fase de análisis de riesgo

Poniendo en perspectiva el panorama de la situación, se identificaron los siguientes riesgos que podían comprometer el cumplimiento del objetivo:

- Existía una falta de profundidad en los detalles respecto a las necesidades del negocio
- No había un consenso respecto a la definición adecuada de los requerimientos, pues la mayoría de los usuarios principales estipulaban que sus requerimientos eran los más críticos y de alta prioridad
- Permanecía la incertidumbre respecto al tiempo de retorno de la inversión, pues el cliente no estaba seguro si valía la pena tanto esfuerzo, contra el tiempo en el que los beneficios serían visibles y medibles.

3.3.1.3 Fase de ingeniería

Para atender las fuentes de riesgo y la problemática involucrada en el cumplimiento de los objetivos se realizaron las siguientes actividades:

- Encuestas, análisis y reuniones con el equipo funcional (cliente) para determinar de manera concreta las necesidades del negocio, así como también sus quejas y peticiones.
- Se tuvieron reuniones con los equipos de soporte e infraestructura para generar el perfil técnico de la aplicación actual en términos de versión de componentes, tamaño de la base de datos, llamadas de servicio frecuentes, tipos de fallas frecuentes con sus respectivos RCAs ("Root-Cause Analysis", es decir análisis de la causa raíz). Parte del perfil técnico resultante fue el siguiente:
    - Servidor de base de datos: Oracle 9.2.0.4
    - Servidor de aplicaciones: JBoss 3.2.1
    - Versión de Framework: Casper 3.9
    - Librerías de terceros: fop205.jar, jdom.jar y jxl.jar
    - Versión de Java: J2EE 1.4
    - Desarrollado y contenido en ambiente compartido ubicado en la ciudad de Alpharetta, Georgia.

- Se analizó la posibilidad de que el equipo de soporte pudiera manejar las nuevas modificaciones sin tener que crear un equipo de desarrollo nuevo.

3.3.1.4 Fase de evaluación

Una vez finalizadas las actividades de ingeniería, se obtuvieron los siguientes resultados:

- Después de revisiones con el equipo funcional y usuarios clave se generó el documento llamado Business Requirements Document (BRD), que contenía la lista de requerimientos generados por el cliente y aprobados por los dueños del sistema. Asimismo, estipulaba la visión y justificación de los requerimientos después de un consenso entre el equipo funcional y los usuarios.
- Se llegó a la conclusión de que la versión actual del sistema ya no cumplía con las necesidades del negocio. A su vez, un grupo reducido de requerimientos tampoco podían cubrir la totalidad de las nuevas necesidades y, por tanto, el equipo de soporte no era capaz de ejecutar un proyecto de esta magnitud por su propia cuenta. Como se estipula en los acuerdos de nivel de servicio, el equipo de soporte sólo dispone de 280 horas efectivas para desarrollo de mejoras, mismas que son por mucho insuficientes para el desarrollo de la nueva versión.
- Finalmente, se toma la decisión de crear un nuevo proyecto de desarrollo para producir una nueva versión del sistema actual.

En esta primera vuelta el principal entregable resultó ser el documento BRD y la aprobación del proyecto de desarrollo. Para el cliente el siguiente paso consistió en desempeñar las tareas necesarias para conseguir un proveedor que pudiera desarrollar el proyecto.

3.3.2 Vuelta 1: Propuesta general al cliente

3.3.2.1 Fase de planificación

Tomando como base el documento BRD que el cliente proporcionó, el siguiente objetivo fue proporcionar una respuesta oportuna y apropiada y hacer una propuesta basada en estimaciones de tiempo y costo de los requerimientos a desarrollar. El proceso de selección de proveedor fue definido por el cliente y la idea era ser seleccionados para trabajar en el nuevo sistema.

3.3.2.2 Fase de análisis de riesgo

El mayor riesgo identificado en esta etapa fue la posibilidad de generar una estimación inadecuada de los requerimientos. La causa probable de una mala estimación podría ser debido a preguntas técnicas y funcionales no aclaradas.

Otro riesgo a atender fue el considerar la posibilidad de no contar con los recursos y capacidades necesarias para desarrollar el proyecto, en caso de ser elegidos como el proveedor.

3.3.2.3 Fase de ingeniería

Mediante un análisis detallado de cada requerimiento se generaron los documentos respectivos de estimación en tiempo y costo. Esta tarea la llevó a cabo un equipo conformado por un analista, el equipo de soporte de la aplicación actual, un BRM (Business Relationship Manager), aprovechando un par de sesiones de preguntas y respuestas proporcionadas por el cliente en las que cada posible proveedor tenía oportunidad de aclarar dudas. Los roles involucrados en las sesiones fueron los siguientes:

- BRM
- Advance Solutions Engineer
- Experto de aplicación
- Equipo de soporte
- Dueño de aplicación (conocido como IM Owner)
- Líderes funcionales
- Analista de negocios

Por otro lado, se trabajó con el área de recursos humanos de la compañía para conocer la situación de la misma en cuanto a recursos humanos disponibles.

3.3.2.4 Fase de evaluación

Una vez finalizadas las estimaciones, se generó un documento con la propuesta para el cliente, la cual incluía una versión preliminar del plan del proyecto. Dicho documento fue enviado al área de Performance Services para su evaluación y comparación con las demás propuestas.

En la estimación, se propuso la siguiente estructura en el equipo de desarrollo:

- Un analista certificado Black belt en Six Sigma
- Un BRM
- Un Operational Leader
- Un Project Leader
- Tres desarrolladores
- Tres testers

Respecto a los recursos humanos disponibles, se concluyó que se contaba con la suficiente capacidad para abordar el proyecto. La decisión del cliente fue que la compañía debía hacerse cargo del proyecto.

Los principales entregables de esta etapa fueron la estimación y propuesta al cliente, así como también el plan de trabajo de alto nivel. Los siguientes pasos a trabajar fueron obtener los detalles actuales de la aplicación, tales como documentación, localización de servidores, detalles de acceso, entre otros, así como también programar reuniones para obtener el máximo detalle de la aplicación y sus requerimientos que resultaran en el diseño de la misma.

3.3.3 Vuelta 2: Definición detallada de requerimientos y documentación

3.3.3.1 Fase de planificación

Para esta iteración del modelo en espiral, el objetivo se centra en obtener de cierta manera el primer prototipo del sistema en papel, esto es, mediante la creación de la documentación relacionada con el desarrollo. Las restricciones encontradas radicaron en la falta del nivel de detalle adecuado para generar los documentos de manera precisa. Dado que ya existía la documentación del sistema en uso, se contaba con la alternativa de apoyarse en la misma para generar la nueva documentación o crearla desde cero.

3.3.3.2 Fase de análisis de riesgo

Dependiendo de la alternativa elegida se definían diferentes riesgos. Por un lado, la documentación existente podía servir de apoyo para agilizar el proceso de análisis y escritura de los nuevos documentos, pero se sabía que la misma estaba desactualizada e incompleta. Por el otro lado, generar la documentación desde cero involucraba mayor

esfuerzo y tiempo que podía retrasar los entregables y generar insatisfacción por incumplimiento de las fechas prometidas.

Finalmente, un riesgo que frecuentemente está latente es que el cliente puede cambiar de parecer en cualquier momento, ya sea modificando definiciones de requerimientos ya establecidas o agregando mayor alcance al proyecto de desarrollo con nuevos requerimientos.

3.3.3.3 Fase de ingeniería

Con el apoyo y experiencia del equipo de soporte se pudieron basar los nuevos documentos en los ya existentes, tomando reservas de la existente desactualización de estos últimos. Aún así, la retroalimentación con el equipo de soporte sirvió para identificar los errores y no pasarlos a los nuevos documentos

3.3.3.4 Fase de evaluación

La evaluación de los documentos fue inicialmente manejada internamente en el equipo de desarrollo antes de enviarlos a los dueños de la aplicación. Contamos con el apoyo del Business analyst para hacer una evaluación preliminar y proporcionar retroalimentación respecto a la calidad de los documentos en cuanto a contenido y forma. Algunos errores fueron encontrados y corregidos, antes de enviar finalmente los documentos resultantes listados a continuación:

- Documento de especificación funcional
- Documentos de la interfaz de usuario
- Documentos de requerimientos

3.3.4 Vuelta 3: Diseño del sistema y primera liberación de capa de presentación

3.3.4.1 Fase de planificación

Una vez que los requerimientos han sido aclarados, documentados y validados tanto por el cliente como por el proveedor, la siguiente iteración del modelo se encargó del diseño de solución de los requerimientos. De igual manera, esta vuelta de la espiral fue utilizada para generar una versión de la aplicación, donde se desarrollará la capa de presentación del sistema sin contar con las funciones requeridas pretendiendo mostrar al cliente un demo del sistema.

Las principales restricciones para lograrlo fueron las siguientes:

- Los desarrolladores necesarios para el proyecto aún no estaban listos, de acuerdo al informe del área de recursos humanos.
- El ambiente de pruebas para el nuevo sistema aún no estaba definido, por lo que no había lugar en la infraestructura del cliente para alojar el prototipo de la aplicación.
- Existía una base de datos de desarrollo, sin embargo los datos no eran recientes y no reflejaban la realidad del sistema actual.
- Se carecía de capital de trabajo para comenzar con el desarrollo, pues el cliente no tenía planeado financiar el proyecto hasta después de un mes aproximadamente.

Las alternativas de solución a las anteriores restricciones se muestran a continuación:

- No crear la versión preliminar y esperar a que una versión completamente funcional estuviera lista.
- Crear la versión preliminar enfocando esfuerzo adicional para conseguir un ambiente de pruebas adecuado y conseguir los desarrolladores necesarios a la brevedad.

- Coordinarse con el cliente para obtener el capital de trabajo mínimo para iniciar las operaciones del proyecto.

3.3.4.2 Fase de análisis de riesgo

Por lo anteriormente mencionado, los riesgos radicaban en la posibilidad de retraso en la entrega del proyecto final, que la inversión no proporcionara la relación costo, beneficio y tiempo esperado, o que el cliente confundiera la versión prototipo con una versión funcional sino se aclaraban los detalles del mismo.

3.3.4.3 Fase de ingeniería

En lo que respecta a la ingeniería de este ciclo se realizó una revisión detallada de los documentos de análisis existentes en la versión anterior del sistema. El primer objetivo fue identificar las áreas y funcionalidades existentes relacionadas con los nuevos requerimientos para tener un mejor entendimiento y complementar el análisis del código fuente. De esta mantera, se buscó apegarse al diseño de la arquitectura general de la aplicación en sus anteriores versiones, dado que el código existente estaba bien organizado y estructurado.

Para el proyecto en cuestión, hubo una reingeniería de procesos del lado del negocio, lo que dio como resultado la generación de nuevas funcionalidades en la aplicación web y la restructuración y plan de mejora de otras ya existentes. Asimismo, se ejecutó un plan de análisis detallado, tanto en el código fuente como en la base de datos, para entender cada funcionalidad que se requería modificar y obtener una mejor visión del diseño de los requerimientos relacionados a los módulos nuevos.

Un ejemplo del análisis desempeñado se muestra en la figura 3.3, donde se parte de la pantalla mostrada en el browser para conocer los componentes involucrados mediante la dirección URL asociada. Siguiendo el modelo MVC, se puede saber que la interacción puede conformarse tanto por archivos de comandos tipo Java como los relacionados a la presentación del sistema al usuario (archivos JSP, Javascript o HTML). Al conocer el detalle de los comandos asociados se puede obtener el manejador de contexto, cuya función es la de servir como el contenedor de objetos y datos significativos para la interacción. También son identificados los servicios involucrados en la funcionalidad, utilizados para mantener una apropiada organización de las operaciones que definen la lógica del negocio en la aplicación y que son previas a la extracción de datos.

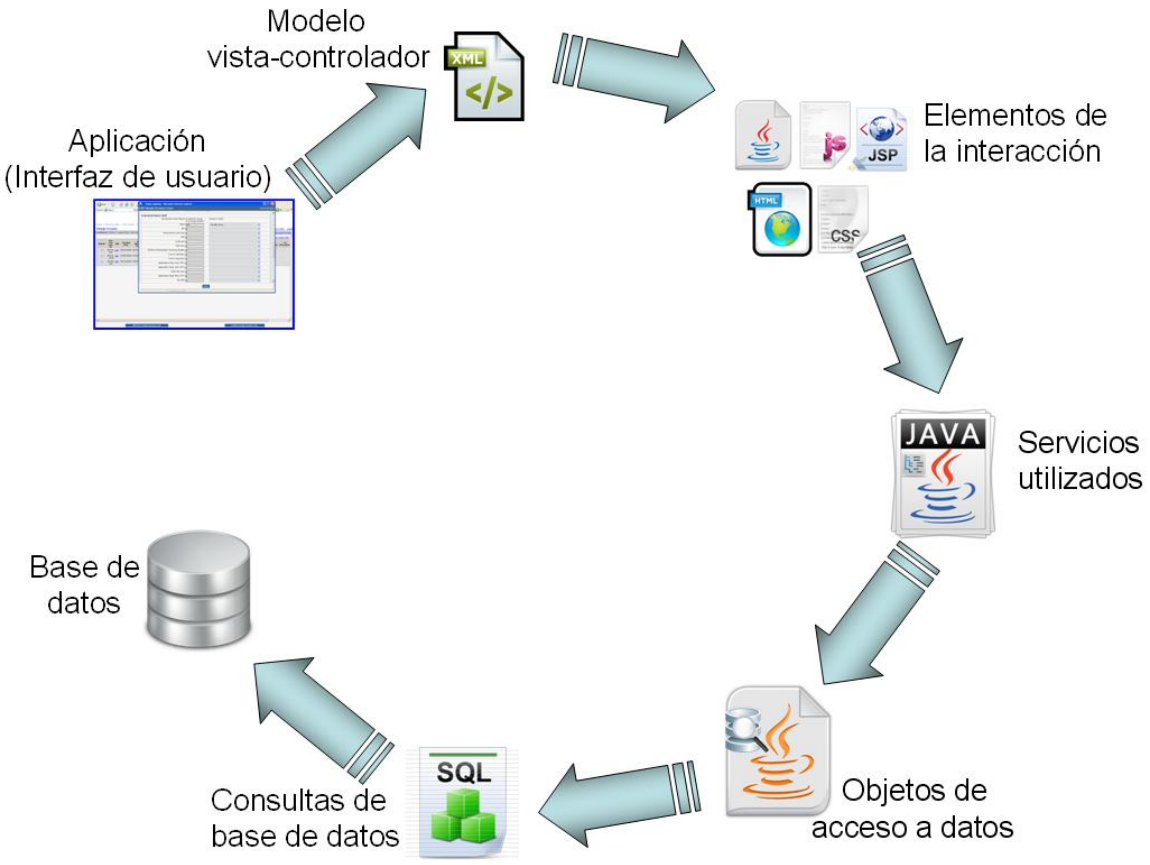


Figura 3.3 Seguimiento de análisis de interacciones del sistema

Posteriormente se ubican objetos de acceso a datos, los cuales se encargan de generar una interfaz apropiada entre la aplicación web y la base de datos mediante la configuración de la conexión a base de datos, la especificación de búsquedas apropiadas y la definición de los parámetros asociados a las consultas. En lo que respecta a la base de datos, se desempeña otro análisis similar para comprender los datos, tablas, triggers, vistas, procedimientos y paquetes que la aplicación necesita para brindar el servicio deseado. Como se mencionó previamente, esta aplicación tiene varias integraciones con otros sistemas, por lo que también necesitamos saber el mapeo entre los objetos fuente y destino la lógica de extracción. Dentro de las tecnologías y lenguajes utilizados en este proyecto se encuentran Java, JSP, XML, HTML, SQL, PL/SQL, entre otros.

Siguiendo la lógica de la interacción se conocen las operaciones, datos y procesos involucrados para reunir la información simplificada y relacionarla con el nuevo requerimiento.

Finalmente, el ejercicio de análisis ayudó a definir el diseño de cada uno de los requerimientos para alinearlos lo mejor posible a la estructura existente. El propósito fue mantener en todo momento la estructura para que el código no se desgastara ni se desorganizara.

3.3.4.4 Fase de evaluación

Al final de la fase de ingeniería se generó el prototipo junto con la siguiente documentación:

- Documento de arquitectura del sistema
- Especificación técnica del diseño del sistema

Cuando los usuarios evaluaron el prototipo generaron una lista de preguntas, errores encontrados, comentarios y sugerencias que fueron atendidos en la siguiente iteración del modelo.

3.3.4 Vuelta 4: Construcción de la parte funcional y pruebas

3.3.4.1 Fase de planificación

Con la retroalimentación del cliente los objetivos de esta vuelta se enfocaron a dar respuesta a las inquietudes y sugerencias del cliente, así como también en corregir los errores encontrados. En términos de reglas de negocio, el esfuerzo también se enfocó a su implementación en el sistema, para así generar una versión completamente funcional y que pudiera ser sometida a un proceso de pruebas.

Asimismo, las restricciones encontradas fueron relacionadas a los errores, dado que podían ser lo suficientemente relevantes como para consumir demasiado tiempo en su corrección. Respecto a los recursos humanos disponibles existía una limitante importante, pues en este punto la rotación del personal dentro del equipo fue grande y un nuevo proceso de adquisición de recursos se inició.

Finalmente, para la aplicación activa, previa a nuestro sistema en desarrollo, se generaron varios requerimientos de mantenimiento correctivo, pues los usuarios reportaron defectos de suma repercusión que no podían esperar a la liberación del nuevo sistema.

Las alternativas de solución a la problemática de la etapa actual fueron las siguientes:

- Debido a la magnitud del proyecto y a la cantidad de requerimientos se contempló la posibilidad de dividir el proyecto en dos liberaciones para atender primero los requerimientos de mayor prioridad, incluyendo aquellos relacionados con los

requerimientos de mantenimiento correctivo solicitados por los usuarios como urgentes.

- Continuar con el proyecto bajo el esquema de una sola liberación del sistema y que el equipo de soporte trabajara a la par para atender los problemas actuales urgentes.
- Atender primero los problemas urgentes para después continuar con el desarrollo de los requerimientos del proyecto.

3.3.4.2 Fase de análisis de riesgo

Durante esta vuelta, a nuestro modelo en espiral se generó una lista de riesgos relacionada a las restricciones y alternativas mencionadas en la fase anterior más las que los que tuvieron que ver con la administración de la infraestructura de base de datos, como se explica a continuación:

- El riesgo principal fue el no contar con los recursos humanos necesarios para continuar con el proyecto, pues podía retrasar gravemente el calendario de entregables.
- Los requerimientos de mantenimiento correctivo eran de tamaño considerable, ya que podían interferir con las actividades de desarrollo, considerando que los ambientes de desarrollo y calidad eran únicos y tanto las pruebas de desarrollo, como las de correcciones se ejecutaban en el mismo ambiente.
- El equipo de infraestructura por parte del cliente notificó que en las siguientes semanas las versiones de base de datos debían ser actualizadas de la versión 9i Oracle a la 10g del mismo proveedor.

3.3.4.3 Fase de ingeniería

Atendiendo a los requerimientos de documentación del sistema se generaron los siguientes entregables:

- Guía de configuración y despliegue del sistema
- Guía de soporte a producción

En cuanto a las actividades necesarias para llevar a cabo las pruebas pertinentes, se inició con la generación del documento de plan de pruebas en el que se definen las actividades requeridas para tener un ciclo de pruebas apropiado. Los documentos generados se listan a continuación:

- Documento de plan de trabajo
- Matriz de pruebas
- Documento de casos de pruebas
- Consolidado de resultados

3.3.4.4 Fase de evaluación

Al concluir la etapa de ingeniería, el analista de negocios procedió a realizar las pruebas correspondientes, de las cuales surgió una larga lista de errores identificados principalmente relacionados con la lógica de las reglas del negocio, así como también con el alcance de los requerimientos. Debido a la gran cantidad de errores identificados por el analista de negocio, se aplicó un proceso integral de validación y verificación, incluyendo al cliente a través de su equipo funcional de aplicación para descartar falsos errores y tener una lista real antes de planear los esfuerzos de la siguiente vuelta.

3.3.5 Vuelta 5: Alcance agregado y transición al nuevo sistema

3.3.5.1 Fase de planificación

Como resultado del ejercicio de validación y verificación en la vuelta anterior, se identificó un número considerable de falsos errores, dado que se trataba de alcance agregado no especificado apropiadamente desde el inicio del proyecto. A causa de que el cliente suponía la implementación de estos requerimientos no contemplados inicialmente, procedió a calificarlos como errores. Después de las aclaraciones correspondientes, se genera un acuerdo para agregar los nuevos requerimientos al proyecto ajustando el plan inicial.

Por estas razones, los objetivos de esta nueva vuelta fueron los siguientes:

- Corregir los errores reales de la liberación anterior para cumplir con lo requerido por el cliente

- Agregar los nuevos requerimientos al proyecto de tal manera que exista el mínimo impacto en el desarrollo.

Las principales restricciones de esta vuelta fueron relacionadas a los recursos humanos disponibles, pues ahora se trataba de más requerimientos a desarrollar en el mismo tiempo establecido. En consecuencia, el presupuesto disponible era limitado para contratar más desarrolladores. Finalmente, el tiempo permanecía como limitante, pues a pesar del alcance agregado la fecha de entrega era inamovible.

3.3.5.2 Fase de análisis de riesgo

El hecho de incorporar nuevos requerimientos a un proyecto en curso genera diversas fuentes de riesgo. Por un lado, puede provocar inconsistencia entre los requerimientos ya existentes, ya que los nuevos requerimientos pueden estar en contra de la lógica de negocio original o no ser compatibles con el sistema. Respecto a la programación de los objetos

existe el riesgo de generar problemas de versiones en los componentes de código, pues es posible que varios requerimientos utilicen las mismas clases y/o métodos, provocando que algunas funcionalidades ya implementadas y probadas sean modificadas negativamente o incluso borradas.

Por otro lado, se presenta el riesgo de no contar con un ambiente de pruebas, pues por política interna del cliente, existe un programa de actualización de servidores durante el transcurso de esta vuelta al modelo en espiral.

3.3.5.3 Fase de ingeniería

Referente a la actualización de los servidores de pruebas, se tuvieron varias reuniones para planear las actividades, de tal manera que no afectaran el flujo del desarrollo del SIGP. Se acordó aplicar la ventana de mantenimiento en fin de semana para no impactar la disponibilidad del ambiente.

Ahora con los nuevos requerimientos se procedió a actualizar la documentación existente para que incluyera el alcance agregado y de manera simultánea se ejecutó una revisión de las especificaciones técnicas y funcionales para evitar incompatibilidades con los requerimientos ya implementados. Afortunadamente la incorporación del nuevo alcance no requirió volver a diseñar o construir lo ya implementado, pues sólo se ampliaban algunas funcionalidades manteniendo su esencia, por lo que la implementación en código fue ejecutada. Respecto a los errores reportados en la vuelta anterior, se trabajó exhaustivamente para eliminarlos, coordinando tanto a los desarrolladores como al equipo de pruebas. De esta manera se aseguró la calidad en la implementación de los requerimientos. A estas alturas del proyecto todos los requerimientos ya estaban implementados e integrados en un ambiente de pruebas adecuado. Debido a esta situación

fue factible ejecutar pruebas de desempeño al sistema, con el propósito de ver su respuesta y hacer las modificaciones pertinentes de ser necesario.

3.3.5.4 Fase de evaluación

Dado que ya se contaba con una versión completamente funcional del SIGP, la fase de evaluación en esta vuelta del modelo en espiral requirió de un equipo robusto de usuarios claves y líderes funcionales por parte del cliente. Así, se busca someter a nuestro sistema a pruebas de flujos de trabajo reales manipulados por las personas que lo utilizarán día a día. Después de dos semanas intensas de pruebas y evaluaciones por parte de un equipo funcional compuesto por 15 personas, se obtuvieron resultados satisfactorios, donde se reportaron sólo algunos defectos mínimos que no representan una amenaza en contra de la liberación de nuestro sistema, siendo de esta manera el punto de partida para el plan de despliegue del SIGP en ambiente de producción.

3.3.6 Vuelta 6: Liberación final

3.3.6.1 Fase de planificación

A pesar de tener el proyecto prácticamente terminado, aún restaba corregir los errores menores y planear las actividades relacionadas al despliegue del SIGP en ambiente de producción. El objetivo principal fue lograr una liberación del sistema de manera correcta sin problemas de configuración y efectuar la transición del sistema previo al nuevo sistema sin impactos en la disponibilidad del servicio.

La principal restricción consistió en efectuar la transición sin impactar a los usuarios en sus operaciones cotidianas. Asimismo, la coordinación con los equipos de infraestructura

representaba una limitante, pues de no tener el apoyo de todos los involucrados el plan de liberación podía verse afectado.

Las alternativas para efectuar el despliegue de nuestro sistema fueron coordinar la liberación en un día hábil, pero en horario no laborable o efectuarla en fin de semana, mediante la negociación con los equipos de infraestructura.

3.3.6.2 Fase de análisis de riesgo

De los principales riesgos en la liberación del SIGP fue la posibilidad de que la nueva configuración, tuviera problemas al momento de desplegar la aplicación en los servidores de producción. Por otro lado, también era latente el riesgo de ejecutar los scripts o actividades en el orden incorrecto causando fallas importantes en el sistema. Asimismo, era probable que el tiempo dedicado al despliegue de la aplicación no fuera suficiente y causara intermitencia en el servicio.

3.3.6.3 Fase de ingeniería

La primera actividad en la fase de ingeniería fue la corrección de los últimos errores encontrados en las pruebas ejecutadas por el equipo funcional. Dado que se trataba de defectos menores, fueron rápidamente atendidos por el equipo de desarrollo y verificados por el cliente con resultados positivos.

En seguida se trabajó en la generación del plan de despliegue del SIGP, para definir cada una de las tareas necesarias y así lograr una transición adecuada. En el plan se definió el orden de las tareas, los scripts que deben ejecutarse, las adaptaciones y cambios en los

archivos de configuración y los detalles de conexión a otros sistemas y a bases de datos con las que nuestro sistema interactúa.

Una vez completado el plan de despliegue, se efectuaron tres simulaciones de la liberación del SIGP en el ambiente de pruebas disponible para corroborar que todas las tareas se definieron correctamente.

Finalmente, se desplegó la aplicación en ambiente de producción en un fin de semana con el propósito de no impactar las operaciones diarias de los usuarios. A parte de un par de errores de configuración, que fueron corregidos al momento, no hubo ningún problema al desplegar nuestro sistema

3.3.6.4 Fase de evaluación

Al tener el SIGP en ambiente de producción, se efectuaron las evaluaciones pertinentes por parte del cliente, con el apoyo de los líderes funcionales y los analistas de negocios para confirmar el adecuado funcionamiento del sistema. A parte de algunas preguntas de los evaluadores del funcionamiento de la aplicación, no se encontraron problemas en la misma y se notificó de la liberación a todos los usuarios para hacerles saber que la nueva versión del sistema había sido entregada y se encontraba disponible para su utilización. De esta manera, el desarrollo de software bajo el modelo en espiral culmina. Lo siguiente por hacer consiste en generar e implementar un plan de mantenimiento para cubrir las necesidades de la aplicación en caso de ocurrir eventualidades o prevenirlas.

Capítulo 4

Desarrollo basado en casos de uso

El propósito de los ejemplos ilustrados a continuación consiste en mostrar las integraciones funcionales del proyecto de desarrollo, con el fin de brindar una mejor comprensión y entendimiento del trabajo realizado. Las especificaciones funcionales son elaboradas con notación UML en su comportamiento y estructura. De esta manera se identifican los elementos básicos del diseño involucrados en el cumplimiento de la funcionalidad requerida. Se busca hacer una descomposición del flujo de elementos descritos en el caso de uso. Después del reconocimiento de las partes y responsabilidades se formaliza la estructura con un diagrama de clases. También se identifican patrones que permitan adaptar los elementos de software para reutilizarlos y facilitar el mantenimiento.

4.1 Ejemplo 1 Explicación del requerimiento Y

4.1.1 Preámbulo

El requerimiento Y del SIGP está relacionado a un reporte de expiración. Dicha funcionalidad es totalmente nueva; el proceso previo era realizado a través de requerimientos enviados al equipo de soporte de la aplicación. Antes de la implementación del requerimiento, la información del reporte era obtenida directamente la base de datos por medio de consultas manuales, sin la intervención de la aplicación.

4.1.2 Alcance de alto nivel del requerimiento

El objetivo fue agregar un nuevo reporte que maneje información de expiración de cumplimiento de ciertas etapas de los proyectos administrados en el SIGP. A continuación se enlistan las etapas que los proyectos en el SIGP tienen:

- Etapa R3. Definida para la recopilación de información preliminar del proyecto.

- Etapa R4. Utilizada preparación de documentos relacionados a las órdenes de adquisición del proyecto.
- Etapa de conciliación financiera. Encargada de obtener los estados de resultados financieros del proyecto.
- Etapa AR. Se trata de una etapa presente en algunos proyectos, con requerimientos sencillos que no requieren del proceso completo de administración a través del SIGP. Esta etapa ha sido creada para simplificar su seguimiento.
- Etapa de carta de apertura. Se define el contrato final del proyecto, previamente aprobado por todas las entidades del negocio involucradas (administradores, líderes de proyecto, proveedores, etc.).
- Etapa de cierre. Encargada de las actividades relacionadas a la culminación del proyecto.

El reporte relacionado a este requerimiento cuenta con los siguientes criterios de búsqueda:

- Tipo de reporte
    - Expiración R4
    - Conciliación financiera
    - Espera de cierre
    - Excepción AR
    - Expiración de carta de apertura
- Segmento
    - Gas
    - Generator
    - Steam

- Administrador del proyecto

Las columnas básicas serán como se muestra en la Tabla 4.1

Tabla 4.1 Estructura del reporte de expiración

Segmento	Proyecto	Admón. del Proy	Analista Financiero	Días

4.1.3 Dependencias con otros requerimientos del desarrollo:

Para prevenir inconsistencias al momento de implementar el reporte de expiración, se generó una matriz de dependencias con otros requerimientos, como se muestra en la tabla 4.2. Debido a que el reporte no realiza operaciones de escritura, sino únicamente de lectura, las dependencias son de tipo conceptual, por lo que su implementación no representa riesgo para los demás requerimientos.

Tabla 4.2 Dependencias del reporte de expiración con otros requerimientos

Requerimiento	Presentación	Lógica de negocio
Requerimiento B2	-	El status del proyecto es una de las principales condiciones usadas en el reporte, por tanto interviene en la salida del reporte
Requerimiento P	-	La aprobación de R3 afecta directamente la salida del reporte
Requerimiento U	-	El tipo de Orden de Cambio “AR to SR” impacta directamente la salida del reporte

4.1.4 Descripción del requerimiento

La pantalla de búsqueda es mostrada en la figura 4.1. El usuario puede seleccionar una de los tipos de reporte ya mencionados. El usuario también es capaz de filtrar los resultados por segmento y/o administrador del proyecto. El reporte resultante puede ser mostrado en formato HTML o exportado en un archivo de Excel. El resultado en HTML divide la

información usando paginación. El reporte tiene una condición implícita que previene mostrar los proyectos abandonados o cerrados.

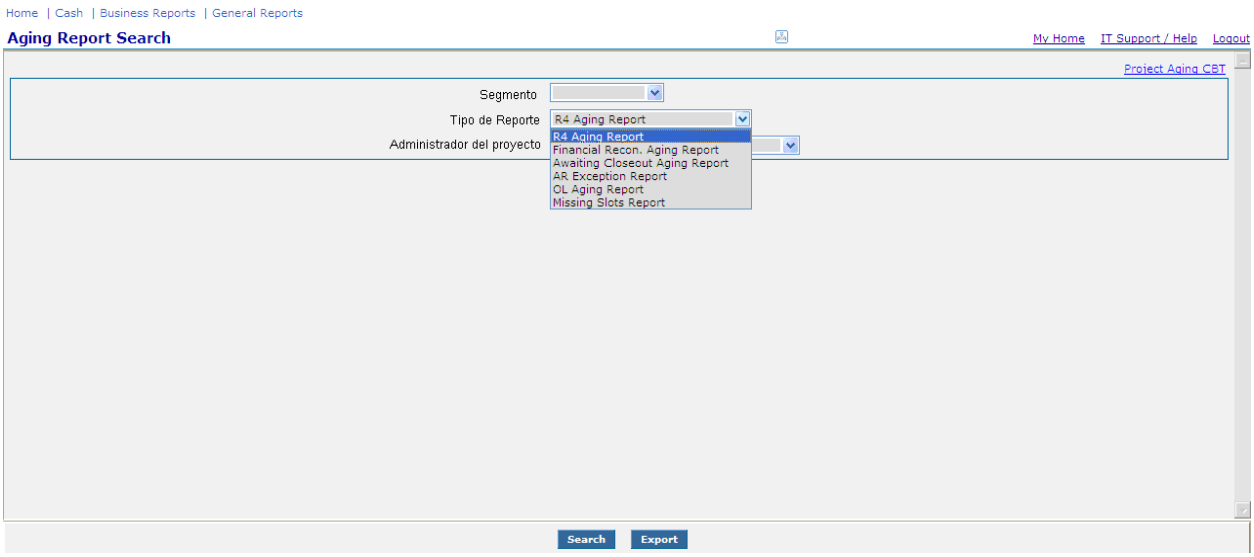


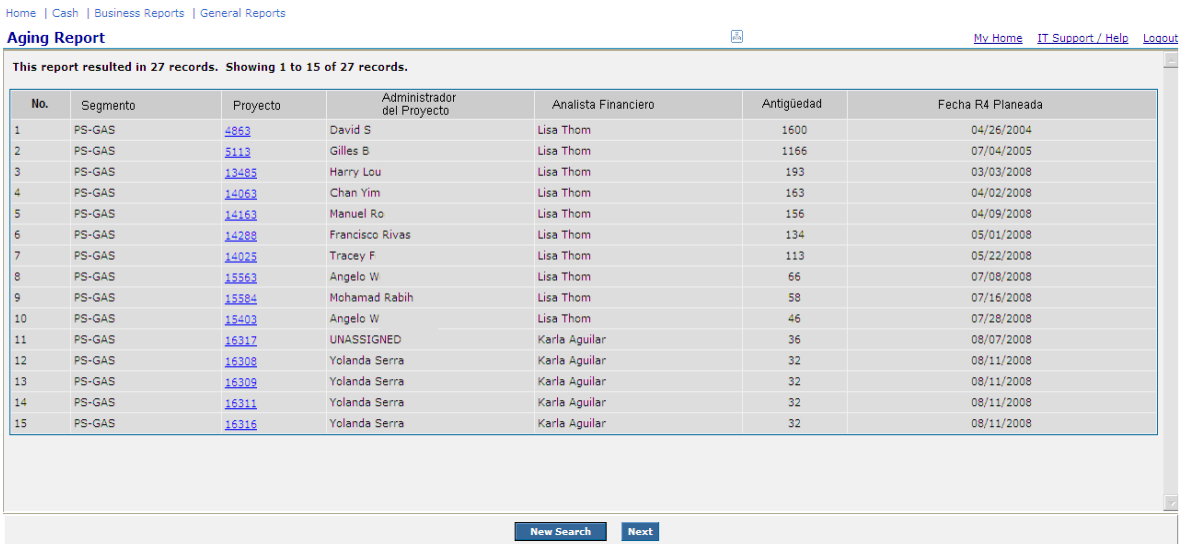
Figura 4.1 Pantalla de selección del reporte de expiración

El reporte ***Expiración R4*** muestra todos los proyectos activos con la fase R3 ya aprobada y la fase R4 aún pendiente de aprobación. Expiración R4 debe tener una columna adicional que muestre la fecha planeada de la fase R4 del proyecto.

La información mostrada es:

- Segmento
- Proyecto
- Administrador del proyecto
- Analista financiero
- Antigüedad (días) = hoy – fecha planeada de R4
- Fecha planeada de R4

La figura 4.2 muestra la presentación de los resultados del reporte Expiración R4.



No.	Segmento	Proyecto	Administrador del Proyecto	Analista Financiero	Antigüedad	Fecha R4 Planeada
1	PS-GAS	4863	David S	Lisa Thom	1600	04/26/2004
2	PS-GAS	5113	Gilles B	Lisa Thom	1166	07/04/2005
3	PS-GAS	13485	Harry Lou	Lisa Thom	193	03/03/2008
4	PS-GAS	14063	Chan Yim	Lisa Thom	163	04/02/2008
5	PS-GAS	14163	Manuel Ro	Lisa Thom	156	04/09/2008
6	PS-GAS	14288	Francisco Rivas	Lisa Thom	134	05/01/2008
7	PS-GAS	14025	Tracey F	Lisa Thom	113	05/22/2008
8	PS-GAS	15563	Angelo W	Lisa Thom	66	07/08/2008
9	PS-GAS	15584	Mohamad Rabih	Lisa Thom	58	07/16/2008
10	PS-GAS	15403	Angelo W	Lisa Thom	46	07/28/2008
11	PS-GAS	16317	UNASSIGNED	Karla Aguilar	36	08/07/2008
12	PS-GAS	16308	Yolanda Serra	Karla Aguilar	32	08/11/2008
13	PS-GAS	16309	Yolanda Serra	Karla Aguilar	32	08/11/2008
14	PS-GAS	16311	Yolanda Serra	Karla Aguilar	32	08/11/2008
15	PS-GAS	16316	Yolanda Serra	Karla Aguilar	32	08/11/2008

Figura 4.2 Pantalla de resultados del reporte de Expiración R4

El reporte de ***Conciliación financiera*** muestra todos los proyectos en status de Conciliación financiera. Debe tener una columna adicional que muestre la fecha de inicio de la Conciliación Financiera

Los siguientes detalles son incluidos en el reporte:

- Segmento
- Proyecto
- Administrador del proyecto
- Analista financiero
- Antigüedad (días) = hoy – fecha en que el proyecto cambio a status de Conciliación financiera
- Fecha de inicio

La pantalla de resultados en el SIGP se muestra en la figura 4.3.

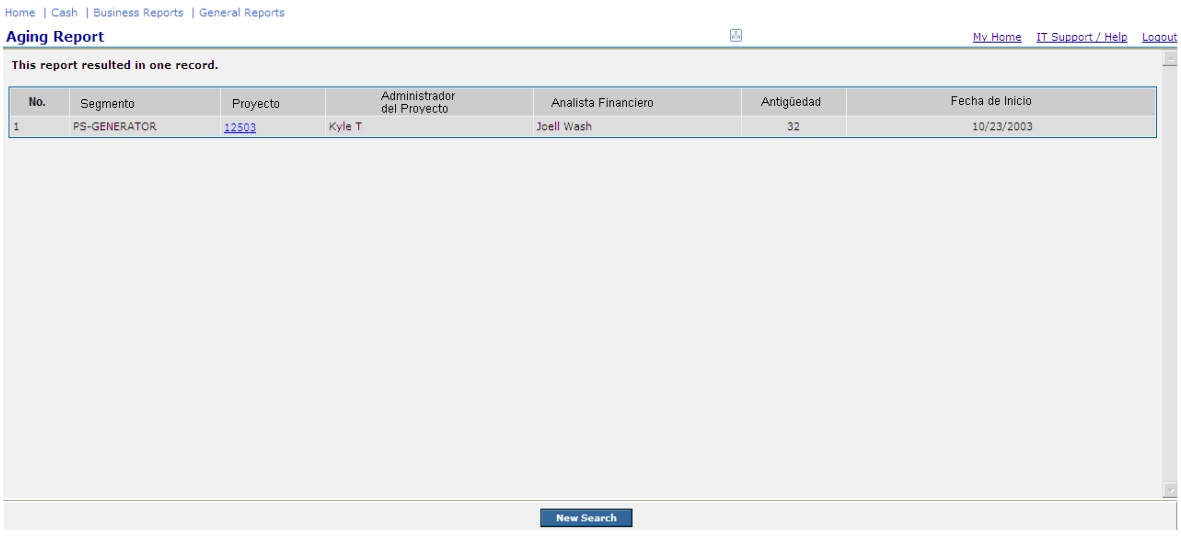


Figura 4.3 Pantalla de resultados del reporte de Conciliación financiera

El tipo de reporte ***Espera de cierre*** muestra todos los proyectos en status de Espera de cierre o Espera de administrador general. Este reporte debe tener una columna adicional para la fecha inicial de espera de cierre.

Los siguientes detalles son incluidos:

- Segmento
- Proyecto
- Administrador del proyecto
- Analista financiero
- Antigüedad (días) = hoy – fecha en que el proyecto cambió a status de Espera de cierre
- Fecha de inicio

En la figura 4.4 se presenta un ejemplo del set de resultados.

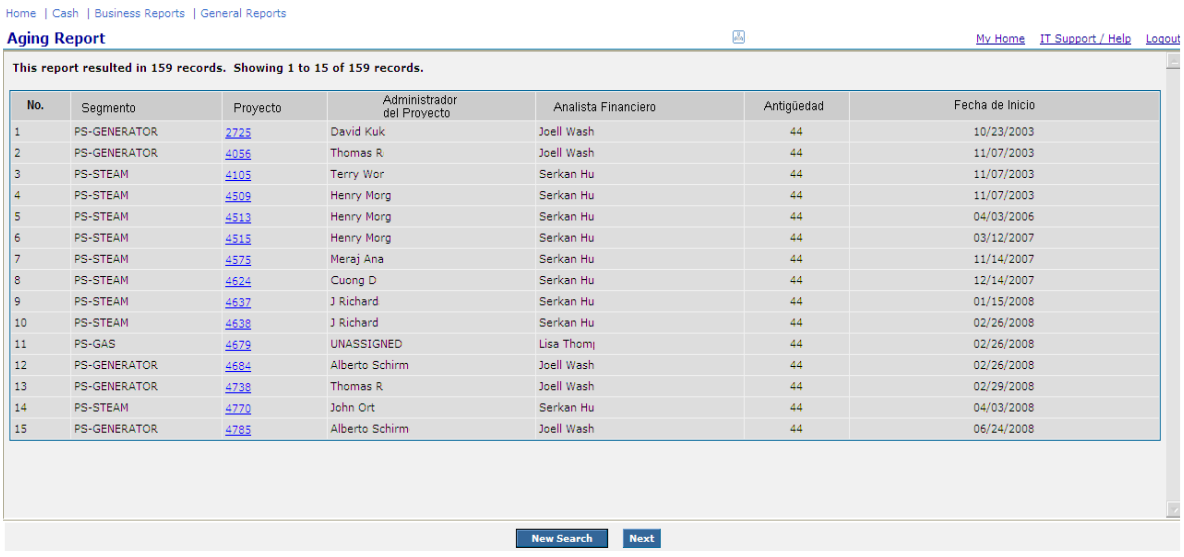


Figura 4.4 Pantalla de resultados del reporte Espera de cierre

El reporte de ***Expiración de carta de apertura*** muestra todos los proyectos con la fase R4 ya aprobada y en espera de aprobación de la carta de apertura. El reporte debe tener una columna adicional que muestre la fecha de aprobación de la fase R4.

Los siguientes detalles son incluidos:

- Segmento
- Proyecto
- Administrador del proyecto
- Analista financiero
- Antigüedad (días) = hoy – fecha en que la lista RM fue aprobada
- Fecha de aprobación de RM

Un ejemplo del reporte de Expiración de carta de apertura se muestra en la figura 4.5

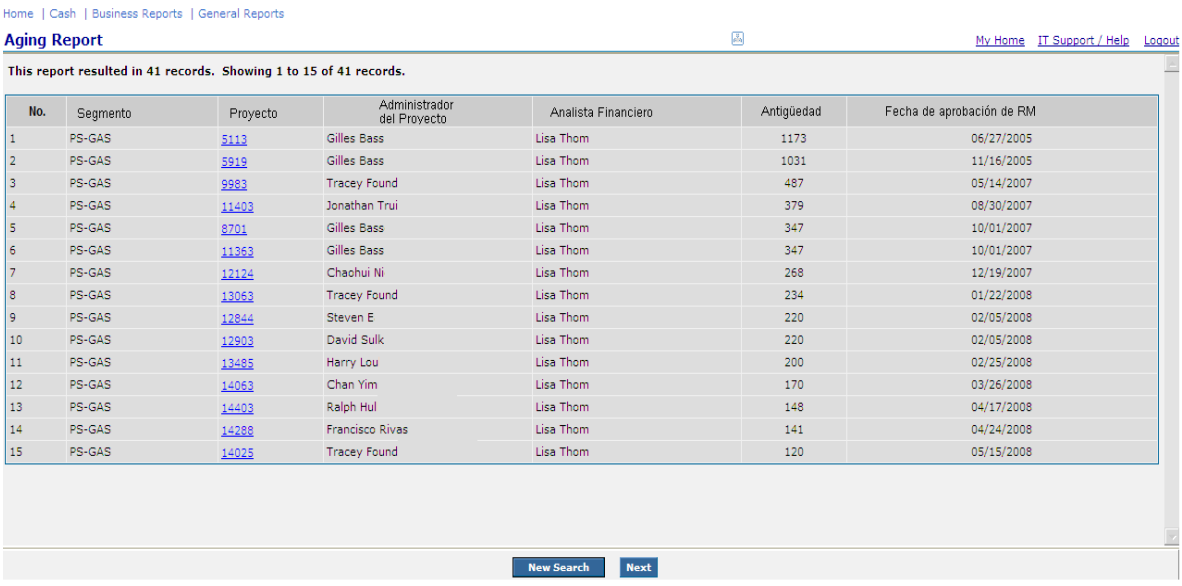


Figura 4.5 Pantalla de resultados del reporte Expiración de carta de apertura

Excepción AR debe tener las siguientes columnas adicionales:

- Fecha Real de R3
- Fecha Planeada de R3
- Indicador de existencia de orden de Cambio del tipo AR a SR
- Indicador de aprobación de Orden de Cambio por el Líder de Segmento

Proyectos sin Slots

Este tipo de reporte muestra todos los proyectos que no tengan ningún slot asignado. Los siguientes detalles son incluidos (véase figura 4.6):

- Segmento
- Proyecto
- Administrador del proyecto
- Analista financiero
- Antigüedad(Días) = Hoy – Fecha en que se creó el proyecto
- Fecha de creación

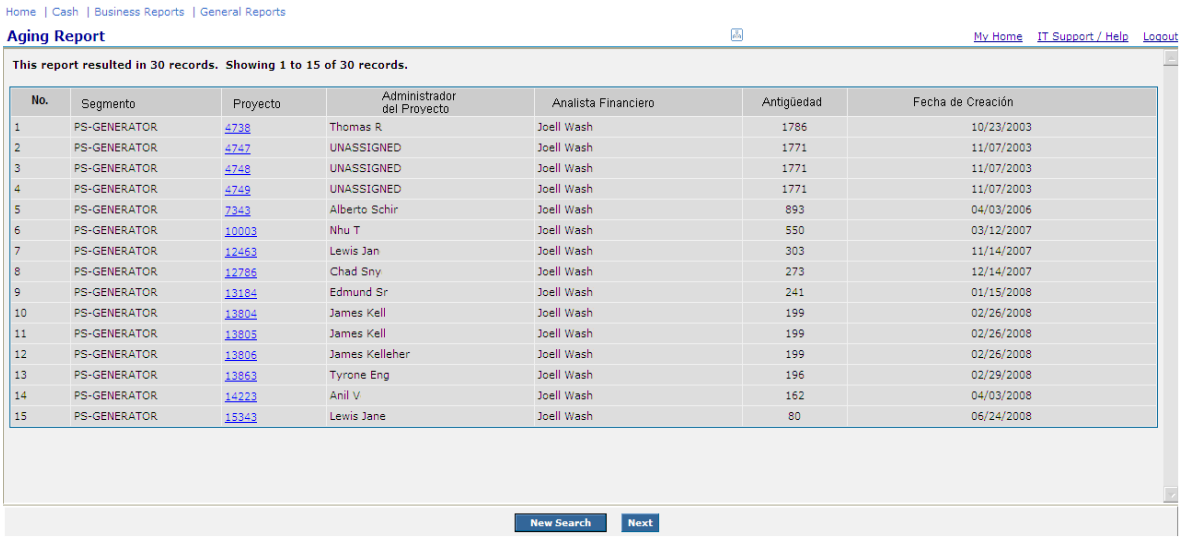


Figura 4.6 Pantalla de resultados de reporte Proyectos sin spots

4.1.5 Flujo de eventos

El caso de uso inicia cuando el actor selecciona la opción de “Reporte de expiración” del menú de “Reportes de negocio”. El sistema obtiene y presenta la pantalla de filtro del reporte con los catálogos existentes en la base de datos. El caso de uso termina cuando el usuario obtiene el reporte requerido, ya sea en la aplicación o en un archivo exportado en formato de Excel.

4.1.6 Diagrama de caso de uso

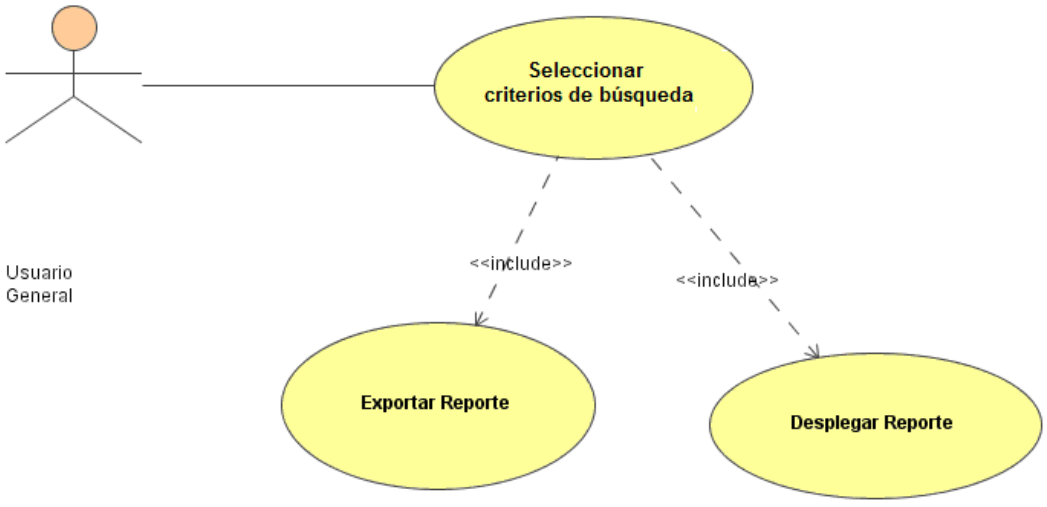


Figura 4.7 Diagrama de caso de uso

En la figura 4.7 se muestra el diagrama de caso de uso. Para una mayor comprensión del desarrollo de eventos de este caso de uso, se descomponen las secuencias e interacciones que intervienen en:

- Carga de la pantalla de búsqueda del reporte
- Buscar resultados
- Exportar resultados

4.1.7 Diagrama de secuencias

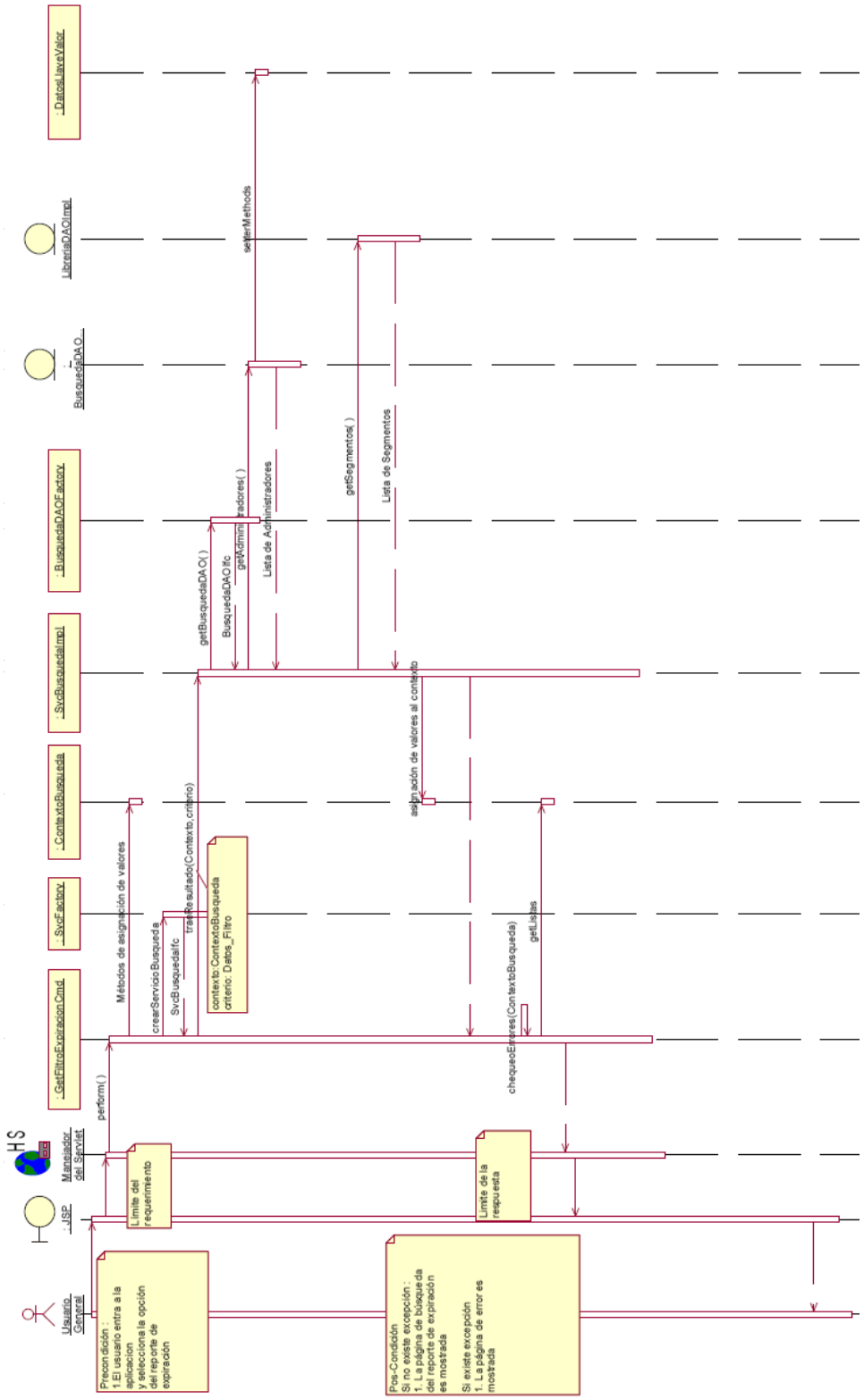


Figura 4.8 Diagrama de secuencias del reporte de expiración

En la figura 4.8 se muestra el diagrama de secuencias relacionado al reporte de expiración del SIGP. Asimismo, la tabla 4.3 muestra la descripción paso a paso del diagrama de secuencias.

Tabla 4.3 Detalle del diagrama de secuencias del reporte de expiración

Paso	Descripción
1	El usuario inicia la interacción al hacer click en el link del reporte
2	El archivo JSP manda un request al manejador del Servlet
3	La instancia getFiltroExpiracionCmd recibe el request
4	A partir de la instancia getFiltroExpiracionCmd se crea una instancia de ContextoBusqueda para almacenar los datos necesarios en la interacción
5	Se manda a llamar al SvcFactory para crear el servicio SvcBusquedaIfc
6	Una vez creado el servicio se utiliza el método traeResultado, pasando el contexto y el criterio de búsqueda
7	Con el método getBusquedaDAO se llama al objeto BusqeudaDAOFactory para generar el DAO BusquedaDAOImpl
8	A través del DAO se obtiene la lista de Administradores de la aplicación, haciendo una consulta a la base de datos
9	Haciendo uso del objeto LibreriaDAOImpl y del método getSegmentos se hace una consulta a la base de datos para obtener la lista de segmentos
10	Tanto la lista de segmentos como la lista de administradores son asignadas al contexto en SvcBusquedaImpl
11	El contexto con los valores obtenidos es regresado a getFiltroExpiracionCmd
12	Se hace un chequeo de excepciones para validar que los datos sean correctos
13	Los atributos necesarios son parte del Response del servlet para ser mostrados en la pantalla

Referente al diagrama de colaboración, la figura 4.9 muestra cómo se relacionan las clases que intervienen en la generación del reporte de expiración.

4.1.8 Diagrama de colaboración

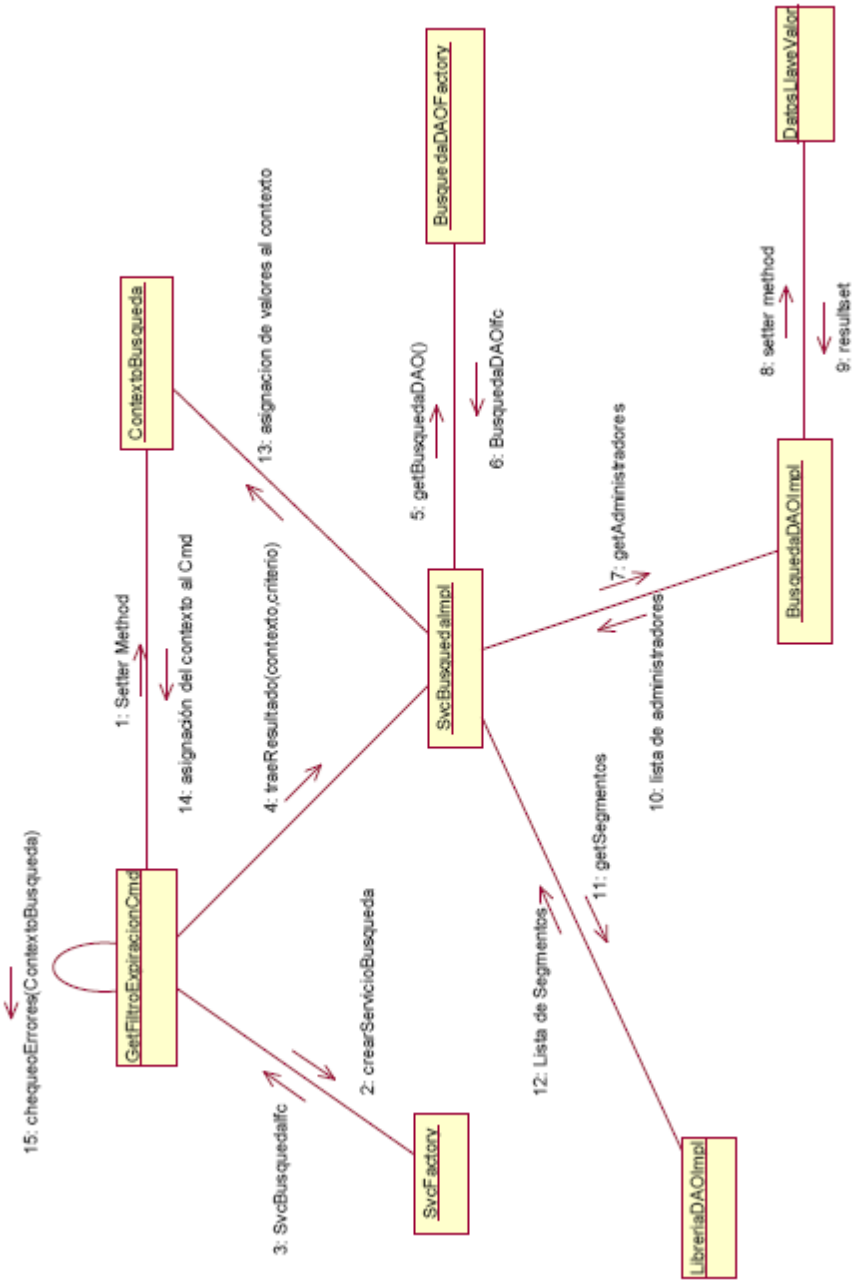


Figura 4.9 Diagrama de colaboración

4.2 Ejemplo 2 Explicación del requerimiento I

4.2.1 Preámbulo

El requerimiento I del SIGP cubre las necesidades de reconstruir una funcionalidad existente mediante un nuevo diseño de pantalla para hacerla más amigable al usuario y agregar controles dedicados a la preservación de la integridad de la información. El usuario ingresa a la aplicación y selecciona un proyecto, ya sea de la sección de favoritos o a través de una búsqueda en particular. El usuario selecciona la opción de detalles de subpartes, como se muestra en la figura 4.10.

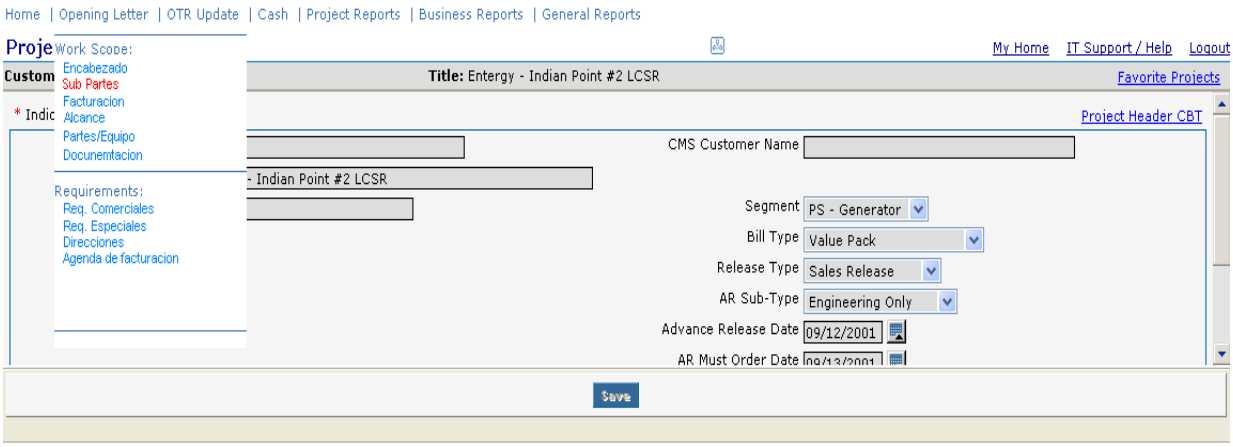


Figura 4.10 Menú de selección de pantalla

La pantalla de subpartes es mostrada y la información es clasificada por segmento, subparte y detalle de subparte, como se muestra en la figura 4.11. Esta clasificación representa la forma en que el sistema desplegaba la pantalla, antes de la implementación de la nueva versión del SIGP.

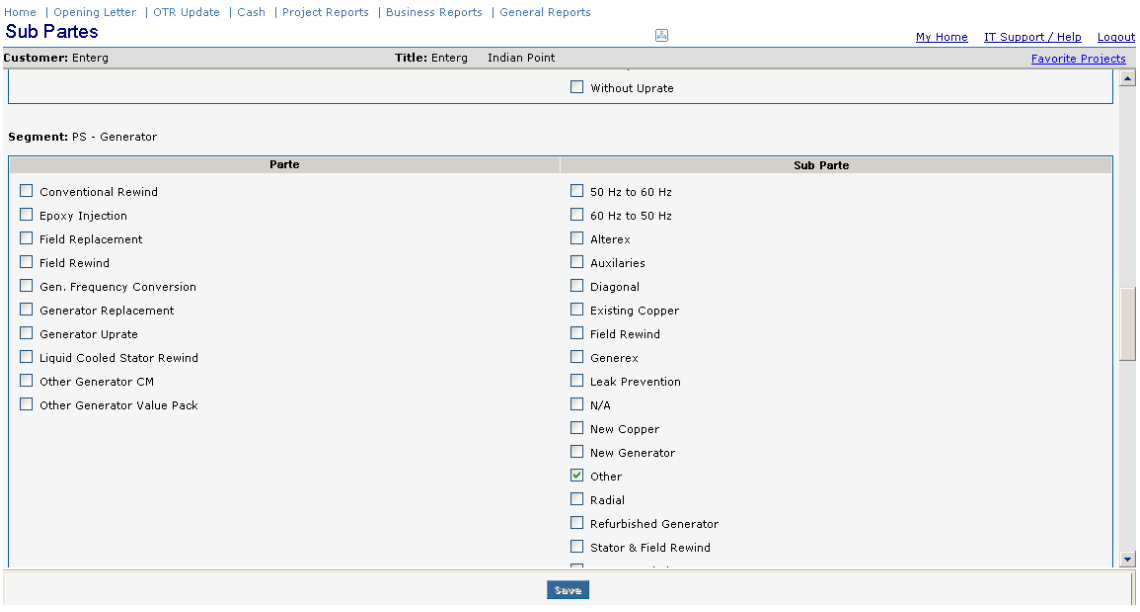


Figura 4.11 Pantalla de subpartes

4.2.2 Alcance de alto nivel del requerimiento

Se necesita que la pantalla de subpartes sea rediseñada y mejorada para ser mostrada en multiniveles de segmento, producto y subparte. Asimismo, se requiere un nuevo mapeo en los datos que incluya, tanto los datos existentes, como los que se ingresarán después de la entrega del proyecto.

4.2.3 Dependencias con otros requerimientos

La tabla 4.4 muestra las dependencias del requerimiento I con otras funcionalidades a implementar. En este caso, si existen fuertes dependencias con dos requerimientos, por lo que se registró en el proceso de gestión de riesgos para prevenir problemas de incompatibilidad con los demás requerimientos.

Tabla 4.4 Dependencias de la pantalla de Subpartes con otros requerimientos

Cambio	Presentación	Lógica de negocio
Requerimiento P	-	La aprobación de la fase R3 está condicionada a que sólo se pueda aprobar si existe al menos una subparte agregada al proyecto
Requerimiento Q	La pantalla de pronósticos de ventas debe mostrar correctamente el mapeo de las subpartes	-

4.2.4 Descripción del requerimiento

El usuario ingresa a la aplicación y selecciona un proyecto, ya sea de la sección de favoritos o a través de una búsqueda en particular. El usuario selecciona la opción de detalles de subpartes como se muestra en la figura 4.12.

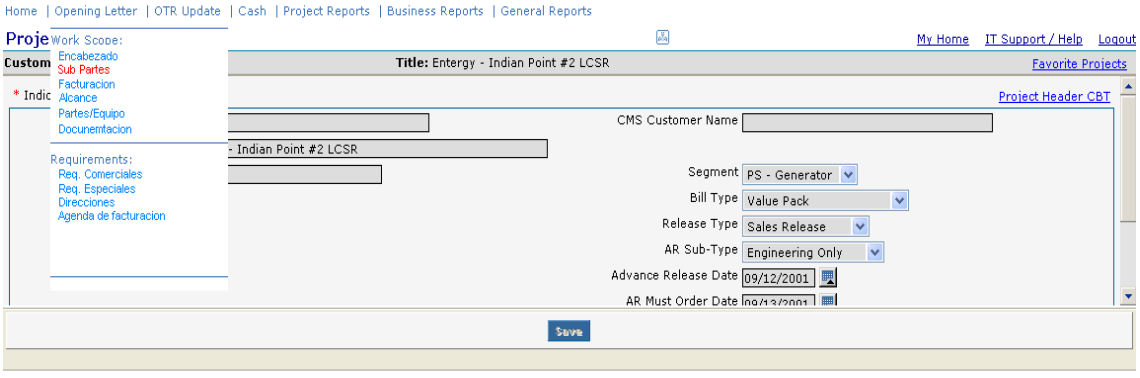


Figura 4.12 Menú de la aplicación

La pantalla de subpartes es mostrada y la información es clasificada por Segmento, Producto y subparte. Los datos ya agregados al proyecto son mostrados en una tabla como se muestra en la figura 4.13.

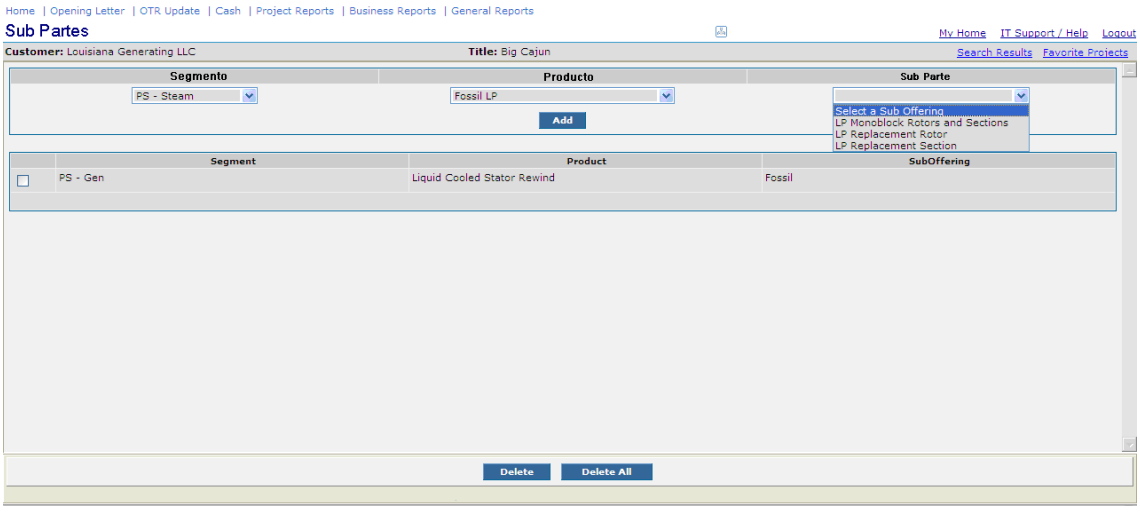


Figura 4.13 Nueva pantalla de subpartes

4.2.5 Flujo de eventos

El caso de uso inicia cuando el actor selecciona la opción subpartes del menú “Carta de Apertura” cuando un proyecto ya ha sido abierto. El sistema obtiene y presenta tres menús desplegables para los campos segmento, producto y subparte. También presenta una tabla para visualizar las subpartes ya agregadas al proyecto. El caso de uso termina cuando el usuario ha agregado o eliminado las subpartes necesarias en el proyecto en cuestión.

4.2.6 Diagrama de caso de uso

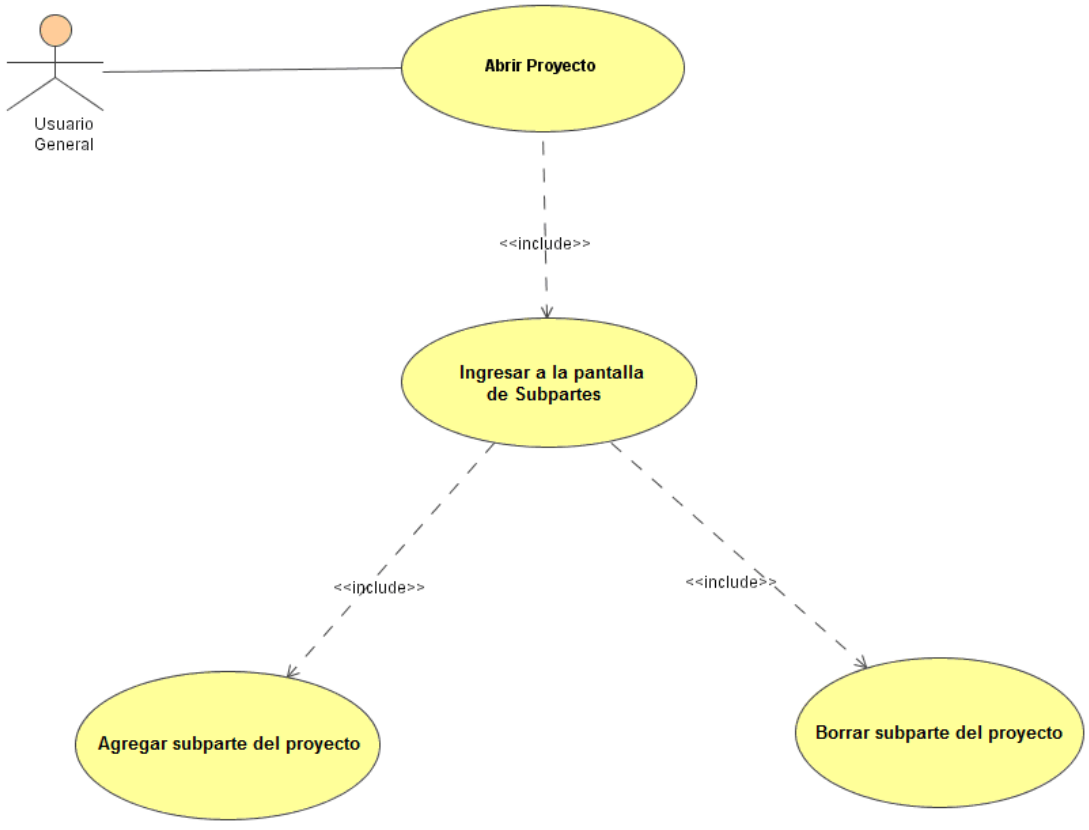


Figura 4.14 Diagrama de caso de uso

La figura 4.14 muestra el diagrama de caso de uso para el requerimiento relacionado a la pantalla de subpartes. Como se puede apreciar, las acciones posibles son ver las subpartes agregadas al proyecto, agregar más subpartes o eliminar existentes.

4.2.7 Diagrama de secuencias

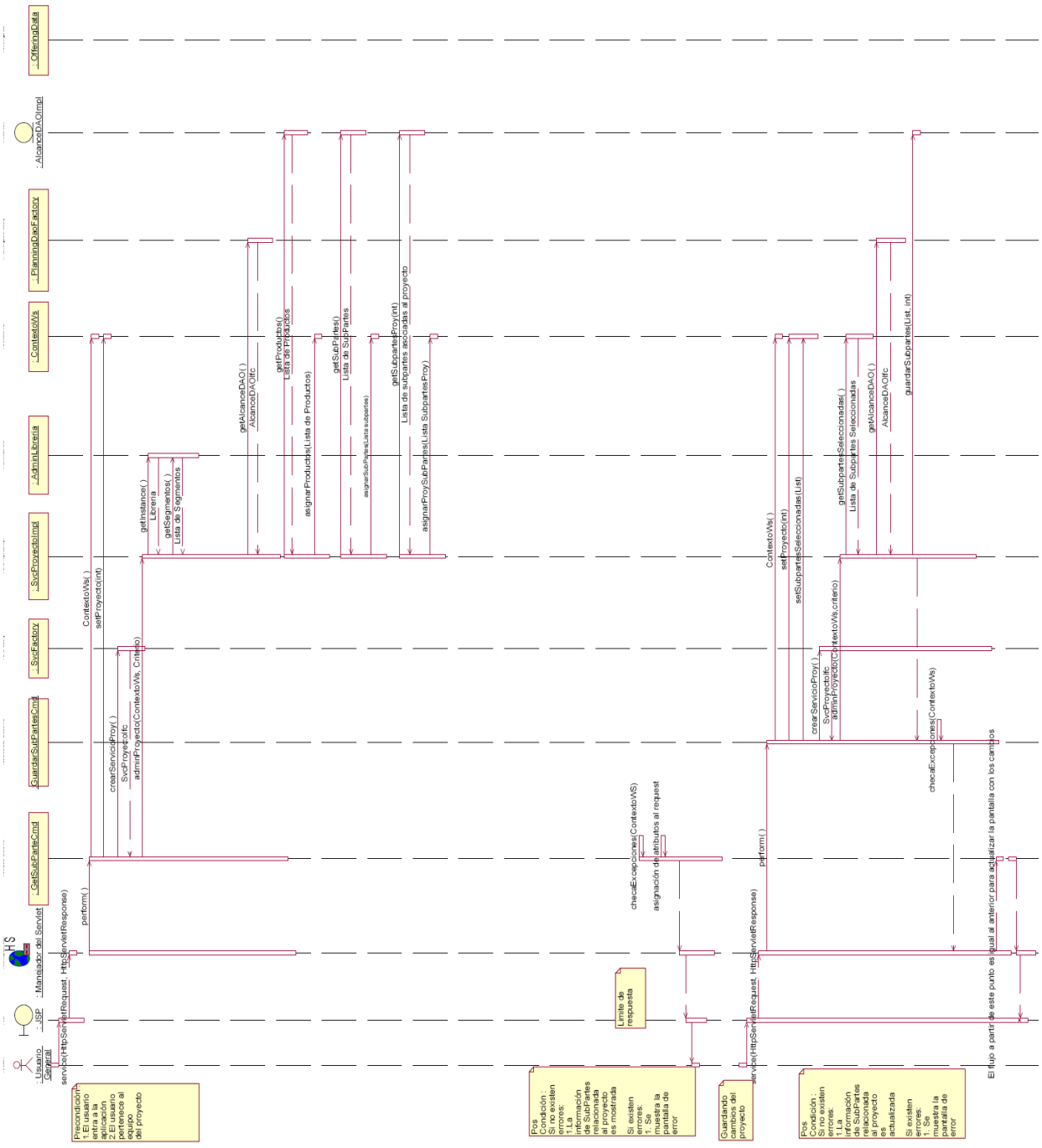


Figura 4.15 Diagrama de secuencias

En la figura 4.15 se muestra el diagrama de secuencias referente a la funcionalidad de subpartes. Cada paso en el proceso de la pantalla de subpartes es descrito en la tabla 4.5

Tabla 4.5 Detalle del diagrama de secuencias de la pantalla Subpartes

Paso	Descripción
1	El usuario inicia la interacción al hacer click en el link de subpartes
2	El archivo JSP manda un request al manejador del Servlet
3	La instancia getSubParteCmd recibe el request
4	A partir de la instancia getSubParteCmd se crea una instancia de Contextos para almacenar los datos necesarios en la interacción
5	Se manda a llamar al SvcFactory para crear el servicio SvcProyectoIfc
6	Una vez creado el servicio se utiliza el método getSegmentos hacia el objeto AdminLibreria.
7	Con el método getAlcanceDAO se llama al objeto PlanningDAOFactory para generar el DAO AlcanceDAOImpl
8	A través del DAO se obtiene la lista de Productos de la aplicación, haciendo una consulta a la base de datos
9	De manera similar se obtiene la lista de subpartes de la aplicación, haciendo una consulta a la base de datos
10	Las listas de segmentos, productos y subpartes son asignadas al contexto en SvcBusquedaImpl
11	El contexto con los valores obtenidos es regresado a getSubParteCmd
12	Se hace un chequeo de excepciones para validar que los datos sean correctos
13	Los atributos necesarios son parte del Response del servlet para ser mostrados en la pantalla
14	Una vez mostrada la pantalla existe una segunda interacción cuando el usuario desea modificar la lista de subpartes. Para ello, se utiliza el comando GuardarSubPartesCmd
15	Del request tipo Post se extrae el ID del proyecto y las subpartes seleccionadas para asignarlas al contexto
16	Nuevamente se invocan los Servicios y DAOs pasando los valores del request
17	Con el método guardarSubpartes definido en AlcanceDAOImpl se abre una conexión a la base de datos para hacer el INSERT respectivo
18	Se actualiza la pantalla con el mismo comportamiento descrito del paso 2 al 13

Finalmente, se presenta en la figura 4.16 el diagrama de colaboración entre los componentes de software desarrollados para la implementación de la nueva pantalla de subpartes.

4.2.8 Diagrama de colaboración

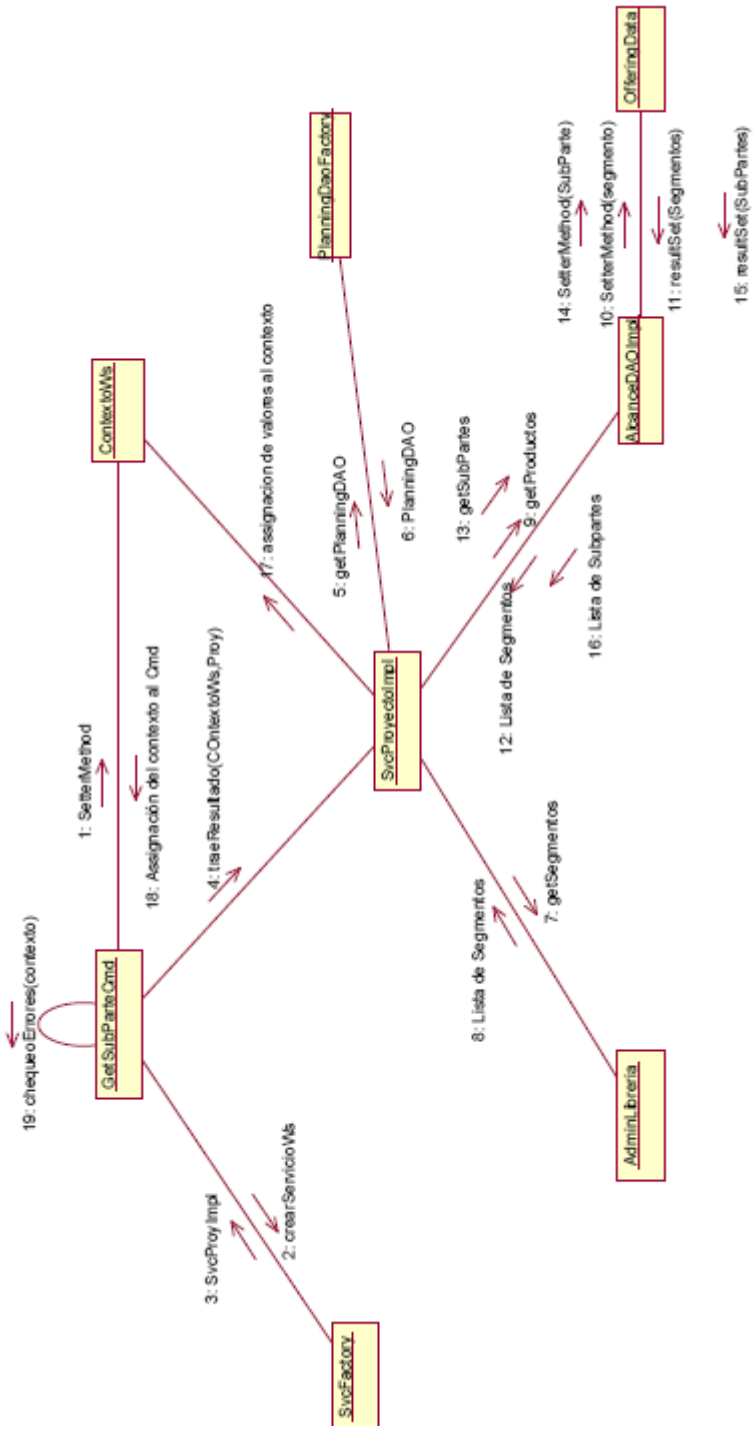


Figura 4.16 Diagrama de colaboración

4.3 Resultados y entregables

A continuación se mencionan los resultados y entregables después de más de 10 meses de dedicación al desarrollo de la versión 5.0 del SIGP.

Referente a la documentación del sistema, se generaron 45 documentos detallados de análisis de requerimientos, así como también 45 documentos de pruebas. Se modificaron 6 documentos funcionales, un documento de arquitectura, un documento de diseño, un documento de configuración e instalación y un documento de soporte a producción para mantenerlos actualizados conforme a la nueva versión.

Referente a la infraestructura de la aplicación, se realizó una actualización del servidor web de la aplicación de la versión Jboss 3.2.1 a la versión 4.0. También se hizo una migración de la aplicación a otros servidores físicos más recientes y con mejor capacidad de servicio. También se hizo una actualización en la base de datos de la versión Oracle 9.2 a la versión Oracle 10g.

Para la versión 5.0 del SIGP hubo una constante restructuración de datos, dado que los requerimientos involucraron nuevos mapeos en diversos campos de la base de datos así como también cambios de tipos de datos. Las actividades relacionadas en este proyecto incluyeron migraciones de datos, alteración de tablas para acomodar los cambios de acuerdo a los requerimientos, así como también cambios en código fuente para recibir los nuevos tipos de datos. También en coordinación con los equipos de integraciones se trabajó en conjunto para mantener la integridad de los sistemas externos con la nueva versión del SIGP.

Se realizaron las pruebas unitarias, pruebas de regresión, pruebas de desempeño y pruebas de carga pertinentes para la versión 5.0 del SIGP, hasta que se validaron todos los casos de prueba.

En este proyecto se hizo uso de herramientas automáticas de revisión de código para generar los reportes de cada objeto y sus fallas, en cuanto a seguridad y estándares específicos del cliente. Después de hacer las correcciones y adaptaciones necesarias, se requirió de la aprobación del equipo especial de Revisión de código para que la aplicación pueda ser entregada. Asimismo, se hizo una revisión de seguridad en la aplicación para disminuir las posibles vulnerabilidades después de las modificaciones en el código. El reporte final y aprobaciones forman parte de la documentación.

También se realizaron monitoreos en la aplicación antes y después de la entrega de la nueva versión. Los comparativos se muestran a continuación.

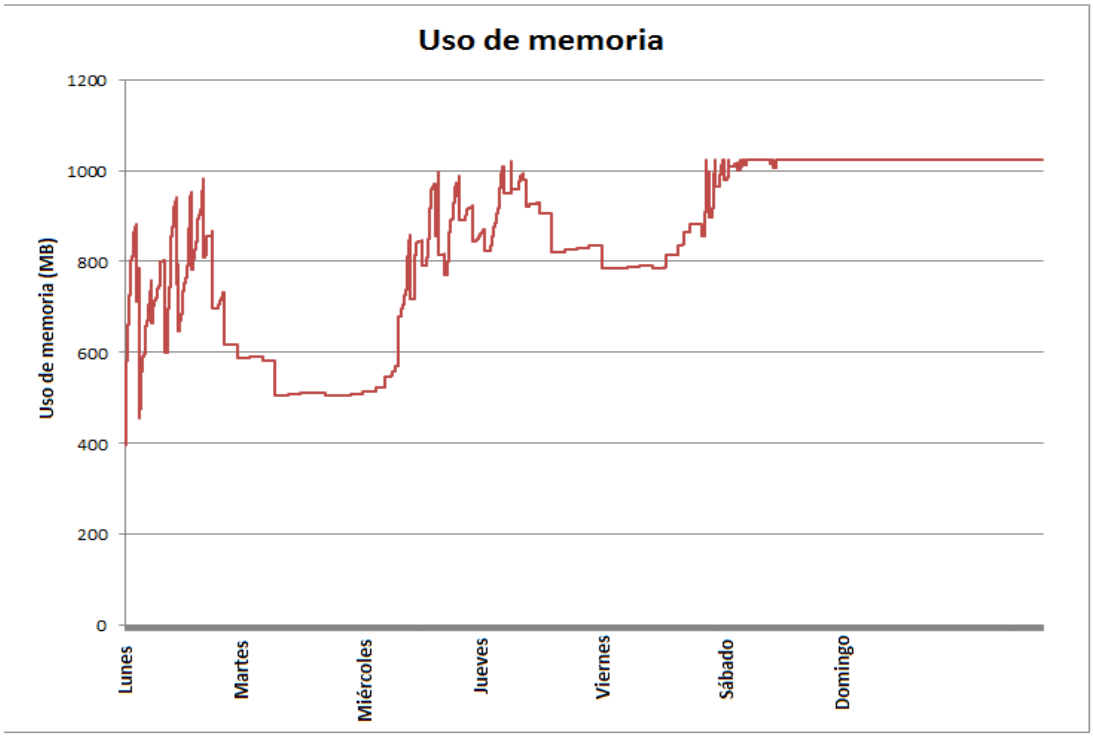


Figura 4.17 Uso de memoria de la versión 4.0

Figura 4.18 Uso de memoria de versión 5.0

Se puede apreciar en la figura 4.17 que los niveles de memoria utilizada en los servidores aumentaban drásticamente conforme pasaban los días, lo cual impactaba negativamente al desempeño del sistema. Una vez implementado el SIGP version 5, se ve en la figura 4.18 que el manejo de memoria del sistema fue mejorado pues los niveles de utilización se redujeron prácticamente a la mitad.

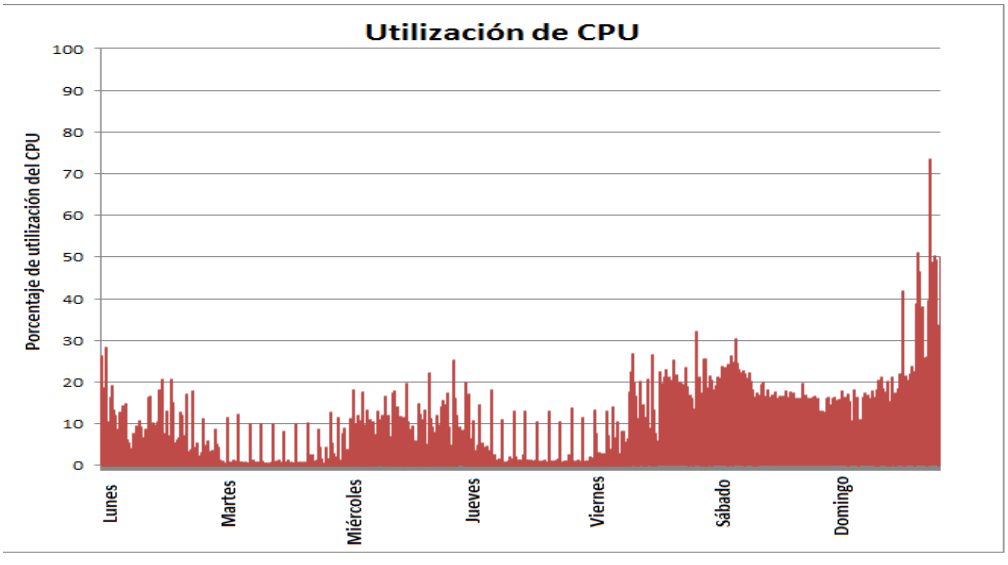


Figura 4.19 Utilización del CPU de la versión 4.0

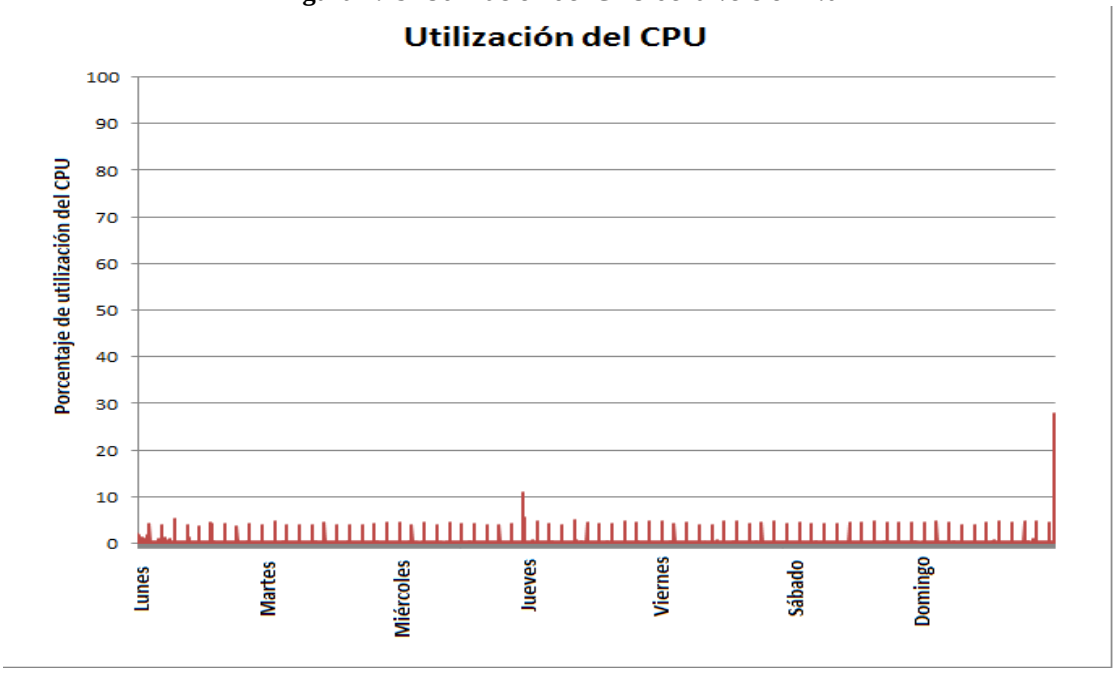


Figura 4.20 Utilización del CPU de la versión 5.0

En lo que respecta a la utilización del CPU, sucede algo similar al manejo de memoria. Previo a la implementación del SIGP versión 5, se muestra en la figura 4.19 que los CPUs del servidor tendían a saturarse conforme la semana de actividades laborales avanzaba. No así, en la figura 4.20 se aprecia que la carga de los CPU's fue reducida y el desempeño fue mejorado.

Después de la entrega del sistema, se dio soporte de post-producción durante un mes en caso de que alguna falla fuera detectada. Sólo hubo necesidad de corregir 3 defectos cosméticos en la aplicación que no comprometieron la integridad de la información.

Se otorgó el entrenamiento pertinente al equipo destinado al soporte y mantenimiento de la aplicación durante 2 semanas para asegurar el total entendimiento de las nuevas funcionalidades.

Capítulo 5
Mantenimiento del sistema

5.1 Preparación del sistema para el mantenimiento

Después de todo el proceso de desarrollo, se llevó a cabo la asignación SIGP, a un equipo de soporte para brindarle el mantenimiento necesario. Las siguientes secciones explican cómo se maneja este proceso.

5.2 Estructura del área mantenimiento y soporte de aplicaciones

La empresa en la que el SIGP es utilizado está estructurada en diferentes áreas de negocios a nivel mundial. Esta estructura de negocio requiere a su vez un modelo de mantenimiento que pueda atender las necesidades y problemas a tiempo y a bajo costo y que cubra geográficamente todas las regiones, donde se necesite del servicio. Dentro de la organización existe el Área de Mantenimiento y Soporte Global (AMSG) dedicada a atender estas necesidades. Uno de los objetivos principales dentro de AMSG es contar con un equipo de alto desempeño que administre un portafolio de aplicaciones, que manejan máximos valores del negocio y proveer soporte internacional, así como también estandarizar procesos.

Para conservar los niveles de servicio se debe ser constante en el propósito de mejora del producto, con el objeto de ser competitivo y mantenerse en constante actualización. De esta manera es posible proveer un valor agregado a los clientes, empleados e inversionistas.

El trabajo en conjunto de los miembros del equipo ayuda a facilitar la administración general del proyecto y mejorar todo el conjunto de aplicaciones soportadas mediante el uso de metodologías comunes y consistentes ya antes mencionadas (Six Sigma y CMMI).

Otro de los objetivos del programa de mantenimiento es dar un continuo soporte de aplicaciones y mejorar su desempeño, haciéndolas más rápidas y estables con alta eficiencia dentro del alcance de horarios y servicios que se estipula en el contrato.

Para tener una mejor organización en AMSG, se cuenta con diferentes áreas y grupos de trabajo con lo que las empresas proveedoras de servicios de outsourcing pueden dirigir mejor los requerimientos y tener una comunicación más eficiente.

Después de que un producto de software ha sido desarrollado y entregado, como es el caso del SIGP, la siguiente etapa en la ingeniería de software es el mantenimiento. Dentro del esquema de AMSG se debe seguir un proceso para tomar posesión del producto y brindarle servicios de mantenimiento y soporte.

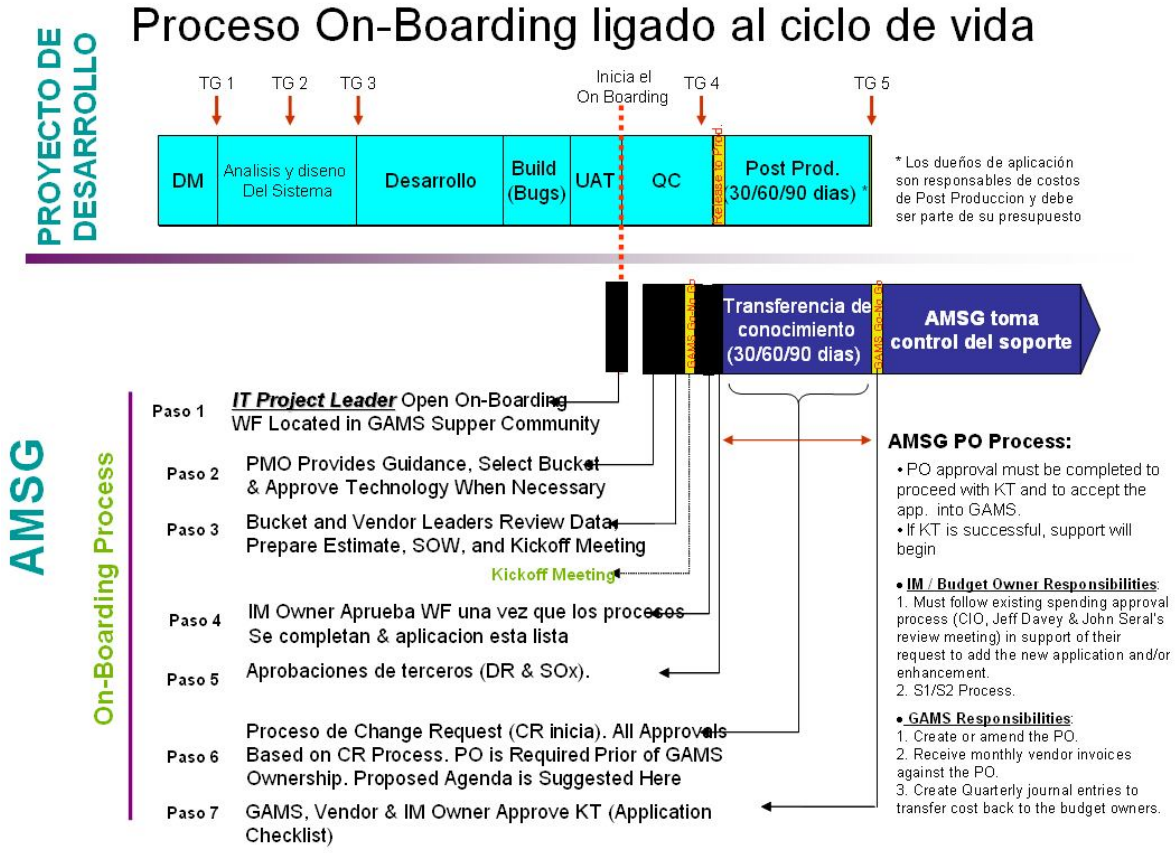


Figura 5.1 Proceso de on-boarding ligado al ciclo de vida

Tomando como referencia las librerías de ITIL en su versión 3, se utilizan los procesos definidos en la etapa de Transición del servicio para definir las actividades necesarias que permitan implementar los aspectos relacionados al mantenimiento del sistema.

El primer proceso, denominado Gestión de configuración y activos, se refiere a la identificación, registro y control de los activos que se relacionan a un servicio para asegurar su integridad y correcta utilización. Como se muestra en la figura 5.1, inicialmente el líder de proyecto encargado del grupo de expertos dedicados al mantenimiento y soporte, crea un flujo de trabajo dentro de la comunidad AMSG para asentar el requerimiento de soporte de la aplicación.

El segundo proceso que define ITIL v3 aborda la Gestión del cambio, el cual pretende asegurar la adecuada evaluación, autorización, priorización y planeación de las actividades necesarias para establecer el servicio deseado. Para ello, la Oficina de Administración de Proyectos interna guiará el proceso mediante la selección del área donde la aplicación será atendida y aprobará la tecnología empleada en caso de ser necesario.

Una vez que se ha decidido quién se encargará de la aplicación, se hará una revisión entre el líder funcional de la aplicación y el líder del equipo de mantenimiento para analizar la información disponible, preparar una estimación, establecer un manifiesto de trabajo y calendarizar una junta de arranque del proyecto.

Ya que estos pasos se han completado, los responsables funcionales del lado del cliente dan su aprobación en el flujo de trabajo y la aplicación está lista para que sea provista de los servicios de soporte y mantenimiento. De ser requerido también se necesitará la aprobación de terceros.

Cuando las aprobaciones son obtenidas entonces se genera un contrato, en el cual se necesita la aprobación del dueño de la aplicación antes de que el equipo del AMSG tome posesión del producto de software. Finalmente, la transferencia de conocimiento técnico y funcional es discutida entre los líderes funcionales y el equipo de mantenimiento mediante una lista de control para evitar que algún punto importante deje de ser considerado

El proceso de Gestión del conocimiento definido por ITIL v3 ayuda a asegurar que la persona indicada tenga el conocimiento indicado para proporcionar un servicio. Para la creación del plan de mantenimiento del SIGP, se involucran tanto a los desarrolladores como a los expertos funcionales de la aplicación, para asegurar que los encargados en brindar el soporte y mantenimiento tengan el conocimiento completo y puedan mantener la aplicación sin ninguna dificultad, ya que este nuevo grupo de expertos será quien se enfrente a las dificultades y problemas que la etapa de mantenimiento y soporte involucra.

Cabe señalar que parte del entrenamiento de un experto consiste en conocer a la organización y dominar los procesos relacionados a la cuarta etapa del marco de trabajo de ITIL v3 denominada Operación del servicio, en la que se incluyen:

- Gestión de incidentes
- Gestión de problemas
- Cumplimiento de solicitudes
- Gestión de eventos
- Gestión de accesos

Relacionado a la interfaz de soporte, inicialmente se contaba con varias aplicaciones web en las que se daba el seguimiento a todos los requerimientos y desarrollos del cliente. Posteriormente, se hizo una migración a otra aplicación cliente-servidor en la que se

concentra toda la información y funcionalidades necesarias para la gestión correcta de requerimientos de mantenimiento y soporte a través de llamadas de servicio por parte de los usuarios.

5.3 Valor agregado del modelo de soporte

Dado que el cliente se encuentra en otro país, la solución bajo la que se generó el modelo de mantenimiento de nuestro sistema fue diferente del modelo tradicional de outsourcing, pues el servicio también fue puesto a licitación entre varias consultorías, las cuales están localizadas en Asia. Algunas de las ventajas inmediatas obtenidas mediante el modelo ofrecido son:

- Fuertes metodologías CMM (Capability Maturity Model) y procesos de mejora Six Sigma aseguran que el proyecto será terminado a tiempo y dentro del presupuesto brindando confiabilidad al cliente. El proceso de desarrollo probado a lo largo de varios proyectos exitosos provee al cliente la seguridad de que el proyecto estará bajo control en todas las fases, desde el análisis y diseño hasta la entrega.
- Debido a la cercanía entre México y Estados Unidos, los costos de mantenimiento del SIGP son reducidos. La ventaja reside en la capacidad para reducir el número de recursos humanos establecidos físicamente en la empresa del cliente. Para ello, se ha desarrollado un proceso maduro y utilizando las mejores prácticas de 7 años de experiencia con el cliente para lograr un nivel del 5% en cuanto a recursos ubicados en las instalaciones del cliente. De esta manera, se minimiza el costo en el espacio de oficina, computadoras, pago de honorarios y administración.
- Los recursos humanos que están fuera de las oficinas del cliente pueden ser contactados en cualquier momento en horas laborales de Estados Unidos por medio

de chat, teléfono o correo electrónico, tal y como si estuvieran ubicados dentro de la empresa del cliente.

- Los centros de desarrollo y soporte donde se llevó a cabo el SIGP, están ubicados a sólo algunas horas de las ciudades más importantes de Estados Unidos. En el caso de que se necesite enviar un recurso a las instalaciones del cliente, se puede hacer por periodos cortos de tiempo. Esto se traduce en traslados más baratos, duración de vuelo menor y viáticos no costosos.
- Existe una similitud cultural importante, pues México, por estar geográficamente cerca de Estados Unidos, comparte una cultura de negocios y afinidad que facilita la integración cotidiana del equipo.

5.4 Modelo del soporte

El mantenimiento y soporte de aplicaciones está enfocado a asegurar que los usuarios tengan acceso continuo a los servicios apropiados para poder ejecutar las actividades del negocio. De acuerdo a los estándares establecidos para el modelo de mantenimiento de software las actividades básicas relacionadas al mismo así como el flujo de ellas se muestran en la figura 5.2.

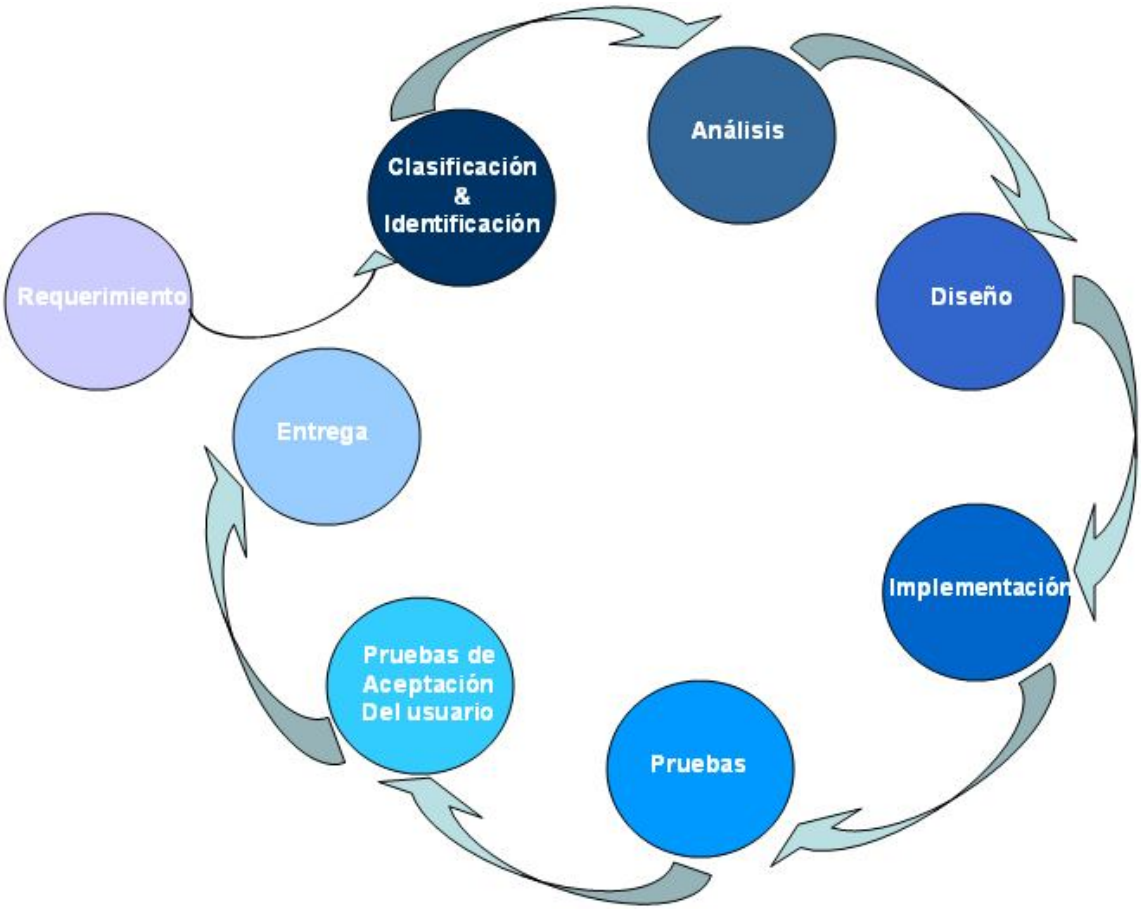


Figura 5.2 Modelo del soporte de aplicaciones

De acuerdo a la figura 5.2, los requerimientos de mantenimiento y soporte siguen la dirección del flujo representada por las flechas del diagrama, donde inicialmente se clasifica el tipo de requerimiento que se atenderá. Posteriormente, se procede a analizar la situación y el contexto de la petición para identificar las causas por las que el requerimiento ha sido levantado. Después de la fase de análisis se diseña la manera en que el problema o la petición serán atendidos, para así desarrollar e implementar la solución correspondiente. Una vez cumplidas las etapas mencionadas, lo siguiente es realizar las pruebas pertinentes de la implementación, ejecutadas tanto de manera interna dentro del equipo de soporte,

como por el cliente para obtener su aceptación respecto a la solución desarrollada. Finalmente, se genera el entregable para su liberación ante el cliente.

Como miembro del equipo de mantenimiento se está involucrado en:

- La atención en la petición de cambios
- La constante y oportuna comunicación y seguimiento de los requerimientos
- Resolver dificultades tales como fallas en módulos o fallas generales del sistema
- El desarrollo de mejoras en los sistemas para incrementar la productividad del negocio, ya sea orientadas a algunos módulos o el desarrollo completo de una nueva versión.

5.5 Plataforma de servicio y mantenimiento de aplicaciones

Actualmente, el punto de contacto entre los usuarios y los expertos es una aplicación en línea que llamaremos HelpPoint. En ella se registran los requerimientos para que sean dirigidos al correspondiente Grupo de trabajo o en su defecto al Helpdesk en caso de no saber con quién dirigirse.

Por otro lado, los expertos hacen uso de un Service Desk Virtual, ya que los usuarios del SIGP se encuentran en varios países y este tipo de estructura permite estar situada y ser accesible en cualquier parte del mundo, gracias a redes de telecomunicaciones avanzadas y de alto desempeño, reduciendo así costos operacionales y mejorando el uso de los recursos disponibles.

En esta interfaz se pueden visualizar los requerimientos que el usuario ha solicitado. Referente a los procesos de la Operación del servicio definidos por ITIL v3, se contemplan llamadas de servicio o solicitudes, incidentes, problemas, eventos y accesos. Asimismo, la interfaz está configurada para gestionar automáticamente las escalaciones correspondientes

y los acuerdos de nivel de servicio que se han definido bajo la segunda etapa de ITIL v3 llamada Diseño de servicio, la cual tiene como propósito definir los servicios y procesos que cumplan con las expectativas del negocio, proporcionar sistemas de medición y métricas para identificar y minimizar riesgos. Bajo este esquema, inicialmente se tiene una Llamada de Servicio (Service Call) en la que se tratará de resolver la solicitud del usuario en no más de 5 días, dependiendo de la prioridad del requerimiento. Si en ese tiempo no es posible proporcionar una solución final, entonces esa Llamada de Servicio se convertirá en un Incidente dado el impacto que tiene en la aplicación.

De acuerdo a ITIL v3, el objetivo del Manejo de incidentes es restablecer el servicio normal de operación (entendido como un servicio dentro de los límites el Acuerdo de Nivel de Servicio o SLA) de manera normal tan rápido como sea posible y minimizar los efectos en las operaciones del negocio, para así lograr los mejores niveles de calidad de servicio y disponibilidad.

Los incidentes pueden desencadenar a su vez varios procesos de ITIL v3, tales como Manejo de incidentes, Manejo de problemas, Manejo de cambios, etc.

El objetivo del Manejo de problemas es resolver la causa raíz de los Incidentes y minimizar el impacto en el negocio de los mismos, para así prevenir la recurrencia de incidentes relacionados con los errores de la infraestructura de las tecnologías de información.

Un problema es la causa de uno o más incidentes y un ‘error conocido’ es un problema que ha sido diagnosticado exitosamente y a través del cual se ha generado una solución temporal o permanente.

Se debe marcar la diferencia entre el Manejo de incidentes y Manejo de problemas. El propósito principal del Manejo de problemas es encontrar y solucionar la causa raíz del

problema para prevenir incidentes. Por otro lado, el Manejo de incidentes es reanudar el servicio

Todas estas cadenas de procesos que define ITIL v3 son registradas en el Service Desk Virtual, mediante una base de datos que almacena los procesos llamada CMDB (Configuration Management Database).

En resumen el Service Desk Virtual ayuda a:

- El control de incidentes a través del ciclo de vida del requerimiento
- La comunicación para que el usuario esté informado del progreso y solución del requerimiento

Para un soporte más directo, se tienen las siguientes características:

- Sólo hay un lugar en el que se pueden registrar los requerimientos (HelpPoint)
- La interfaz debe ser sencilla para el usuario
- Debe haber integridad de datos del requerimiento
- La comunicación es lineal

Esta solución es implementada para proveer un mejor servicio a los clientes. Se logra a través del esfuerzo conjunto de la administración del equipo de mantenimiento, administración del cliente y calidad.

5.6 Soluciones y áreas de servicio

En el proyecto se cuenta con un grupo de soluciones multiplataforma que se divide en 4 áreas:

5.6.1 Administración del servicio

Esta área es la encargada de asegurarse que el servicio será ininterrumpido y se hará de la mejor manera, mediante el análisis periódico de métricas que permitirán saber si los niveles de servicio acordados por el cliente se están cumpliendo. Esto se logra mediante reportes, juntas con el cliente, creación de métricas y monitoreo. Algunas de las métricas más utilizadas son:

- Entrega a tiempo
- Entrega correcta al primer intento
- Tiempo de respuesta
- Costo de calidad

Otra labor que se comprende es la apropiada capacitación, tanto técnica como funcional de los expertos, para que puedan brindar el servicio esperado y que, las etapas de transiciones internas del equipo, no afecten de ninguna forma al cliente.

5.6.2 Soporte a producción

Esta área se encarga de que, el portafolio de aplicaciones asignadas, tenga el mantenimiento adecuado para que su uso sea ininterrumpido y así asegurar la continuidad de la aplicación. Todo esto se logra mediante la corrección de errores, el monitoreo de la aplicación, el monitoreo de la base de datos y soporte On Call de 24X7 en caso de ser necesario.

5.6.3 Soporte del negocio

Las actividades realizadas en un área de este tipo son del tipo funcional, ya que el experto tiene que conocer profundamente las funcionalidades de la aplicación y cómo se relacionan con la lógica del negocio a la que está dedicada. De esta manera se logra que los usuarios

tengan un experto que podrá asistirlos en el uso de la aplicación, para que las operaciones del negocio se den adecuadamente. También ayudarán a los líderes funcionales a identificar problemas en el sistema y así crear propuestas para mejorar la aplicación.

5.6.4 Mantenimiento y mejoras

En esta área de servicio se contemplan tanto aspectos técnicos como funcionales. Los expertos se encargarán del mantenimiento correctivo y preventivo de la aplicación. Existe todo un proceso controlado para que los cambios se den de una manera ordenada y segura para así evitar un impacto irreversible que pueda producir una situación crítica.

Otras tareas contemplan mejoras menores y mayores, como nuevas versiones de la aplicación o funcionalidades que cubran nuevas necesidades en el negocio, administración de la bases de datos, seguridad y revisión de código.

En resumen las metas que fijamos en AMSG son:

1. Mantener los sistemas en completa funcionalidad sin ninguna interrupción.
2. Mantener los niveles de SLA
3. No detener ningún proceso de manufactura debido a alguna falla en el sistema
4. Manejo eficiente de los recursos para responder a diferentes requerimientos
5. Anticipación a las necesidades del negocio
6. Soportar mediante un modelo 24x7 las aplicaciones críticas para el negocio
7. Proveer soporte técnico 24x7 con respuestas efectivas y precisas
8. Mantener el código fuente, analizar y reparar errores y fallas en los procesos
9. Adaptar los sistemas a cambios en las reglas del negocio mediante mejoras con un previo análisis y diseño ejecutado.

Capítulo 6

Resultados, impacto y conclusiones

6.1 Resultados

Para hacer una comparación de resultados en el desempeño de la aplicación, se calendarizó un monitoreo de utilización de recursos previo a la entrega del producto final. De esta manera, podemos darnos cuenta de cómo responden ciertas variables de la infraestructura antes de la implementación de la nueva versión del SIGP. En dicha actividad se obtuvieron mediciones de carga de la aplicación, transferencia de datos y uso del CPU de la aplicación. Cabe mencionar que la aplicación sufría inestabilidad que provocaba fallas recurrentes y se requería del reinicio de los servidores, así como también de la administración de sesiones de base de datos inactivas.

Una vez finalizado el proyecto de desarrollo, se hizo un nuevo monitoreo bajo las mismas condiciones, pero con resultados diferentes y satisfactorios. En la figura 6.1 se muestra el comparativo de los servidores anteriores y los actuales tomando muestras diarias a lo largo de una semana completa.

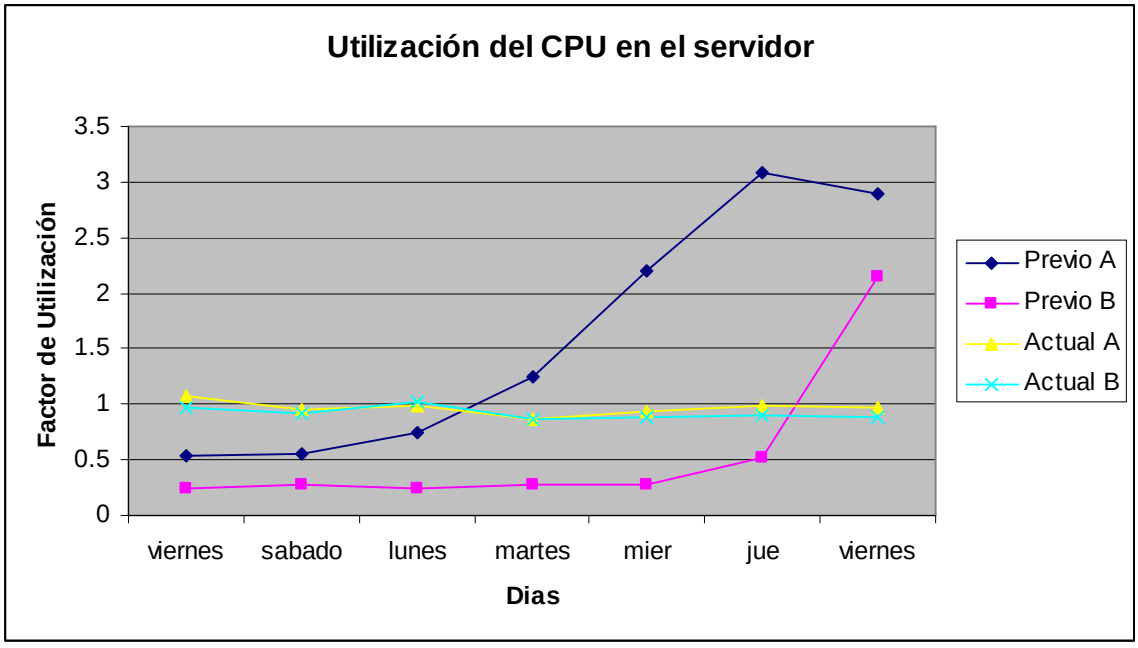


Figura 6.1 Comparativo de la utilización del CPU

Se puede apreciar en la figura 6.1 que las tendencias de las líneas Previo A y Previo B, referentes a los servidores anteriores seguían un patrón creciente en el tiempo, lo que provocaba las interrupciones e inestabilidades del sistema. Por otro lado, para las líneas Actual A y Actual B se nota una estabilidad constante con mínima variación, que demuestra un sistema estable. Esto se logró tras el desarrollo y mejoras del sistema relacionadas a correcciones en el manejo de excepciones y conexiones con la base de datos. Asimismo la actualización de componentes, como versión de base de datos y del servidor de aplicaciones, ayudó a esta mejora.

Referente a los problemas reportados al equipo de soporte y mantenimiento, también se registraron mejoras de gran relevancia después de la implementación del nuevo sistema.

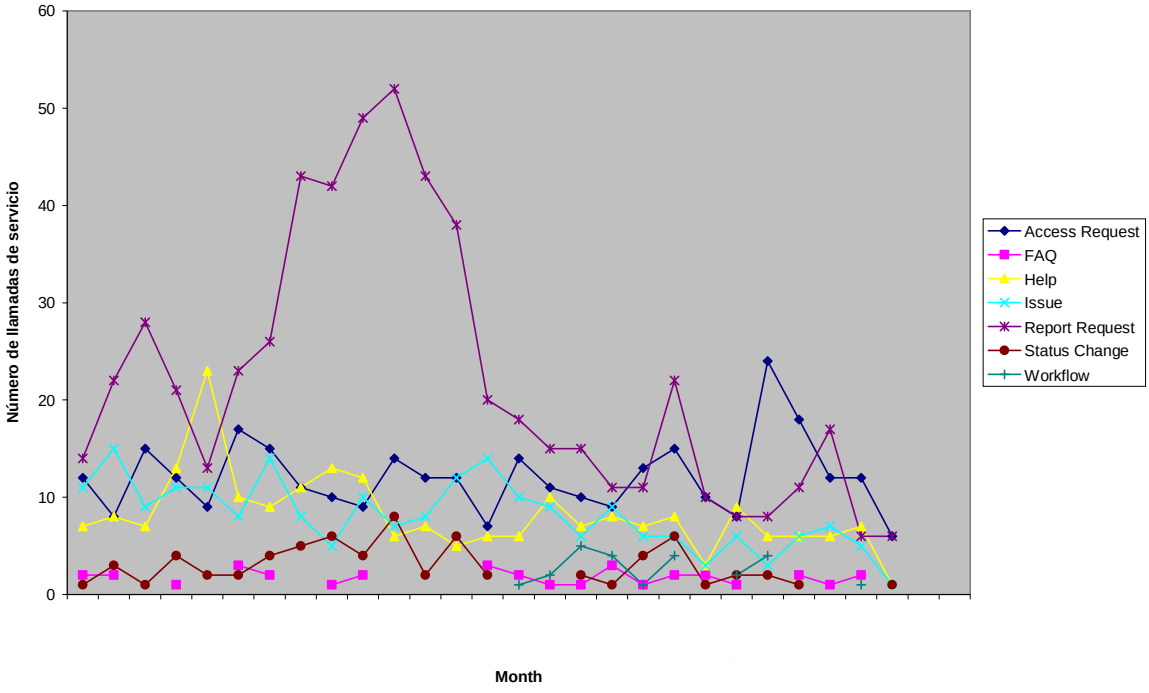


Figura 6.2 Tendencias de los diferentes tipos de llamadas de servicio

En la figura 6.2 se muestra el comparativo entre 2009 y 2010 de las llamadas de servicio. Se puede apreciar una gran disminución de las llamadas de servicio, a partir de que el sistema fue entregado e instalado. Para tener una comparación uno a uno por cada categoría de llamada, se generó la figura 6.3, en la que es evidente la mejora en cuanto a disminución del número de llamadas de servicio recibidas por parte de los usuarios. Por ejemplo, relacionado a las peticiones de reportes por parte de los usuarios, hubo una disminución del 71% gracias a los reportes automatizados que se implementaron en la nueva versión del SIGP. De manera similar, en 2010 los problemas reportados en la aplicación fueron inferiores en un 53%, gracias a las correcciones y mejoras implementadas a nuestro sistema.

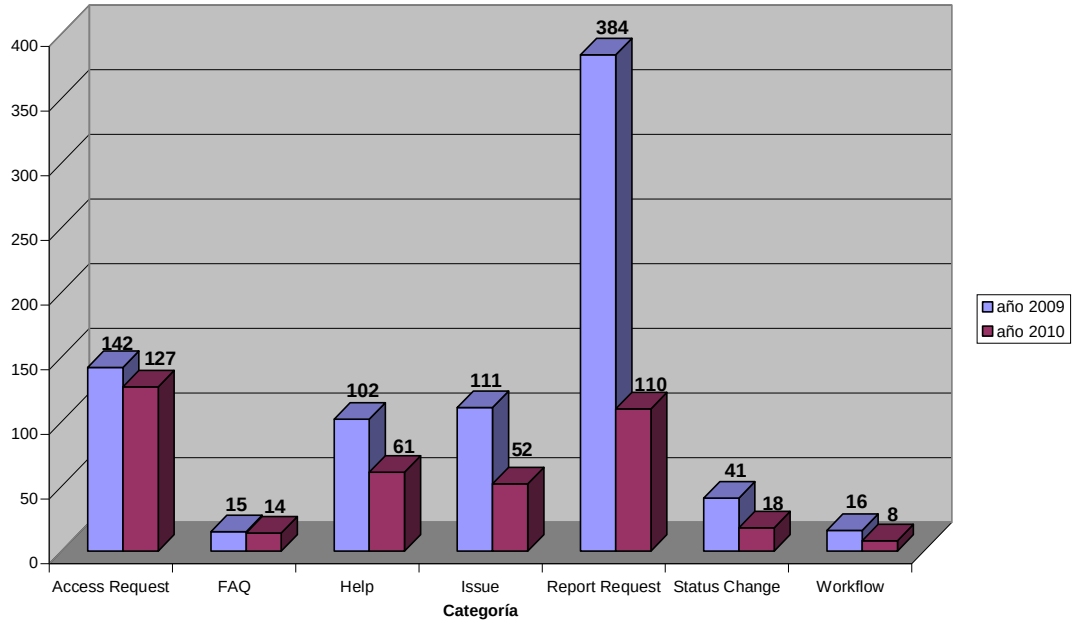


Figura 6.3 Comparativo de llamadas de servicio (2009 vs 2010)

En la figura 6.4 se presenta el consolidado de los porcentajes de mejora de cada categoría, lo que demuestra que la nueva versión del SIGP ha atendido satisfactoriamente las fuentes de falla y ha mejorado la productividad para los usuarios.

Categoría	% de disminución
Access request	11%
FAQ	7%
Help	40%
Issue	53%
Report request	71%
Status change	56%
Workflow	50%

Figura 6.4 Porcentajes de mejora de llamadas de servicio

En cuanto a las fallas generales del sistema, de igual manera hubo diferencias positivas muy importantes, pues la disponibilidad de nuestra aplicación fue mayor comparada a la del 2008 y 2009, lo cual es evidente por la disminución del número de estos eventos ocurridos a partir de la instalación del sistema. La figura 6.5 muestra la cantidad de fallas por cada trimestre del 2009 al 2010.

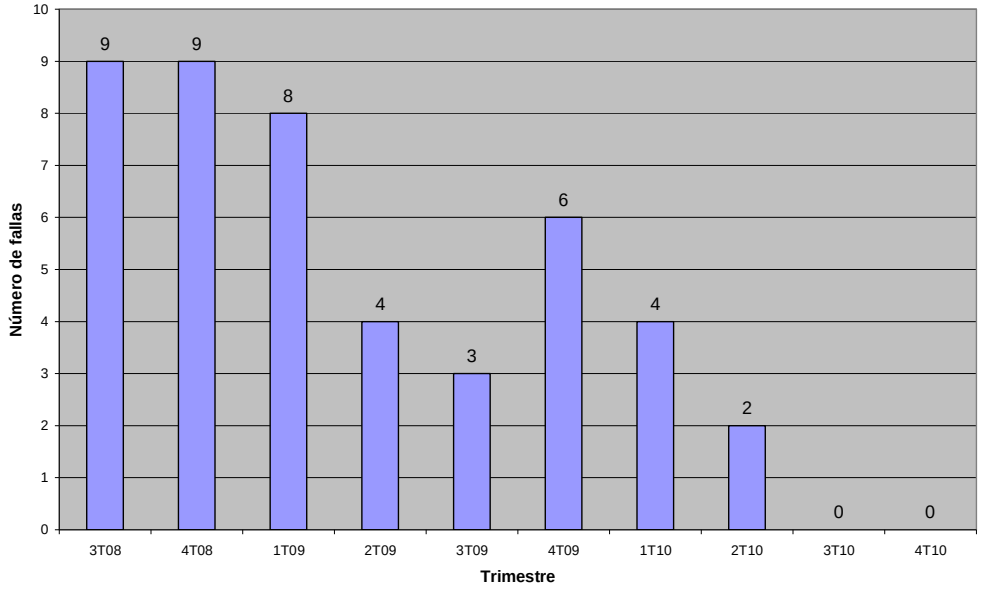


Figura 6.5 Fallas generales trimestrales del Sistema informático para la gestión de proyectos

En la figura 6.6 se puede apreciar la tendencia descendente de los eventos de falla, llegando incluso a tener disponibilidad ininterrumpida del sistema durante la segunda mitad del 2010, pues no se presentó ninguna falla durante ese periodo.

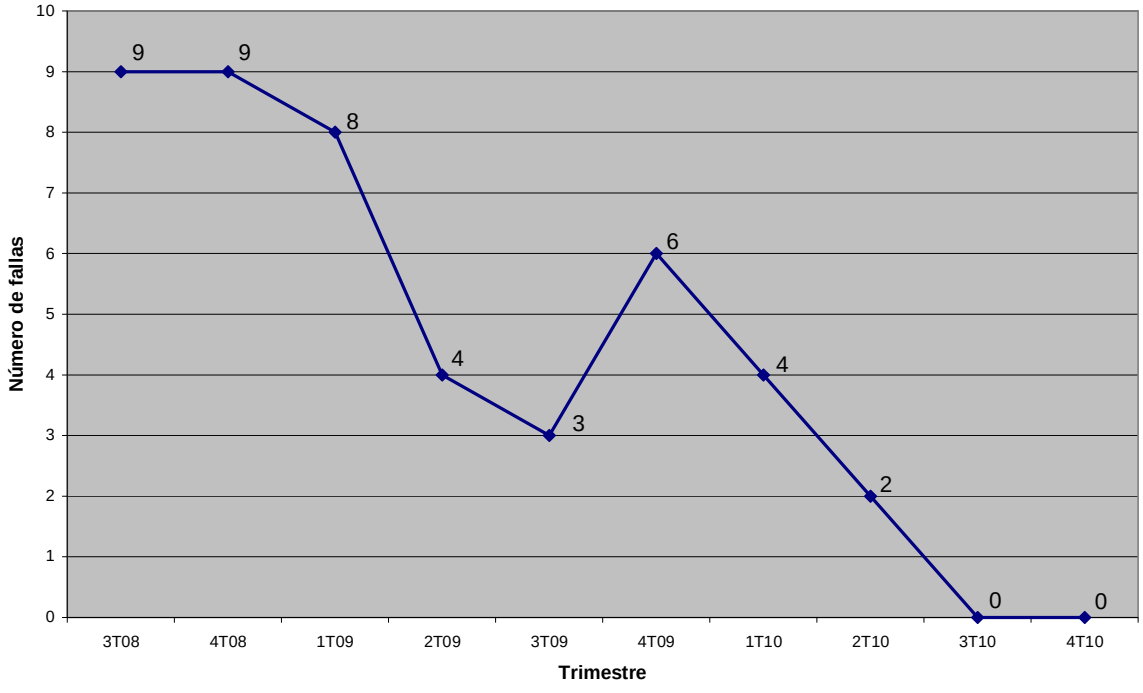


Figura 6.6 Tendencia trimestral de fallas generales del sistema

Finalmente, encontramos en la figura 6.7 la comparación anual de las fallas generales del sistema en el periodo comprendido entre 2008 y 2010. Se puede apreciar que, de manera drástica, los eventos disminuyeron en el último año, pues sólo se presentaron 6 incidencias contra las 21 ocurridas en el 2009, lo que resulta en una disminución del 71%. Haciendo un análisis de las fallas en 2010, se encontró que las causas de las mismas fueron ajenas a la propia aplicación, pues se relacionaron a problemas generales de red o infraestructura de la misma compañía y no a fallas particulares del SIGP. Gracias a ello, las operaciones del negocio no se vieron afectadas y la respuesta de los usuarios al nuevo sistema fue totalmente positiva.

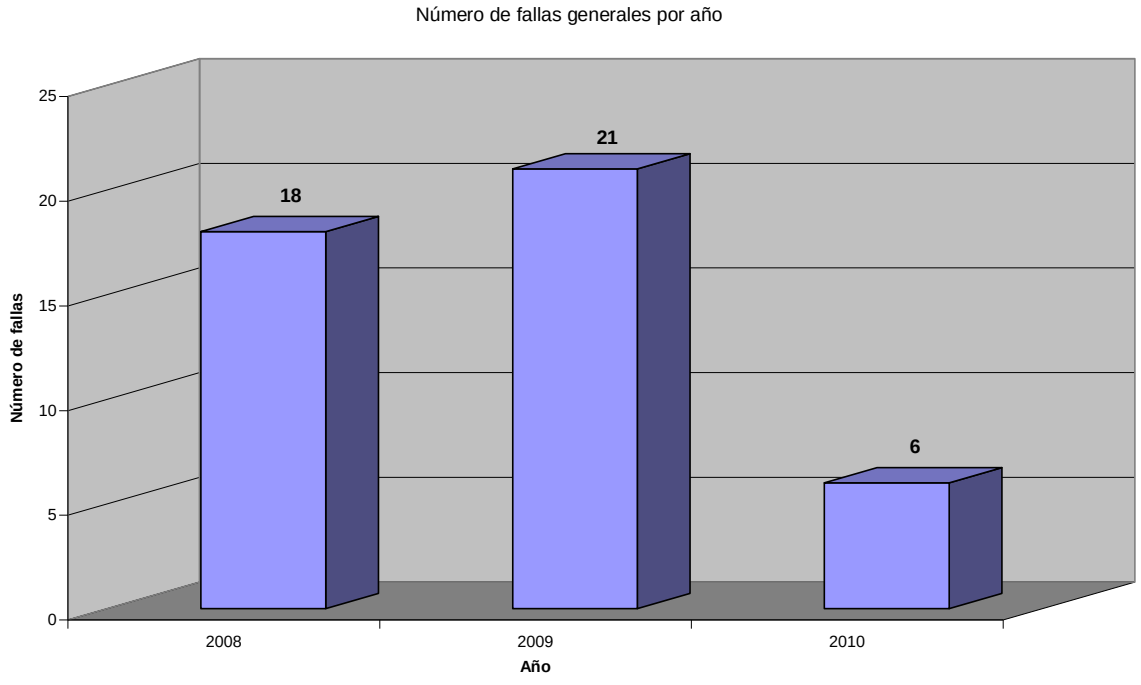


Figura 6.7 Fallas generales por año

6.2 Impacto

Referente al impacto que el desarrollo del SIGP tuvo para el cliente, se realizó un análisis de los proyectos terminados administrados por el sistema mencionado, con el fin de obtener un indicador de la productividad de los usuarios, después de la implementación de la nueva versión. La figura 6.8 muestra el comportamiento de cierres de proyectos exitosos en los últimos tres años. Con los datos recopilados se aprecia una mejora considerable, pues en 2010 se cerraron 100 proyectos más que en 2008 y 107 proyectos más que en 2009, lo que representa un incremento del 35% y 38% respectivamente. De esta manera comprobamos que la nueva versión del SIGP ha beneficiado de manera importante a los usuarios y cumplido con el objetivo de incrementar la productividad en el área donde se desempeña.

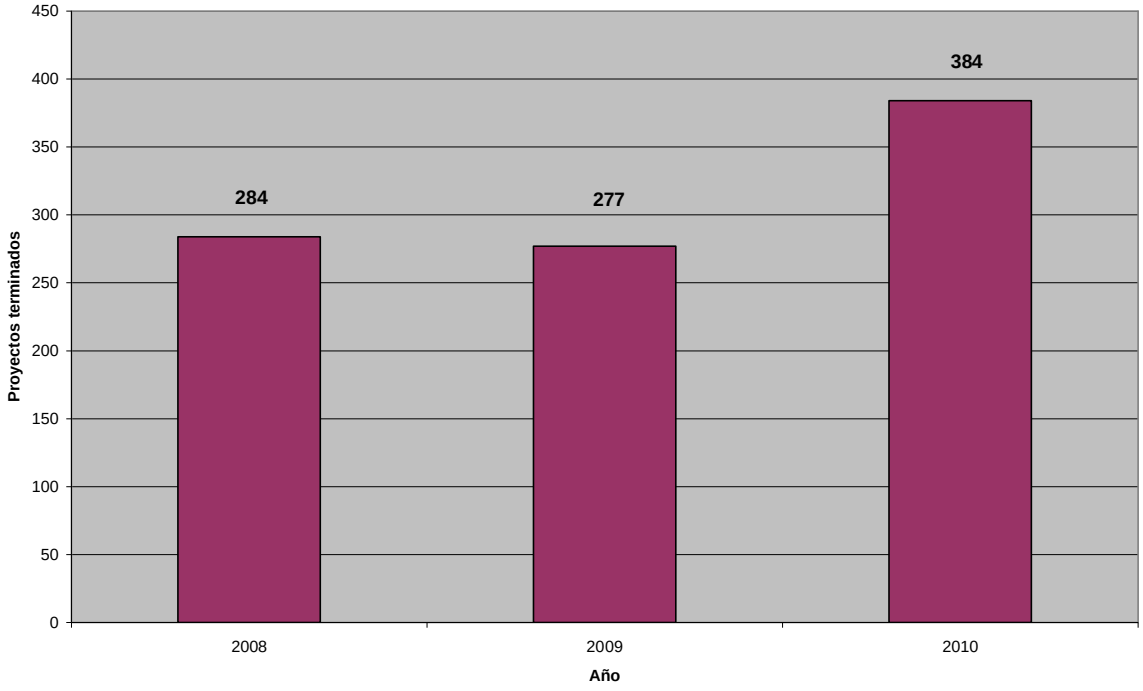


Figura 6.8 Proyectos terminados por año en el Sistema informático para la gestión de proyectos

Por otra parte, para obtener una retroalimentación directa del cliente, respecto a los cambios efectuados en la aplicación, se generó una encuesta de 4 preguntas a 20 usuarios clave del SIGP, con el fin de medir el grado de satisfacción un año después de la implementación de la nueva versión del sistema en cuestión. Las preguntas de la encuesta aplicada fueron las siguientes:

1. ¿Cómo considera los cambios implementados en el Sistema informático para la gestión de proyectos?
2. ¿Las modificaciones efectuadas al Sistema informático para la gestión de proyectos satisfacen las necesidades de su negocio?
3. ¿Considera que los cambios efectuados al Sistema informático para la gestión de proyectos corrigieron los errores de la versión anterior?

4. ¿Las mejoras al Sistema informático para la gestión de proyectos han incrementado las ventas de la compañía?

En la figura 6.9 se muestra la distribución de respuestas relacionadas a la primer pregunta de la encuesta aplicada a los usuarios, donde se obtuvo que el 80% de los encuestados considera que los cambios implementados fueron excelentes, mientras que el 20% los considera como buenos. No se obtuvieron opiniones negativas al respecto.

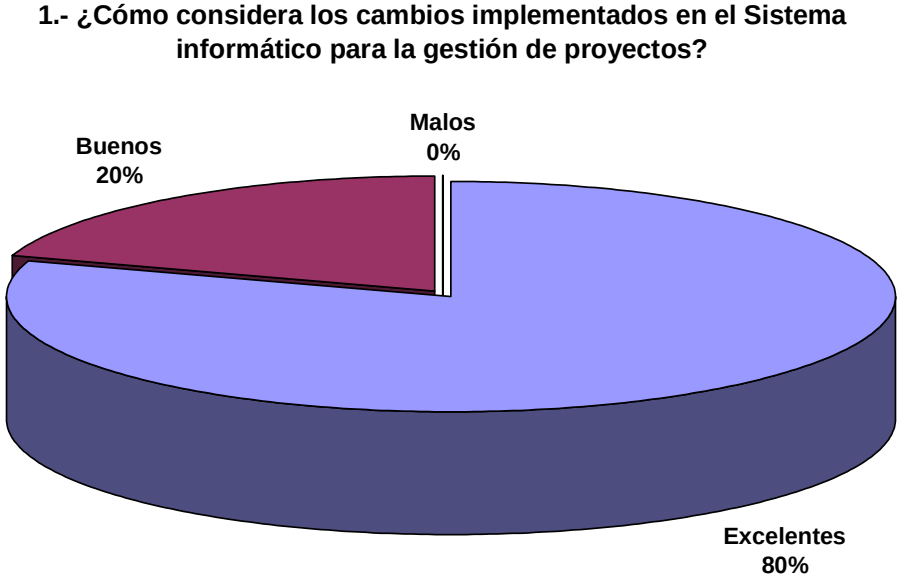


Figura 6.9 Resultados de la pregunta 1 de la encuesta

Referente a la segunda pregunta de la encuesta, en la figura 6.10 se muestra que el 95% de los usuarios clave encuestados opinan que el Sistema informático para la gestión de proyectos satisface las necesidades del negocio, mientras que sólo un 5% opina que aún no cubre la totalidad de lo que se necesita para ser un sistema integral y apegado a los procesos de negocio. Profundizando con esta minoría, se identificó que su opinión se debe a percepciones diferentes en cuanto al alcance del Sistema informático para la gestión de

proyectos, pues esperan más de lo que está estipulado en la definición de la aplicación y no se descarta que en una futura versión se atienda el incremento en el alcance del sistema para robustecerlo.

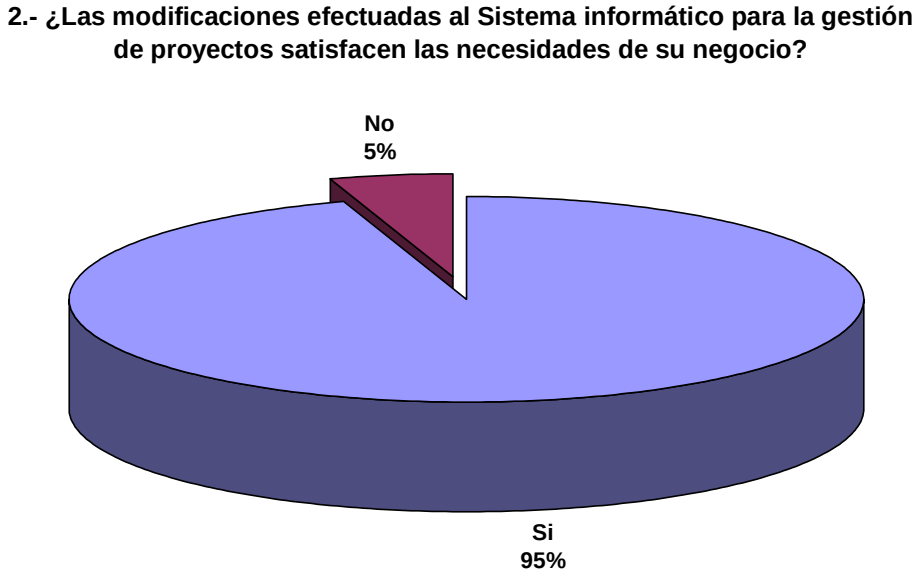


Figura 6.10 Resultados de la pregunta 2 de la encuesta

En cuanto al tema de los errores de la versión anterior del Sistema informático para la gestión de proyectos, atendido en la tercer pregunta de la encuesta, el cien por ciento de los encuestados concuerda en que se corrigieron los errores que existían, tanto aquellos relacionados a codificación como a los referentes a discrepancias entre las actuales reglas del negocio. De esta manera, se obtiene un indicador de que el objetivo de eliminar los errores y adaptar la aplicación a los nuevos procesos del negocio ha sido atendido de manera satisfactoria, como se muestra en la figura 6.11.

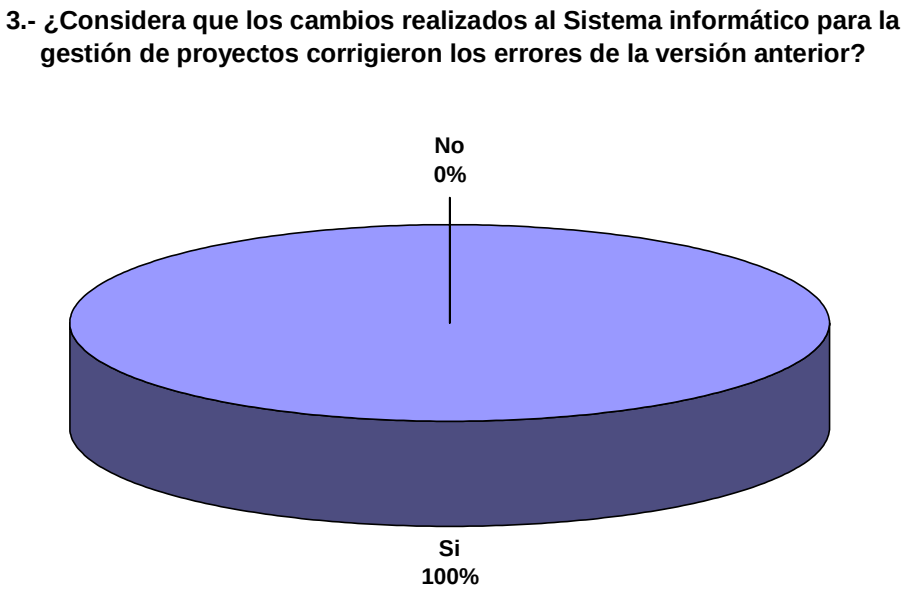


Figura 6.11 Resultados de la pregunta 3 de la encuesta

Relacionado a la tendencia de las ventas de la compañía, que se inquiere en la última pregunta de la encuesta, se obtuvieron respuestas favorables, pues el 90% de los usuarios clave respondieron que el Sistema informático para la gestión de proyectos ha incrementado las ventas. Los resultados de esta última pregunta de la encuesta aplicada se aprecian en la figura 6.12 y, puede corroborarse con las tendencias anteriormente mostradas en la figura 6.8, pues un aumento en la productividad genera más proyectos terminados y a su vez más ganancias para el cliente.

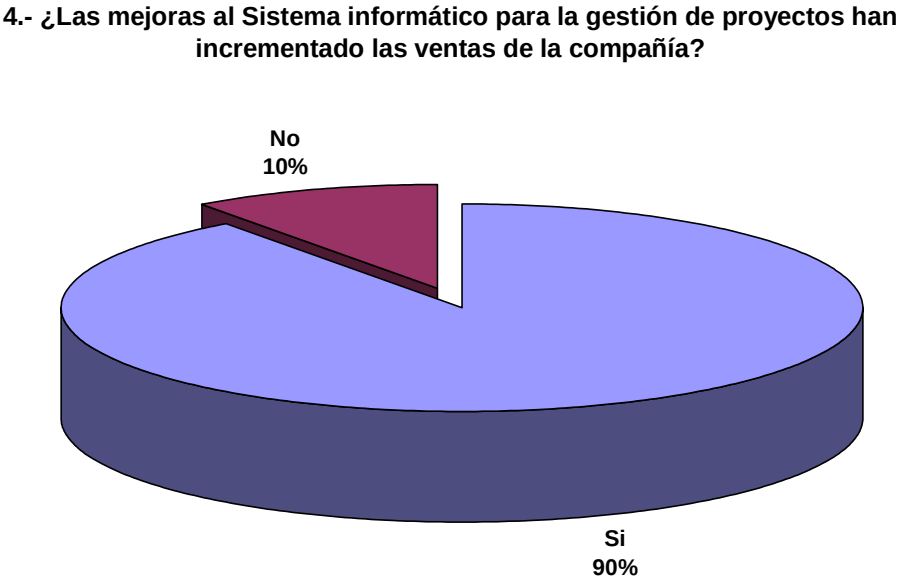


Figura 6.12 Resultados de la pregunta 4 de la encuesta

En la tabla 6.1 se resumen los requerimientos del cliente y se presentan los resultados obtenidos.

Tabla 6. 1 Relación de requerimientos, resultados e impacto

Requerimientos	Resultados e impacto
Mejorar la productividad de los usuarios mediante la integración de los nuevos procesos al sistema.	*Hubo un incremento del 38% de la productividad en términos de proyectos cerrados.*
Implementar puntos de control críticos para mantener consistencia en la información	*Se redujeron los problemas de la aplicación en un 53%, logrando confiabilidad e integridad en la información*
Incorporar nuevos requerimientos derivados de la reingeniería de procesos del negocio	*Los requerimientos solicitados por el negocio fueron implementados. De acuerdo a encuestas el 95% de los usuarios considera que el sistema satisface las necesidades del negocio*
Incrementar las ventas y reducir gastos y tiempos de entrega	*Debido al aumento de productividad de los usuarios, se generaron más proyectos logrando mayores ventas en menor tiempo. Se redujo el trabajo duplicado en los proyectos logrando una disminución en los gastos de operación.*
Reducir fallas generales en la aplicación que interrumpan el servicio de la misma	*Se redujeron las fallas generales en un 71% logrando alta disponibilidad del sistema.*
Mejorar el desempeño de la aplicación	*Se mejoró el manejo de memoria y conexiones en el código de la aplicación. De igual manera se mejoró la infraestructura en términos de recursos como servidores físicos y actualizaciones en base de datos y servidor de aplicaciones*
Generar el análisis y diseño de la nueva versión	*Se generaron los documentos relacionados al análisis y diseño de la nueva versión, utilizando UML como herramienta.*
Desarrollo de la nueva versión	*La nueva versión fue desarrollada bajo el modelo en espiral de ingeniería de*

	software. Este acercamiento al proyecto generó satisfacción en el cliente dado que se cumplieron sus expectativas.
Soporte en la entrega al ambiente productivo, post-producción y transferencia de conocimiento	*Apegados a ITIL se efectuaron las actividades necesarias para gestionar e implementar el plan de mantenimiento del sistema. Dicho proceso fue transparente para los usuarios.*
Documentación del sistema	*Se generaron los documentos especificados en la lista de entregables del cliente y fueron depositados en un repositorio para su consulta en caso de que el equipo de mantenimiento lo necesite.*

6.3 Conclusiones

Para proyectos grandes como el Sistema informático para la gestión de proyectos, la metodología en espiral para el desarrollo de software resulta ser factible, pues se trata de un enfoque más realista en la ingeniería del software tradicional para sistemas grandes, ya que gracias a la fase de análisis de riesgos que incorpora, permite tanto a los desarrolladores, como al cliente entender y reaccionar tempranamente posibles problemas que se detectan en cada vuelta a la espiral. También, la generación de prototipos es utilizada como mecanismo de reducción del riesgo y mantiene el enfoque del ciclo de vida clásico, pero incorporándolo dentro de un proceso iterativo que refleja de forma más realista el mundo real. Otra ventaja es que promueve una mayor interacción y comunicación con el cliente, dado que su enfoque no es únicamente a los documentos sino también, a la respuesta y comentarios del cliente durante el proceso de desarrollo. Por tanto, metodología en espiral para el desarrollo del Sistema informático para la gestión de proyectos resultó crucial y de gran relevancia por las ventajas ya descritas.

Por otro lado, la utilización de UML en el desarrollo del Sistema informático para la gestión de proyectos, comprobó que es un leguaje gráfico de gran ayuda para analizar y especificar los requerimientos del sistema, facilitando el entendimiento de los mismos y agilizando las etapas de desarrollo de software. UML fue utilizado tanto para representar los elementos del sistema como para expresar y documentar las relaciones existentes entre ellos. De igual manera, la generación de los diagramas facilitó la transferencia de conocimiento al equipo de mantenimiento y soporte del Sistema informático para la gestión de proyectos.

En lo que se refiere al mantenimiento del Sistema informático para la gestión de proyectos, se concluye que la fase de transición entre la liberación de la aplicación y la etapa de mantenimiento debe ser gestionada de manera adecuada para asegurar el correcto funcionamiento y la alta disponibilidad del sistema. Asimismo, el mantenimiento activo del Sistema informático para la gestión de proyectos debe ser proporcionado bajo la alineación de las necesidades del negocio. La utilización de las librerías de administración de servicios de ITIL en estas etapas, demostró ser un acercamiento confiable y adecuado, pues la transición a la etapa de mantenimiento fue exitosa y la gestión de servicios, accesos, incidentes y problemas posteriores a la liberación del Sistema informático para la gestión de proyectos, se ha efectuado correctamente con respuestas positivas por parte de los usuarios. Los registros históricos de llamadas de servicio son constantemente monitoreados para preservar el control del servicio e identificar rápidamente áreas de oportunidad, manteniendo el enfoque de mejora continua.

Finalmente, gracias a los conocimientos de ingeniería en computación obtenidos en la Facultad de Ingeniería, fue posible contar con una formación integral y participar

activamente en el desarrollo del Sistema informático para la gestión de proyectos, desde su concepción, pasando por las etapas de desarrollo descritas en este trabajo, hasta el despliegue y mantenimiento del sistema, logrando así que los objetivos especificados por el cliente fueran cumplidos. De esta manera se concluye de manera exitosa el proyecto del Sistema informático para la gestión de proyectos, obteniendo aprendizaje y reafirmación del conocimiento ya adquirido y experiencia profesional útil para futuros proyectos de ingeniería.

Referencias bibliográficas

Alison Cartlidge. *An introductory overview of ITIL V3*. The UK Chapter of the itSMF. Reino Unido. 2007.

Hern Dennis M. *CMMI Distilled: A practical introduction to integrated process improvement*. Addison Wesley. Tercera edición. Estados Unidos de América. 2008.

López Quijado, José. *Domine JavaScript*. Alfaomega. Segunda edición. Madrid, España. 2007.

Office of Government Commerce. *ITIL Lifecycle Publication Suite Books*. The Stationery Office. Tercera edición. Reino Unido. 2007.

Parsons, David. *Desarrollo de aplicaciones web dinámicas con XML y Java.*Anaya Multimedia. Londres, Inglaterra. 2009.

Pender, Thomas A. *UML weekend crash course*. Wiley Publishing Inc. Nueva York. 2002.

Pressman, Roger S. *Software Engineering: A practitioner's approach*. McGraw-Hill Inc. Quinta edición. Nueva York. 2001.

Sierra, Kathy. *Head First Servlets and JSP*. O'Reilly Media. Segunda edición. Estados Unidos de América. 2008.

Skonnard, Aaron. *Essential XML Quick Reference*. Addison-Wesley Professional. Estados Unidos de América. 2001.

Sommerville, Ian. *Ingeniería del software*. Pearson educación, S.A. Séptima edición. Reino Unido. 2005.

Wampler, Bruce E. *The essence of object-oriented programming with Java and UML*. Addison Wesley. Estados Unidos de América. 2001.

Watson, John. *Oracle database 10g*. McGraw Hill. Emeryville, California. 2005.