UNIVERSIDAD ANDINA SIMÓN BOLÍVAR

Sede Ecuador

Área de Gestión

Programa de Maestría en Dirección de Empresas

PROPUESTA DE INSTRUMENTOS PARA MEJORAR EL ÁREA DE CONTROL DE CALIDAD DE SISTEMAS INFORMÁTICOS DE UNA EMPRESA DE DESARROLLO DE SISTEMAS PARA EL SECTOR PÚBLICO

Sonia Alexandra Moposita Vásquez

2013

CLAUSULA DE CESION DE DERECHO DE PUBLICACION DE TESIS/MONOGRAFIA

Yo, Sonia Alexandra Moposita Vásquez, autor/a de la tesis intitulada **Propuesta de Instrumentos para mejorar el Área de Control de Calidad de Sistemas Informáticos de una Empresa de Desarrollo de Sistemas para el Sector Público** mediante el presente documento dejo constancia de que la obra es de mi exclusiva autoría y producción, que la he elaborado para cumplir con uno de los requisitos previos para la obtención del título de Magister en Dirección de Empresas en la Universidad Andina Simón Bolívar, Sede Ecuador.

1. Cedo a la Universidad Andina Simón Bolívar, Sede Ecuador, los derechos exclusivos de reproducción, comunicación pública, distribución y divulgación, durante 36 meses a partir de mi graduación, pudiendo por lo tanto la Universidad, utilizar y usar esta obra por cualquier medio conocido o por conocer, siempre y cuando no se lo haga para obtener beneficio económico. Esta autorización incluye la reproducción total o parcial en los formatos virtual, electrónico, digital, óptico, como usos en red local y en internet.

2. Declaro que en caso de presentarse cualquier reclamación de parte de terceros respecto de los derechos de autor/a de la obra antes referida, yo asumiré toda responsabilidad frente a terceros y a la Universidad.

3. En esta fecha entrego a la Secretaría General, el ejemplar respectivo y sus anexos en formato impreso y digital o electrónico.

Fecha. 07 de enero del 2013

Firma: é é é é é é é é .....

UNIVERSIDAD ANDINA SIMÓN BOLÍVAR

Sede Ecuador

Área de Gestión

Programa de Maestría en Dirección de Empresas

PROPUESTA DE INSTRUMENTOS PARA MEJORAR EL ÁREA DE CONTROL DE CALIDAD DE SISTEMAS INFORMÁTICOS DE UNA EMPRESA DE DESARROLLO DE SISTEMAS PARA EL SECTOR PÚBLICO

Sonia Alexandra Moposita Vásquez

Tutora: Ing. Cecilia Jaramillo

Quito, 2013

RESUMEN

En el sector de Desarrollo de Software es muy común encontrar sistemas tanto a nivel de organizaciones privadas como públicas; que si bien están operando, no satisfacen todas las necesidades de sus usuarios, lo que ocasiona que no sean utilizados en su totalidad y lo que es peor, a perder la confianza del cliente en cuanto a quien lo desarrolló, para evitar esto, se debe fomentar el uso de modelos, metodologías ó buenas prácticas que permitan establecer el camino a seguir y el control que se debe hacer entre el fin e inicio de cada etapa del ciclo de desarrollo de software.

El presente trabajo, se lo realizo en la empresa ñSYSGENSAò en la cual se vio la necesidad de mejorar sus procesos, y empoderar a su recurso humano a fin de desarrollar aplicaciones que cumplan con niveles de calidad, tiempos y recursos de acuerdo a una planificación en función de las expectativas de sus clientes. Se proponen instrumentos y herramientas que apoyen al proceso de control de calidad en el ciclo de desarrollo de software de la Empresa.

En la Tesis se desarrolla, en el capítulo 1, el marco teórico relacionado con calidad, en el capítulo 2 , se realiza un análisis breve sobre el sector de la industria del desarrollo de software en el Ecuador, en el capítulo 3, se presenta el análisis situacional de la empresa SYSGENSA, en el capítulo 4, se plantea la propuesta de mejora al proceso de desarrollo de software de la empresa, y un grupo de herramientas informáticas libres y propietarias que pueden apoyar a nivel de gestión de requerimientos, control de calidad, gestión de configuración y versiones, y gestión del proyecto.

DEDICATORIA

A ti mi Dios, que me has permitido seguir creciendo espiritual y profesionalmente.

A ti mamita María, por estar junto a mí en cada paso que doy.

A mis Padres, por su apoyo y confianza incondicional.

A mi princesa Da, por su comprensión y sacrificio.

INDICE GENERAL

CAPITULO III

CAPITULO IV

INDICE DE FIGURAS

INDICE DE TABLAS

INDICE DE ANEXOS

INTRODUCCION

El creciente nivel competitivo y la necesidad constante de las organizaciones de maximizar su productividad y competitividad para mantenerse en el mercado, ha obligado a que se busquen mecanismos para satisfacer las exigencias de sus clientes en cuanto a calidad de productos, procesos y servicios; lo cual conlleva una inversión en la mejora de sus procesos, la innovación de sus técnicas y diseño de nuevos productos.

La manera de lograr que una organización sea realmente productiva en términos de eficiencia y eficacia y que sirva de referencia para el resto de las del sector, es mediante la implementación de sistemas de calidad completamente documentados, mismos que pueden ser configurados de acuerdo a las necesidades y a las diferentes normas aplicables al sector al que se desenvuelve la empresa.

Para el sector de Desarrollo de Software es muy común encontrar sistemas que si bien están operando, no satisfacen todas las necesidades de sus usuarios, lo que ocasiona que no sean utilizados en su totalidad y lo que es peor a perder la confianza del cliente en cuanto a quien lo desarrollo, para evitar esto, se debe fomentar constantemente el trabajo en equipo usuarios-empresa de desarrollo durante todo el ciclo de desarrollo de software, utilizando modelos, metodologías ó buenas prácticas que permitan establecer el camino a seguir y el control que se debe hacer entre el fin e inicio de cada etapa de dicho ciclo.

Para el caso, se tomó a la empresa ñSYSGENSAò que pertenece al sector de Desarrollo de Software, misma que vio la necesidad de mejorar sus procesos, y empoderar a su recurso humano a fin de desarrollar aplicaciones

que cumplan con niveles de calidad, tiempos y recursos de acuerdo a una planificación en función de las expectativas de sus clientes.

En el presente trabajo se plantean instrumentos y herramientas que apoyen al proceso de control de calidad en el ciclo de desarrollo de software de la Empresa SYSGENSA, de tal forma que aporten efectiva y eficientemente en la construcción de sistemas informáticos que cumplan las necesidades de sus clientes tanto del sector privado como del sector público. Para esto se plantea fortalecer el área de control de calidad de la empresa en algunos aspectos; mejorando sus procesos y planteando alternativas para el cumplimiento de cada fase del ciclo de vida de software, mediante la implantación de las buenas prácticas a través de una metodología como Rational Unified Process más conocida como RUP, misma que se apoyará en la definición de plantillas, formularios, listas de chequeo y herramientas para la gestión y control de cada fase.

En el capítulo 1, se presenta el marco teórico relacionado con calidad.

En el capítulo 2, se presenta un breve análisis del sector de la industria del desarrollo de software en el Ecuador.

En el capítulo 3, se realiza el análisis situacional de la empresa SYSGENSA

En el capítulo 4, se plantea la propuesta de mejora al proceso de desarrollo de software de la empresa, y una gama de herramientas informáticas libres y propietarias que pueden apoyar a nivel de gestión de requerimientos, control de calidad, gestión de configuración y versiones, y gestión del proyecto. Adicionalmente, se incluyen algunas plantillas de documentos que deben generarse en cada proyecto de la empresa.

Finalmente se incluyen las conclusiones y recomendaciones producto del desarrollo de esta tesis.

CAPITULO I
MARCO TEÓRICO

1.1. Introducción

Durante las últimas décadas se ha venido enfatizando la importancia del concepto de calidad, a tal punto de que ahora no solo se puede hablar de hacer las cosas bien; sino de mantener el nivel de calidad del producto o servicio. Antes, se consideraba a la calidad como un factor que implicaba aumento de costos en la generación del producto o servicio, ahora la búsqueda de calidad para una empresa tiene múltiples beneficios cuando se logra su implementación con compromiso y liderazgo.

La innovación, costos y calidad son elementos de la competitividad empresarial, de estos; la calidad corresponde a la capacidad de identificar y satisfacer las necesidades de los clientes entregando los productos solicitados, es ahí donde se reconoce la necesidad de herramientas que garanticen la calidad de los productos que se desarrollen.

La calidad no solo constituye entregar un producto o servicio; se ha convertido en un estilo y modo de vida. La calidad es el medio para generar ventajas ante la competencia.

El término ñCalidadò tiene una implicación muy amplia en el interior de la empresa, por esto es necesario presentar el alcance de cada concepto, a fin de establecer el ámbito de acción en cada nivel del proceso de desarrollo de software, tanto a nivel de equipos de trabajo internos de áreas específicas o departamentos, como a nivel medio y a nivel gerencial.

En este capítulo se expondrán conceptos generales relacionados a calidad y su evolución de manera general y aplicada al desarrollo de sistemas informáticos.

1.2. Calidad

Calidad es el ñGrado en el que el conjunto de características inherentes cumple con los requisitosó[1].

Se habla de calidad cuando se trata de construir las cosas de una manera correcta, de tal forma que los deseos del cliente sean cumplidos óptimamente. Además, de que los responsables de su construcción se sientan satisfechos y motivados con el reconocimiento del producto final tanto a nivel clientes, como interno de la empresa para la cual trabajan.

La calidad está estrechamente relacionada con el ser humano y su desarrollo, es así que su evolución a lo largo del tiempo ha tomado diferentes definiciones, alcances y enfoques de acuerdo a la época y medio de aplicación.

1.2.1. Evolución del concepto de Calidad

Calidad es un término dinámico por estar condicionado a los cambios acelerados que la sociedad ha sufrido en los últimos tiempos; su evolución histórica inicia a principios del siglo con una asociación simple de inspecciones a los atributos del producto. Con el transcurso del tiempo la concepción cambia y se orienta a un control estadístico, en el que se verifica el producto pero tomando en cuenta muestras representativas (control de calidad). Después, se llega a considerar a todos los miembros de la empresa como responsables del logro de la calidad (aseguramiento de la calidad), posteriormente, la calidad para a ser vista como una oportunidad competitiva, la orientación se concibe como que la calidad se administra (Gestión de calidad) Y finalmente, se llega al estado actual en el que la calidad es considerada como parte de la estrategia

[1] ISO 9000 Sistemas de Gestión de la Calidad ï Fundamentos y Vocabulario

empresarial; los productos o servicios según los perciba el cliente, pasan a ser la fuerza propulsora para el funcionamiento de la empresa (Gestión Total de la Calidad). Esto, demuestra la existencia de cinco etapas[2] en la evolución histórica del concepto de calidad, que se presentan en la Figura 1 y que son: inspección, control de calidad, aseguramiento de la calidad y gestión estratégica de la calidad.

En la Tabla 1, se resumen los cambios sustanciales a nivel de las organizaciones en cuanto a establecimiento de objetivos, uso de recursos, procesos, metodologías, beneficiarios y responsables.

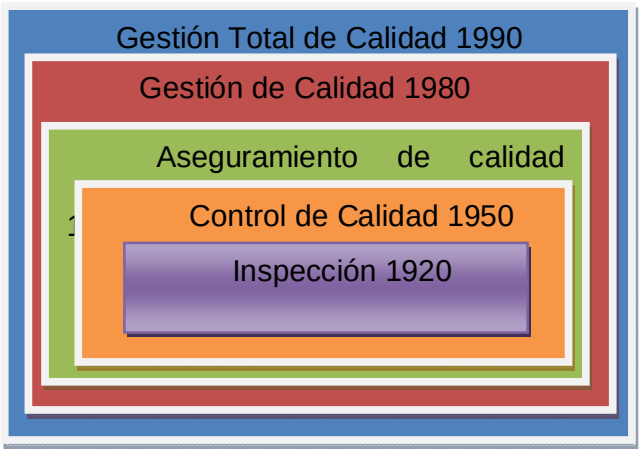

Figura 1. Etapas de la Calidad

Año	Función	Características	Enfoque
1920	Inspección	Orientada a la inspección de la calidad del producto final y descartando los defectuosos.	Producto

[2] Sandor Luis Miranda y Arturo Luis Romero, *La calidad, su evolución histórica y algunos conceptos y términos asociados*.,2006, p.5,6, en http://www.gestiopolis.com

1950	Control de calidad	Detección y prevención de los defectos en el proceso de fabricación. Se aplican conceptos estadísticos para el muestreo y control del proceso. Se aceptan o rechazan lotes de productos.	Proceso
1970	Aseguramiento de calidad	Orientada a acciones preventivas, satisfacción del cliente y reducción de costos considerando a todos los procesos de la cadena productiva.	Sistema
1980	Gestión de Calidad	Orientada a la satisfacción integral de necesidades de los clientes internos y externos en forma participativa y continua hacia el logro de la calidad.	Personas
1990	Gestión Total de Calidad	Extensión del logro de la calidad a todas las actividades que realiza la organización.	Organización

Tabla 1. Evolución Histórica de la Calidad

1.2.2. Calidad del Software

La calidad aplicada al software hace referencia al cumplimiento de la especificación de los requerimientos de negocio o funcionales del cliente en el producto de software.

De acuerdo a la definición del Instituto de Ingenieros Eléctricos y Electrónicos ñLa calidad del software es el grado con el que un sistema,

componente o proceso cumple los requerimientos especificados y las necesidades o expectativas del cliente o usuarioò[3].

ñConcordancia con los requisitos funcionales y de rendimiento explícitamente establecidos con los estándares de desarrollo explícitamente documentados y con las características implícitas que se espera de todo software desarrollado profesionalmenteò[4]

ñEl conjunto de características de una entidad que le confieren su aptitud para satisfacer las necesidades expresadas y las implícitasò[5]

1.2.3. Factores que determinan la Calidad el Software

Según McCall, se establecen factores que deben ser evaluados para la determinación de la calidad del software, mismos que están clasificados en tres categorías relacionadas directamente con el producto de software, así: con las características operativas (operación), la capacidad para soportar cambios (revisión) y la adaptabilidad a nuevos entornos (transición), que se presentan en la Tabla 2.

Categoría	**Características**	
Operación	Corrección	Hace lo que quiero?
	Fiabilidad	Lo hace de forma fiable todo el tiempo?
	Eficiencia	Se ejecutará en mi hardware lo mejor que pueda?
	Seguridad-Integridad	Es seguro?

[3] Std IEEE, 610-1990

[4] Roger S, Pressman, *Ingeniería del software, un enfoque práctico*, Madrid, Mcgraw Hill, 2002, p.135

[5] ISO 8402 (UNE 66-001-92)

	Facilidad de Uso	Está diseñado para ser usado?
Revisión	Facilidad de mantenimiento	Puedo localizar los fallos?
	Flexibilidad	Añadir nuevas opciones?
	Facilidad de Pruebas	Puedo probar todas las opciones?
Transición	Portabilidad	Podré usarlo en otra máquina?
	Reusabilidad	Podré utilizar alguna parte del software en otra aplicación?
	Interoperabilidad	Podrá comunicarse con otras aplicaciones o sistemas informáticos?

Tabla 2. Factores de la Calidad de Software

1.2.4. Métricas de calidad del software.

Las métricas son escalas de unidades sobre las cuales puede medirse un atributo cuantificable, el desarrollo de las medidas que se deben aplicar a los factores de calidad de un producto tienen un grado de dificultad alto. Cada factor de la Calidad ñFcò puede obtenerse como combinación de una o más métricas:

$$Fc = c1 * m1 + c2 * m2 + c3 * m3 \text{ é é } . + cn * mn$$

Donde: *ci* es el factor de ponderación de la métrica *i*, que dependerá de cada aplicación y *mi* es la métrica *i* .

La puntuación en las métricas y en los factores de calidad oscila entre 0 y 10.

Dentro de las métricas para la determinación de los factores de calidad se pueden considerar las que se presentan en la Tabla 3:

Métrica
Facilidad de auditoría
Exactitud
Normalización de las comunicaciones
Completitud
Concisión
Consistencia
Estandarización de datos
Tolerancia a errores
Eficiencia de la ejecución
Facilidad de expansión
Generalidad
Independencia del hardware
Instrumentación
Modularidad
Facilidad de operación
Seguridad
Documentación
Simplicidad
Independencia del sistema software
Facilidad de trazabilidad
Formación

Tabla 3. Métricas de calidad de software

1.3. Control de Calidad (Quality Control)

El control de la calidad es, el método mediante el cual ñpodemos medir la calidad real, compararla con las normas y actuar sobre la diferencia[6]ò

De acuerdo con la norma UNE 66-001-92 el Control de la Calidad tiene dos objetivos fundamentales: mantener bajo control el proceso y eliminar las causas de defectos y define como control de la calidad a las ñtécnicas y actividades de carácter operativo utilizadas para satisfacer los requisitos relativos a la calidadò

El control de calidad tiene por objeto verificar y medir que el entregable tenga la calidad aceptable y que está completo y correctamente finalizado. Se pueden mencionar algunas actividades tales como: revisiones técnicas formales, procesos de pruebas, inspecciones simples, revisiones de pares, entre otras.

1.3.1. Control de calidad del software (Software Quality Control)

Son técnicas y actividades de carácter operativo, utilizadas para satisfacer los requisitos relativos a la calidad, centradas en dos objetivos principales:

a) Mantener bajo control un proceso; y,
b) Eliminar las causas de los defectos en las diferentes fases del ciclo de vida.

En general son las actividades realizadas para evaluar la calidad de los productos desarrollados, se centran en dos objetivos básicos:

- Mantener bajo control un proceso determinado
- Eliminar las causas de los defectos y fallas en las diferentes fases

[6] Juran, Gryna y Bingham, *Manual de Control de la Calidad*, Edit. Reverté S.A., 1990.

del ciclo de vida del software

La obtención de un Software con Calidad implica la utilización de metodologías o procedimientos estándares para el análisis, diseño, programación y prueba del software, que permitan uniformar la filosofía de trabajo, en aras de lograr una mayor confiabilidad, facilidad de mantenimiento y facilidad de prueba, a la vez que eleven la productividad, tanto para la labor de desarrollo como para el Control de Calidad del Software.

La calidad del producto de software abarca los siguientes aspectos:

- Calidad Interna: medible a partir de las características intrínsecas, como el código fuente
- Calidad Externa: medible en el comportamiento del producto, como en una prueba
- Calidad en Uso: durante la utilización efectiva por parte del usuario

1.4. Aseguramiento de la Calidad (Quality Assurance)

El aseguramiento de la calidad según las normas ISO, es el ñconjunto de acciones planificadas y sistemáticas que son necesarias para proporcionar confianza adecuada de que un producto o servicio satisfará los requisitos dados sobre la calidadò

El aseguramiento de la calidad se aplica a los procesos que se utilizan para generar el producto o servicio. Estas acciones deben ser demostrables para proporcionar la confianza adecuada tanto interna como externa de que se cumplen los requisitos del Sistema de la Calidad.

Se puede mencionar algunas actividades tales como: listas de chequeo y auditorías de calidad.

1.4.1. Aseguramiento de la Calidad de Software (Software Quality Assurance)

El aseguramiento de calidad de software según las normas ISO es el ñconjunto de actividades planificadas y sistemáticas necesarias para garantizar que el producto satisfará los requisitos de calidad. El aseguramiento de calidad del software debe diseñarse para cada aplicación antes de comenzar a desarrollarla y no después ò

Aseguramiento de calidad de software ñse enfoca en identificar y evaluar los defectos que puedan afectar al software. Si los errores se pueden identificar de forma temprana en el proceso de software, las características del diseño de software se pueden especificar de modo que eliminarán o controlarán los peligros potenciales, al corregir los errores mucho antes en cada etapa es decir durante el proceso, ahorrando esfuerzos, tiempo y recursos ò[7].

El aseguramiento de calidad consiste en validar los procesos usados para crear los productos. Es una herramienta especialmente útil para administradores y patrocinadores, ya que permite discutir los procesos usados para crear los productos determinando si están dentro del marco de calidad establecido por la empresa. El aseguramiento tiene asociado dos perfiles diferentes: los Ingenieros de Software que realizan el trabajo técnico y un grupo de SQA (Software Quality Assurance) que tiene la responsabilidad de la planificación de aseguramiento de la calidad, supervisión, mantenimiento de registros, análisis e informes.

El aseguramiento de calidad del software se aplica para:

[7] Qualitrain Express, ñAseguramiento de la Calidad de Software ò, 2012, en http://www.qualitrain.com.mx/ Aseguramiento-de-la-Calidad-de-Software

- Métodos y herramientas de análisis, diseño, programación y pruebas
- Inspecciones técnicas formales en todos los pasos del proceso de desarrollo de software
- Estrategias o tipos de pruebas
- Control de documentación del software y de cambios realizados
- Procedimientos de aplicación de estándares
- Métricas de software
- Registro de auditorías
- Elaboración de informes

1.5. Gestión de la Calidad (Quality Management)

Kaoru Ishikawa, un autor reconocido de la gestión de la calidad dio la siguiente definición de Calidad Total: "Filosofía, cultura, estrategia o estilo de gerencia de una empresa según la cual todas las personas en la misma, estudian, practican, participan y fomentan la mejora continua de la calidad".

Siendo la gestión de la calidad un conjunto de actividades y medios establecidos por la dirección de la empresa, para definir e implantar un sistema de la calidad a nivel organizacional; mismo que se implantará mediante la planificación de la calidad, el control de la calidad, el aseguramiento de la calidad y la mejora de la calidad.

La gestión de la calidad se apoya en los modelos de la calidad creados para la aplicación de los sistemas de calidad.

ñLa calidad Total (TQ) es un sistema administrativo enfocado hacia las personas, que busca un incremento continuo en la satisfacción del cliente a un costo real cada vez más bajo. La TQ es un enfoque total de sistemas (no un área ni un

programa independiente) y parte integral de una estrategia de alto nivel; funciona horizontalmente en todas las funciones y departamentos, comprende a todos los empleados, desde el nivel más alto hasta el más bajo, y se extiende hacia y hacia adelante para incluir la cadena de proveedores y la cadena de clientes, la TQ destaca el aprendizaje y la adaptación al cambio continuo como las claves para el éxito de una organización.

El fundamento de la calidad total es filosófico se sustenta en el método científico. La TQ incluye sistemas, métodos y herramientas. Los sistemas permiten el cambio; la filosofía permanece igual. La TQ se fundamenta en valores que resaltan la dignidad del individuo y el poder de acción de la comunidad"[8]

1.5.1. Gestión de Calidad del software (Software Quality Management)

Tiene que ver con la organización interna que ejerce la determinación de los procesos productivos y de las características y cualidades de los productos, es decir es la gerencia o el manejo de los procesos productivos enfocada al mejoramiento continuo.

La gestión de la calidad se aplica a nivel de la empresa, pero se puede tener una gestión de calidad dentro de la gestión de cada proyecto de software.

1.6. Sistema de Calidad

Un Sistema de Calidad es la forma como una organización realiza la gestión empresarial asociada con la calidad. En términos generales, consta de la estructura organizacional junto con la documentación, procedimientos, procesos y recursos que se emplean para alcanzar los objetivos de calidad o establecer la gestión de la calidad y cumplir con los requisitos del cliente.

La dirección de la empresa es la responsable de fijar la política de

[8] James R. Evans y William M. Lindsay, *Administración y Control de la Calidad*, México, International Thompson Editores S.A., 2005, p.17,18

calidad y las decisiones relativas a iniciar, desarrollar, implantar y mantener el sistema de calidad.

Un sistema de calidad tiene varios elementos: Documentación, aspectos físicos (infraestructura, herramientas, computadores, etc.) y el aspecto humano (formación y capacitación del recurso humano, creación y coordinación de equipos de trabajo).

Existen normas, más conocidas como estándares o modelos que enmarcan los conceptos, requisitos y recomendaciones de un Sistema de Calidad, entre los cuales se pueden mencionar las siguientes: ISO, Software Engineering Institute (SEI) ï Capability Maturity Model (CMM) para software, ITIL, SPICE, PSP.

1.6.1. Modelos y Estándares de Calidad de Software

Los Modelos de Calidad son aquellos documentos que integran la mayor parte de las mejores prácticas, proponen temas de administración en los que cada organización debe hacer énfasis, integran diferentes prácticas dirigidas a los procesos clave y permiten medir los avances en calidad.

Los Estándares de Calidad son aquellos que permiten definir un conjunto de criterios de desarrollo que guían la forma en que se aplica la Ingeniería del Software. Los estándares suministran los medios para que todos los procesos se realicen de la misma forma y son una guía para lograr la productividad y la calidad.

- **CMM (Capability Maturity Model):** ñEl CMM tiene como objetivo evaluar los procesos en sus distintos niveles de madurez, identificar los niveles a través de los cuales una organización debe formarse para

establecer una cultura de excelencia en la ingeniería de software[9]. El modelo de madurez de procesos fue generado a través de la experiencia colectiva de los proyectos más exitosos de software, generando así un conjunto de prácticas importantes que deben ser implantadas por cualquier entidad que desarrolla o mantiene software.

- **ISO (International Standard Organization):** La norma ISO/IEC 9003 proporciona una guía necesaria en las organizaciones para la aplicación de la ISO 9001 a la adquisición de suministro, desarrollo, operación y mantenimiento de software y sus servicios relacionados. Identifica todos los aspectos que deberían ser tratados y es independiente de la tecnología, modelos de ciclos de vida, procesos de desarrollo y estructuras organizacionales. La norma ISO 9001, especifica los requisitos para un sistema de gestión de la calidad cuando una organización necesita demostrar su capacidad de proporcionar de forma coherente productos que satisfagan los requisitos del cliente y aspira a aumentar su satisfacción a través de la aplicación eficaz del sistema, incluyendo los procesos para la mejora continua del sistema y el aseguramiento de la conformidad con los requisitos y de acuerdo a las reglamentaciones existentes.
- **PSP (Personal Software Process) /TSP (Team Software Process):** El PSP es una tecnología que tiene como justificación la premisa de que la calidad de software depende del trabajo de cada uno de los ingenieros de software y de aquí que el proceso diseñado debe ayudar a controlar, manejar y mejorar el trabajo de los ingenieros. El objetivo de PSP es

[9] Alfredo Weitzenfeld, *Ingeniería de Software orientada a objetos con UML, Java e Internet,* Edit. Thomson, p. 57

lograr una mejor planeación del trabajo, conocer con precisión el desempeño, medir la calidad de productos y mejorar las técnicas para su desarrollo. La instrumentación de esta tecnología consiste en lo que se denomina ñevolución del PSPò. El TSP se concentra en los aspectos del desarrollo de software realizados por equipos de trabajo, definiendo aspectos como la asignación y control de tareas para los diversos miembros del equipo.

- **SPICE (Software Process Improvement and Capability dEtermination):** El SPICE es un modelo de madurez de procesos internacional. SPICE fomenta productos de calidad, promueve la optimización de procesos y facilita la evaluación del producto a través de los procesos de desarrollo. SPICE tiene diversos alcances, se aplica tanto a nivel directivo como a nivel de usuarios para asegurar que el proceso se encuentra alineado con las necesidades del negocio, apoya en que los proveedores de software tengan que someterse a una sola evaluación para aspirar a nuevos negocios y busca que las organizaciones de software dispongan de una herramienta universalmente reconocida para dar soporte a su programa de mejoramiento continuo.
- **PEMM (Performance Engineering Maturity Model):** El PEMM presenta un modelo para evaluar los niveles de integración, aplicación, ejecución y diseño, llamado ingeniería de la ejecución del modelo de madurez. Al igual que SPICE se apoya en el modelo de madurez de capacidades CMM. El objetivo de PEMM es poder evaluar la Ejecución de la Ingeniería así como la integración del proceso. El modelo sirve tanto

para evaluar una organización como los propios desarrollos de procesos tecnológicos específicos. Sirve también para definir el criterio al escoger un proveedor de software para los productos críticos o semi-críticos de la compañía.

- **TickIt:** Desarrollado por el Departamento de Comercio e Industria del Reino Unido, surge por la poca adopción de las normas internacionales de calidad ISO 9000 para el área de desarrollo de software. TickIt es primordialmente una guía que presenta las estrategias para lograr la certificación en la producción de software a través de la interpretación de los estándares ISO. Los objetivos principales de TickIt son, además de desarrollar un sistema de certificación aceptable en el mercado, estimular a los desarrolladores de software a implementar sistemas de calidad, dando la dirección y guías necesarias para tal efecto.
- **McCall.-** Fue el primer modelo en ser presentado en 1977, y se originó motivado por USA Air Force y DoD. McCall está organizado sobre tres tipos de características de calidad:
    - Factores (especificar)
    - Criterios (construir)
    - Métricas (controlar)

  Este modelo organiza 11 factores en tres ejes o puntos de vista desde los cuales el usuario puede contemplar la calidad de un producto organizado en 23 criterios. Cada factor tiene asociado sus respectivos criterios.
- **FURPS.-** El modelo FURPS propuesto por Robert Grady y Heweltt Packard Co (HP) cuenta con 5 características de calidad del software:

(1) Funcionalidad, (2) Facilidad de uso, (3) Confiabilidad, (4) Performance y (5) Facilidad de soporte. Además plantea 2 categorías de requerimientos, las cuales son:

- Requerimientos funcionales (F): especifican funciones que el sistema debe ser capaz de realizar, sin tomar restricciones físicas a consideración, y se definen a través de las entradas y salidas esperadas.
- Requerimientos no funcionales (URPS): Usability (Facilidad de uso), Reliability (Confiabilidad), Performance y Supportability (Facilidad de soporte). describen atributos el sistema o atributos del ambiente del sistema.

- **Seis Sigma.-** Puede ser visto como una filosofía de gestión que utiliza la medición centrada en el cliente y el establecimiento de metas para crear resultados finales. Es un enérgico llamamiento para escuchar la voz del cliente y la conversión de las necesidades del cliente en requisitos mensurables
- **RUP (Rational Unified Process).-** Es una metodología para el desarrollo de software, basada en los pilares de buenas prácticas entre las cuales están el desarrollo iterativo e incremental, administración de requerimientos, uso de arquitecturas basadas en componentes, modelado visual, verificación de la calidad de software y control de cambios. Se aplica bajo el criterio de Fases e Iteraciones, entre las fases se tiene Inicio, Elaboración, Despliegue y Transición, y disciplinas de Modelado de Negocio, Requerimientos, Análisis y diseño, Implementación, Pruebas, Despliegue, Gestión de cambio y

configuración, Gestión de proyecto y Entorno.

RUP no es un sistema con pasos firmemente establecidos, sino un conjunto de metodologías adaptables al contexto y necesidades de cada organización.

- **ITIL (Information Technology Infrastructure Library).-** Es una colección de documentos públicos que contienen un modelo de referencia basado en procesos y las mejores prácticas de la industria de TI que facilita la Administración de Servicios de una organización de TI con Calidad.

  ITIL como modelo de gestión de servicios de TI cubre las áreas de una organización como redes, operaciones, organización, telefonía, soporte técnico, help desk, telecomunicaciones, etc. Mientras que el desarrollo es exclusivamente software aplicativo, sin embargo éste no es habilitado si no tiene una plataforma de cliente/usuario la cual es montada sobre infraestructura de TI como es: servidores, equipos PC, mainframes, workstations, red, switches, routers, sistemas de enfriamiento, etc. Sin los cuales el software no puede operar.

  Debido a esta relación tan estrecha entre hardware y software es necesario poder administrar adecuadamente los ambientes de producción, desarrollo y pruebas en donde pueda ser ejecutada una aplicación cualquiera. De ahí la importancia de la disponibilidad y capacidad de la infraestructura que soporta los sistemas

  Las mejores prácticas para el desarrollo de software se alinean con ITIL considerando:

  - Administración de requerimientos

- Arquitectura de componentes.
- Modelado visual
- Verificación continua de la calidad
- Control de cambios y configuraciones

1.6.2. Certificación de la Calidad (Quality Certification)

Un sistema de certificación de calidad permite una valoración independiente que debe demostrar que la organización es capaz de desarrollar productos y servicios de calidad.

Los pilares básicos de la certificación de calidad son tres: una metodología adecuada, un medio de valoración de la metodología y la metodología utilizada y el medio de valoración de la metodología deben estar reconocidos ampliamente por la industria.

1.7. Sistema de Gestión de Calidad (SGC)

Según la norma ISO 9000:2005, define al Sistema de Gestión de la Calidad como ñel conjunto de elementos mutuamente relacionados o que interactúan para establecer la política y los objetivos con el fin de dirigir y controlar una organización con respecto a la calidadò[10].

Dentro de los elementos de un sistema de gestión de calidad se encuentra:

- Estructura Organizacional
- Planificación (Estrategia)
- Recursos
- Procesos
- Procedimientos

[10] Pablo Alcalde San Miguel*, Calidad 2da Edición*, España, Ed. Paraninfo, 2010, p.78

CAPITULO II

SECTOR DE DESARROLLO DE SOFTWARE EN ECUADOR

2.1. Mercado de Software en Ecuador

A nivel Internacional el Ecuador se encuentra entre los países que hacen de la información y el conocimiento el recurso base para el desarrollo de su economía. Uno de los sectores que en los últimos años ha sufrido cambios positivos es el de Software, con productos creados y desarrollados en el país, en la década de los ochenta y noventa alcanzo este sector tuvo su auge, por lo que el Ecuador paso a ser un referente regional.

La Asociación Ecuatoriana de Software (Aesoft), se planteo un proyecto en conjunto con el Ministerio de Industrias y otras Instituciones Gubernamentales, para lograr que los productos informáticos nacionales recuperen el reconocimiento internacional alcanzado en décadas pasadas, esta meta está planteada cumplirla para el 2013.

Como se muestra en la Figura 2, en el país se cuenta con 265 empresas registradas en la Aesoft; dedicadas a la industria del software, de ellas el 35% son ecuatorianas.[11] En la Figura 3, del universo de empresas de este sector entre nacionales y extranjeras; el 63% está en la provincia de Pichincha, el 28% en Guayas, el 4% en Azuay, el 2% en Loja y el 1% en El Oro, y 2% para el resto del país. En el Ecuador, existen unas 700 empresas o consultoras independientes con una participación creciente en el mercado del 12% anual[12].

[11] Aesoft, ñEstudio de mercado del sector de software y hardware en Ecuadorò, Quito, septiembre 2011, p. 6 en http://www.aesoft.com.ec/www/index.php/noticias/183-estudio-de-mercado-2011

[12] Prom Peru, ñPerfil del Mercado de Software en el Ecuadorò, 2011, p. 4,6,8,9,11, en http://www.siicex.gob.pe/siicex/resources/sectoresproductivos/36471667rad3D22C.pdf

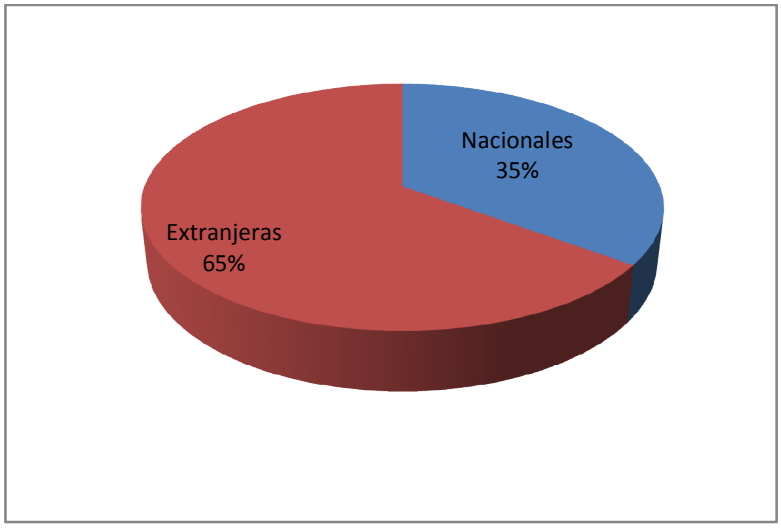

Figura 2. Composición empresarial mercado de software ecuatoriano

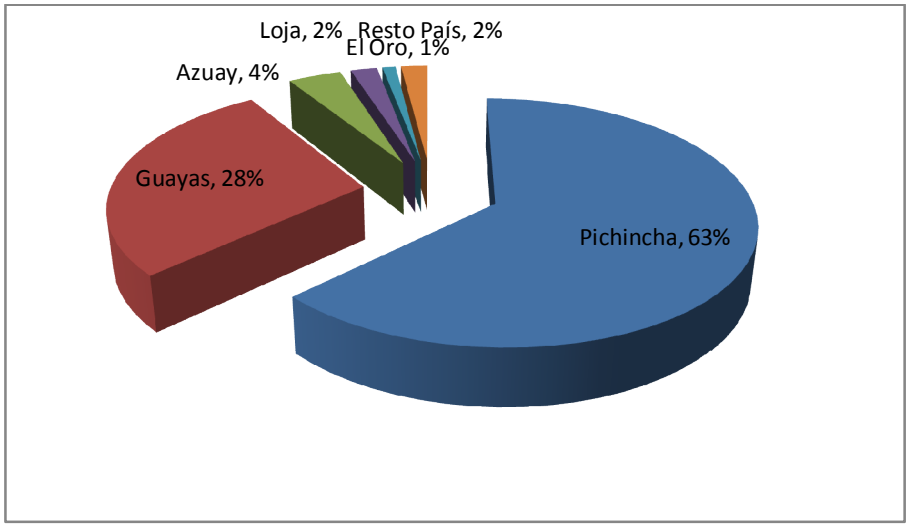

Figura 3. Distribución de empresas por provincias

En Pichincha, las empresas son consideradas como medianas y grandes, a diferencia de las del resto del país. El 46% del total de las empresas se dedican a las consultorías, desarrollo y venta de sus aplicaciones en conjunto. Las empresas pequeñas en su mayoría brindan desarrollo y venta de software (92,8%). De las empresas medianas, a más del desarrollo y venta de software, el 63,3% se dedican a consultorías, auditorías informáticas mientras que las empresas grandes, desarrollan y venden su propio software,

el 80% se dedican a esta actividad. Las pruebas al software se las realiza por dos grupos de personas, primero el personal de la compañía y luego el cliente o posible usuario[13].

En la Superintendencia de Compañías, se encuentra registradas 651 empresas con una participación creciente en el mercado del 12% anual.

En cuanto a la aplicación de estándares de calidad, como se muestra en la Figura 4, el 36.3% de las empresas usan estándares de calidad en el desarrollo de software, de esos solo el 24,6% tenía estándares internacionales.

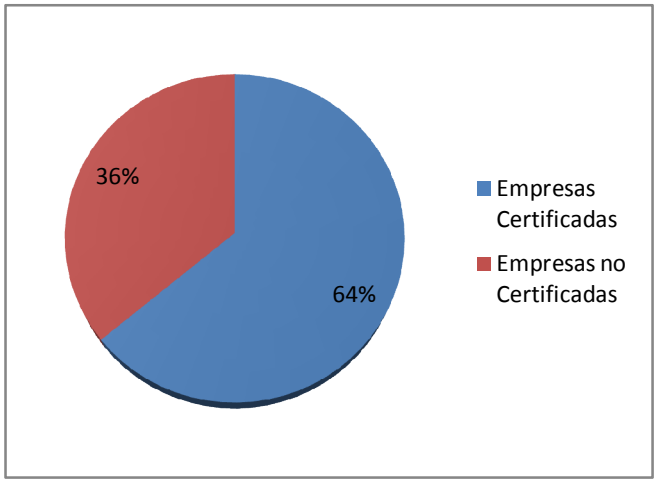

Figura 4. Empresas con Certificación de Calidad

2.2. Ingresos del Sector Software en Ecuador

Según el estudio realizado en el año 2011 por la Aesoft[14], en el Ecuador no existen datos oficiales del sector de software y hardware, sin embargo este utiliza un estimado de fuentes primarias como: la base de datos del Servicio de Rentas Internas (SRI) y las estadísticas del Banco Central del Ecuador (BCE). De la información generada por el SRI, el sector de software en Ecuador en el 2009 alcanzó ingresos de $260 millones. Durante los últimos

[13] ProChile ï Ecuador, ñEstudio de Mercado Servicio Desarrollo de Software en Ecuadorò, Ecuador, 2012, p.6,7

[14] (Aesoft, ñEstudio de mercado del sector de software y hardware en Ecuadorò, p. 17-20)

cinco años la industria ha presentado una importante evolución con una tasa de crecimiento anual compuesta entre 2004 y 2009, del 22.4%. Siendo así en el 2004 de $95 millones a $276 millones en 2009.

En la Figura 5, se puede ver el crecimiento que ha tenido el sector de software en la economía del país entre el 2004 y 2009.

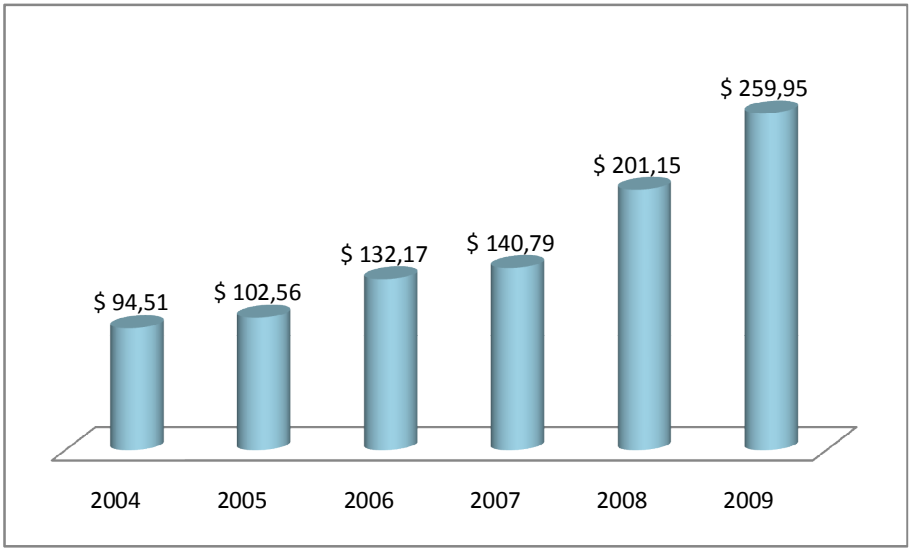

Figura 5. Ingresos Totales del Sector Software en el Ecuador

La oferta de Software en el Ecuador se divide en algunas áreas de especialización, así: Financiero bancaria, administrativo financiero, inteligencia empresarial, aplicaciones BPM (Bussiness Process Management), Tributario, e-Learning, Consultorías y Outsourcing.

El sector ofrece aproximadamente 8,900 puestos de trabajo. Alrededor de US$ 31.7 Millones anuales vienen de Operaciones de empresas internacionales ubicados en Ecuador, mientras US$ 35 Millones anuales vienen de desarrolladores ecuatorianos. La gran mayoría del Software desarrollado en Ecuador está destinado para el mercado interno. El 70% de las empresas Ecuatorianas no exportan y los que exportan no están aprovechando la demanda global. Las ventas de software ecuatoriano se han incrementado en

un 30% entre 2006 y 2008, mientras que el comercio de los programas extranjeros ha crecido un 11%.

Es la industria de todas las industrias, su desarrollo impacta en la competitividad de la Banca, Municipios, Instituciones Educativas; es decir impacta en todas las actividades tanto privadas como gubernamentales

Según la Aesoft, el Gobierno Ecuatoriano es el principal cliente; es así que el 95% de la actividad de consultoría corresponde a requerimientos gubernamentales.

2.3. Certificaciones de Calidad de empresas de software en Ecuador[15]

Las empresas ecuatorianas productoras de programas informáticos cuentan ya con una nueva certificación internacional, otorgada por el Centro de Gerencia Europeo (EUMC, por sus siglas en inglés), con el aval del Ministerio de Industrias (MIPRO).

Esto les permitirá a las empresas convertirse en entes más competitivos y así aumentar sus exportaciones, considerando que el sector es considerado dentro de los prioritarios para la industria nacional.

El software está considerado como un área económica transversal, pues todos los negocios dependen de un programa informático para su desarrollo.

En el Ecuador existen casi un centenar de compañías desarrolladoras de software, que en su mayoría son microempresas, pues en cada una de ellas laboran menos de 20 personas, aunque hay un reducido número de empresas de mayor tamaño.

Priscilla Rodríguez, gerente de EUMC, aseveró que se ha seleccionado el Modelo de Excelencia EFQM para certificar en una primera etapa a 13

[15] European Management Center, ñIndustria Local de Software exporta 30 millones en el 2011ò, 23 de diciembre del 2011, en http://www.eumcecuador.com/noticias_amp.php?id_noticia=14,

compañías, pues se ajusta a las características de las compañías del sector.

El modelo se aplica a cualquier industria que genere valor agregado, con énfasis en la informática, y consiste en compartir las mejores prácticas en la gestión de las tecnologías de la información y comunicación (TIC), en reconocer el valor agregado, entrenarse en áreas específicas y realizar una autoevaluación anual de sus avances, esto les permitirá agilitar sus procesos y llegar a muchos más mercados.

El EFQM tiene como misión impulsar la excelencia en las organizaciones públicas, privadas y educación, está presente actualmente en 38 países y más de 30.000 instituciones, y cuenta con memorando de entendimiento con Naciones Unidas para promover las mejores prácticas de gestión empresarial.

El Gobierno Nacional a través del Ministerio de Industrias, califica al sector del software como "uno de los ejes para catapultar al Ecuador hacia un proceso de diversificación de su oferta exportable y acceso a los mercados internacionales". Es así que, el Estado realizará una inversión de $12 millones para mejorar la calidad de los productos y servicios ecuatorianos, pues potencializará su promoción y certificación para el 2013.

CAPITULO III
ANALISIS SITUACIONAL DE LA EMPRESA

3.1. Introducción

En el presente capítulo se realizará el estudio de la empresa SYSGENSA, misma que está ubicada en Quito, y que tiene una categoría de pequeña empresa al estar conformada por 11 personas.

Esta empresa forma parte del sector de desarrollo de software en el Ecuador, y por ser este sector en los últimos años, un sector que favorece al crecimiento empresarial y económico; es necesario que SYSGENSA analice sus debilidades, y mejore sus procesos con el objetivo de ser más competitiva para aprovechar así las oportunidades del mercado ecuatoriano.

El estudio de la empresa se lo realizo entre julio y diciembre del 2012, con el apoyo de la Gerencia y respectivo equipo de trabajo. En este período, se tuvo 1 proyecto de desarrollo para el sector público conformado por 2 Sistema; el primero de los cuales (Sistema de Tesorería) estaban en una etapa de pruebas y el segundo en la de implementación de la aplicación (Sistema Administrativo-Becas). Se destinaron 8 semanas de seguimiento de novedades (defectos/fallos) en la evolución de los sistemas indicados, en las primeras 4 semanas se tomó la información aplicando el proceso actual y las siguientes considerando ajustes propuestos de mejora.

3.2. Descripción de la Empresa

SYSGENSA es una empresa ecuatoriana dedicada al desarrollo de Sistemas Computarizados, Asesoría, Capacitación y Auditoría Informática, formada en 1994.

Los proyectos que el personal de SYSGENSA ha realizado a lo largo de diez y ocho años de experiencia profesional son un testimonio de los altos

patrones de excelencia, seriedad, integridad y compromiso hacia sus clientes.

3.2.1. Misión

Proveer soluciones informáticas que contribuyan al progreso y bienestar de la sociedad mediante un eficiente y oportuno apoyo a la gestión institucional, utilizando herramientas de tecnología de punta y con estándares de calidad

3.2.2. Visión

Crecimiento a nivel regional como proveedora estratégica de servicios informáticos de calidad, en base a la capacidad de generar productos innovadores y efectivos, logrando la satisfacción de sus clientes.

3.2.3. Objetivos

- Proveer soluciones informáticas de calidad a sus clientes.
- Participar activamente en la comunidad Desarrollo de Software.

3.2.4. Principios y Valores

- Compromiso con la calidad
- Responsabilidad
- Honestidad
- Confidencialidad
- Mejora continua de procesos y servicios.
- Capacitación y aprendizaje continuo.
- Trabajo en equipo

3.2.5. Análisis FODA

Fortalezas

- Compromiso del equipo de trabajo
- Calidad del servicio
- Cumplimiento en el servicio

- Ampliación de servicios
- Costos reducidos al utilizar herramientas libres.

Oportunidades

- Crecimiento del mercado
- Existencia de parques tecnológicos en la región
- Alta demanda de soluciones tecnológicas.

Debilidades

- Poco capacidad de inversión
- Alta rotación de personal
- Poca capacitación
- Tasa de retorno prolongado
- Poca cultura en la utilización de metodologías de ingeniería de software

Amenazas

- Alta competencia
- Falta de cultura del proceso de pruebas por parte las empresas desarrolladoras.

3.2.6. Estructura Organizacional

Organigrama:

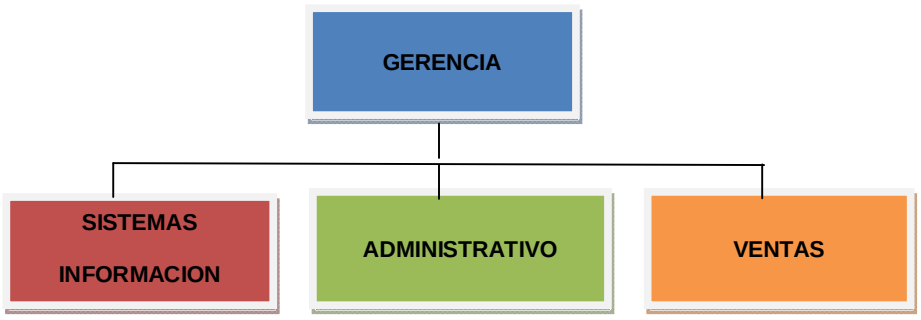

Figura 6. Estructura Organizacional SYSGENSA

Descripción de cargos

- Cargo: Gerente General

Estudios en: Informática y Computación, Administración de Empresas

Perfil: Profesional con conocimientos y experiencia en planificación, supervisión, y coordinación de las actividades de la empresa a nivel estratégico, táctico toma de decisiones aplicando el Cuadro de Mando Integral.

- Cargo: Gerente de Sistemas de Información

Estudios: Ingeniería de Sistemas

Perfil: Persona con conocimientos y experiencia en Ingeniería de Software, Administración de Proyectos, Conocimientos de metodologías y buenas prácticas para desarrollo de soluciones informáticas. Liderazgo de Investigación y Desarrollo de aplicaciones nuevas de la tecnología.

- Cargo: Gerente de Administrativo

Estudios en : Finanzas, administración de talento humano

Perfil: Persona con conocimientos financieros, contables y legales, además de gestión del talento humano.

- Cargo: Gerente de Ventas

Estudios en: Ingeniería de Sistema, o mercadeo

Perfil: Persona con habilidades comerciales, con gran capacidad de comunicación y poder de convencimiento. Que acepte retos y que pueda crear, innovar e implementar estrategias en beneficio de la empresa

En el Departamento de Sistemas de Información se tienen grupos de trabajo organizados de acuerdo al siguiente esquema:

Equipos de Trabajo	Número de Personas
Gerentes de Proyecto	1
Análisis	1
Arquitectura	1
Desarrollo	3
BDD	1
Control de Calidad	1
Operaciones (Infraestructura/SW Base)	1

Cabe mencionar que en este departamento se encuentran profesionales con estudios en ingeniería de sistemas o afines, y que cuando la situación así lo amerita, una misma persona puede apoyar en los diferentes equipos.

3.3. Proceso de Desarrollo de Software

El proceso o ciclo de vida de desarrollo de software en SYSGENSA, está basado en la Metodología RUP (Rational Unified Process de IBM), misma que es un conjunto de buenas prácticas adaptables al contexto y necesidades de las organizaciones.

La metodología RUP organiza las tareas del ciclo de vida de software en dos dimensiones[16]: El eje horizontal representa tiempo y demuestra el aspecto dinámico del proceso, se expresa en términos de ciclos, de fases, de iteraciones, y de hitos o milestones y el eje vertical representa el aspecto estático del proceso: cómo se describe en términos de actividades, de dispositivos, de trabajadores y de flujos, que se presenta en la Figura 7.

[16] IBM Coporation, IBM Rational Method Composer (Rational Unified Proccess Versión 7.0.1), 2005

El ciclo de vida del software está particionado en ciclos, cada ciclo trabaja en una nueva generación del producto. El RUP divide un ciclo de desarrollo en cuatro fases consecutivas: Inicio, Elaboración, Construcción y Transición.

Cada fase constituye un eslabón bien definido, un punto en el tiempo en el cual ciertas decisiones críticas deben tomarse.

Figura 7. Fases y disciplinas de RUP

3.3.1.Fase de inicio

Durante la fase del inicio, se establece el caso de negocio para el sistema y delimita el alcance del proyecto. Para lograr esto debe identificar todas las entidades externas con las cuales el sistema interactúe (los actores) y definir la naturaleza de esta interacción a un nivel alto. Esto implica identificar todos los casos de uso y describir sólo los más significativos. El caso de negocio incluye criterios de éxito, la evaluación de riesgos, la estimación de los recursos necesarios y un plan de la fase que muestre las fechas previstas e hitos importantes.

3.3.2.Fase de elaboración

El propósito de la fase de elaboración es analizar el dominio del problema, establecer una fundación arquitectónica sana, desarrollar el plan del proyecto y eliminar los elementos del riesgo más alto del proyecto. Para lograr estos objetivos, se debe tener una visión completa del sistema. Las decisiones arquitectónicas tienen que tomarse con una comprensión cabal del sistema: su alcance, funcionalidad importante y requerimientos no funcionales tales como requerimientos de performance.

3.3.3. Fase de construcción

Durante la fase de la construcción, todos los componentes y características restantes se desarrollan, se integran en el producto y se prueban a fondo. La fase de la construcción es, en cierto sentido, un proceso de fabricación donde el énfasis se pone en manejar los recursos y controlar las operaciones para optimizar costos, tiempos y calidad. Una arquitectura robusta y un plan comprensible están íntimamente relacionados; es decir, una de las cualidades críticas de la arquitectura es su facilidad de la construcción. Ésta es una razón por la que durante la fase de elaboración, se pone énfasis en el desarrollo equilibrado de la arquitectura y del plan.

3.3.4. Fase de transición

El propósito de la fase de la transición es justamente la transición del producto de software al ambiente de producción. Una vez que el producto se haya entregado al usuario final, surgen algunos temas que llevan al desarrollo de nuevas versiones, a corregir errores, o a terminar algunas características que habían sido pospuestas.

Se ingresa a esta fase cuando el producto está lo suficientemente maduro para comenzar a pasar a producción. Esto requiere que un cierto

subconjunto del sistema se encuentre en un nivel aceptable de la calidad y que la documentación del usuario está disponible de modo que la transición proporcione resultados positivos para todas las partes.

Adicionalmente, esta metodología plantea una serie de entregables para cada fase, llamados artefactos, mismos que son documentos, código fuente o ejecutable de la aplicación desarrollada.

Entre los documentos que se están generando en SYSGENSA se tienen:

- Visión del negocio
- Glosario
- Arquitectura del Negocio
- Casos de uso
- Casos de prueba
- Plan del proyecto
- Plan de Configuración y cambios
- Código Fuente
- Manuales de Instalación y Configuración
- Manuales de Usuario

Cada iteración dentro de una fase, es una mini cascada en la cual se ejecuta un ciclo completo de desarrollo limitado a los hitos a lograr; es decir, en cada iteración se realizan las siguientes actividades:

- Modelado del Negocio
- Análisis de Requerimientos
- Análisis y Diseño de Sistemas
- Implementación , y
- Pruebas.

El Proceso Unificado de Rational (RUP) que se representa en la Figura 8, utiliza enfoques que son conocidos como ñ*mejores prácticas*ò en la industria de desarrollo de software, así:

Figura 8. Buenas Prácticas para Desarrollo de Software

- **Desarrollo iterativo e incremental**.- Es la generación del sistema informático en partes o releases que son progresivos y frecuentes que permiten un mayor involucramiento y opinión de los usuarios, el uso del concepto de iteraciones, para generar releases ejecutables, facilitar la detección temprana de inconsistencias o requerimientos incompletos, diseño y construcción del sistema.
- **Administración de requerimientos.-** Permite conocer o difundir requerimientos claramente definidos, priorizar y monitorear efectivamente, evaluar objetivamente funcionalidad y rendimiento y detección de inconsistencias más fácilmente.
- **Uso de arquitecturas basadas en componentes.-** Se define tempranamente una arquitectura adecuada para el sistema, debidamente probada, antes de continuar con el desarrollo de todo el sistema, se promueve la reutilización de software y desarrollo basado en componentes nuevos o existentes (módulos que juntos completan una funcionalidad del sistema).
- **Modelado Visual**.- La abstracción visual facilita la comprensión de los

diferentes elementos del software, comprender los requerimientos, ver las relaciones entre sus partes y mantener consistencia entre los requerimientos, el diseño y construcción. El estándar visual utilizado es conocido como UML (Lenguaje de Modelado Unificado)

- **Verificar la calidad de software**.- La evaluación de la calidad de un sistema respecto a los requerimientos funcionales y no funcionales, confiabilidad y rendimiento. La actividad principal son las pruebas, mismas que permiten encontrar fallas o defectos antes de la salida a producción. Además, se contempla el aseguramiento de la calidad en todas las actividades que conforman el ciclo de desarrollo del sistema.
- **Controlar los cambios**.- La capacidad de administrar los cambios es de suma importancia, se debe controlar, rastrear y monitorear estos para lograr el desarrollo de las distintas iteraciones planteadas. El control de cambios se realiza sobre todos los elementos de software y se administra la conformación de releases.

3.4. Organización de Proyectos

La política de SYSGENSA para atender a sus Clientes con los Proyectos de desarrollo de Software, es armar equipos de trabajo conformados por diferentes roles así:

- Gerente de Proyecto
- Arquitectos
- Analistas de Sistemas
- Desarrolladores
- Analistas Control de Calidad
- Documentadores

- Capacitadores
- Implantadores

El personal que participe en los proyectos puede desempeñar uno o más roles según avance el proyecto. Cada rol tiene asignadas funciones y responsabilidad específicas por cada fase de desarrollo, de acuerdo con el detalle que se presenta en la Tabla 4:

Fase	Rol	Función/Responsabilidad
Inicio	Gerente de Proyecto	Planificar las actividades que permitan cumplir con los objetivos del proyecto en el tiempo y costo esperados.
	Analista de Sistemas	Apoyar en definición de alcance del proyecto dentro del Plan del Proyecto.
	Arquitecto	Apoyar en definición de tiempos de implementación por cada actividad del proyecto dentro del Plan del Proyecto.
Elaboración	Gerente de Proyecto	Controlar el cumplimiento sistemático de las actividades de análisis y diseño que permitan cumplir con los objetivos del proyecto en el tiempo y costo planificados.
	Analista de Sistemas	Realizar la especificación y análisis de los requisitos. Levantar todos los entregables de análisis y diseño.
	Arquitecto	Definir la arquitectura base para el desarrollo del proyecto.

Construcción	Gerente de Proyecto	Controlar el cumplimiento sistemático de las actividades de desarrollo, ejecución de plan de pruebas que permitan cumplir con los objetivos del proyecto en el tiempo y costo planificados.
	Analista de Sistemas	Guiar a los desarrolladores en la implementación de cada caso de uso. Levantar casos de prueba.
	Arquitecto	Establecer la estructura base para el desarrollo. Apoyar a los desarrolladores en mejores prácticas de programación y resolución de problemas. Configurar ambientes de trabajo.
	Desarrollador	Implementar los casos de uso asignados.
	Analista Control de Calidad	Controlar la calidad del producto y funcionalidad de la aplicación. Ejecutar el plan de aseguramiento de calidad y plan de pruebas.
	Documentador	Levantar los documentos/entregables detallados en el Plan de Gestión del Alcance del Proyecto.
Transición	Gerente de Proyecto	Controlar el cumplimiento sistemático de las actividades de ejecución del plan de pruebas, paso a producción y cierre del

		proyecto que permitan cumplir con los objetivos del proyecto en el tiempo y costo planificados.
	Arquitecto	Apoyar a los desarrolladores en resolución de problemas. Configurar ambientes de trabajo.
	Desarrollador	Ajustar el desarrollo de casos de uso. Paso a producción.
	Analista Control de Calidad	Ejecutar plan de pruebas. Acompañamiento en pruebas funcionales.
	Documentador	Levantar los documentos/entregables detallados en el Plan de Gestión del Alcance del Proyecto.

Tabla 4. Roles participantes de proyectos - SYSGENSA

3.5. Diagramas de Procesos de ciclo de desarrollo SYSGENSA

La Figura 9, muestra el diagrama de proceso general, para el ciclo de desarrollo aplicado en la empresa, como se puede visualizar se cumplen las fases básicas de dicho ciclo: Modelado de Negocio, Análisis de requerimiento y Diseño, Implementación, Pruebas y Despliegue.

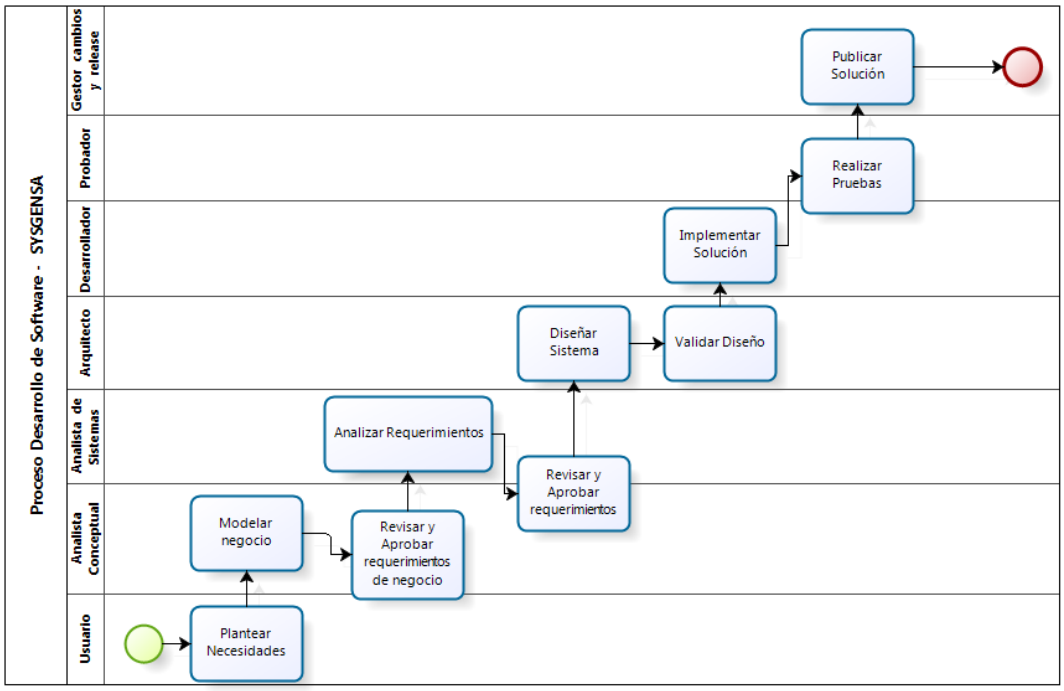

Figura 9. Diagrama de Proceso de Desarrollo de Software ï SYSGENSA

Dentro de las actividades propias del proceso de desarrollo de software, se tienen actividades de soporte que están presentes durante todas las fases o etapas para generar el producto de software, así:

- Gerencia del proyecto
- Gestión de la calidad de los productos
- Gestión de la configuración
- Verificación y Validación
- Entrenamiento
- Documentación

Dentro de cada proceso se han identificado problemas que aparentemente son menores, pero que sin embargo al pasar de una fase a otra toman peso llegando a reflejarse en el producto cuando este pasa a la etapa de pruebas o despliegue.

A continuación se describe cada subproceso:

- **Modelado de Negocio**

Descripción: En este subproceso se plantea la necesidad del cliente (planteamiento del problema) para construir o mejorar un sistema de software, vista desde el enfoque del negocio por parte del cliente. Se plantea la definición del sistema de negocio, identificando la visión, el dominio, los objetivos, las reglas del negocio, la cadena de valor, los diagramas, jerarquía de procesos y actividades, así como la arquitectura del negocio, actores e involucrados y la jerarquía de procesos.

En esta etapa, los analistas de negocio son los responsables de generar la documentación de soporte de estas actividades, mismos que trabajan directamente con los usuarios que representan al cliente y que constituyen los expertos en el dominio del problema.

- **Análisis de Requerimientos**

Descripción: A partir de la documentación que respalda la fase previa ñModelado de Negocioò se identifica en detalle las necesidades de información y automatización que los usuarios requieren. El análisis de requerimientos es un proceso de descubrimiento y refinamiento de las necesidades del cliente, se dividen en funcionales y no funcionales, donde los funcionales corresponden a la condición o capacidad funcional (de negocio) que debe tener el sistema según lo requiera el usuario de acuerdo al objetivo que se quiere alcanzar y los no funcionales que corresponden a la capacidad que debe tener un sistema para cumplir un contrato o estándar que no se relaciona directamente con la funcionalidad del negocio pero que debe estar presente como base de operación para el producto final.

Esta tarea que aparentemente resulta sencilla, es la tarea más

importante, puesto que las destrezas relacionadas a creatividad, comprensión y comunicación están presentes para garantizar que la interpretación que deben realizar los analistas de sistemas a las necesidades de los usuarios sea traducida de forma completa, consistente, correcta y bien establecida.

El analista estudia la documentación del Modelado de Negocio, levanta los requerimientos y los traduce a casos de uso de sistemas (especificación de requerimientos), para posteriormente revisarlos con los usuarios para asegurarse que lo que consta en estos casos de uso, es lo que realmente el usuario necesita y proceder a su aprobación. Además, se inicia con la especificación de casos de pruebas para los requerimientos.

En esta fase, se presentan la mayor parte de los problemas cuando no se llega a consensuar las necesidades del cliente, o peor aún, cuando los clientes tienen cierto grado de dificultad al comunicar que es lo que realmente requieren y aceptan lo que el analista de sistemas les presente, por simple formalismo y cumplir con las tareas asignadas por parte de la Organización Cliente.

- **Diseño de software**

Descripción: En este proceso se diseña el sistema de software, preparándose para cumplir los requisitos que se definieron en las fases de modelado de negocio y análisis de requerimientos. Se plantea el modelo arquitectónico del software, modelo de datos, modelo de interfaces, modelos de componentes, modelos de casos de pruebas.

En esta etapa participan arquitectos de aplicación y base de datos, expertos del negocio, analistas de sistemas e incluso programadores.

- **Implementación/Construcción**

Descripción: En esta etapa se traducen las especificaciones de los requerimientos y de diseño en el producto de software, se generan algoritmos y código fuente; se siguen los estándares impuestos por los arquitectos. Adicionalmente, se realizan las pruebas básicas del programador por cada componente del sistema que se cree y se diseñan las pruebas unitarias e integración.

- **Pruebas**

Descripción: En esta etapa se asegura que el software cumpla con la funcionalidad definida en la etapa de análisis de requerimientos, se corrobora que los datos de entrada y salida sean coherentes con los analizados. Estas pruebas, se realizan en base a los casos de pruebas que se generaron conjuntamente con los requerimientos.

Se diseñan las pruebas de funcionalidad, integración, carga y aceptación del sistema

En este proceso se tiene problemas debido a que no se completa la especificación de casos de pruebas con todos sus escenarios antes y por tanto se requiere una improvisación de las mismas, adicionalmente, el tiempo adicional que se tome en las fase previas afectan a la planificación esta fase.

- **Despliegue**

Descripción: Consiste en transferir el sistema de un ambiente de desarrollo/pruebas a un ambiente de producción, una vez que se entrega el producto, entra en modo de mantenimiento, esto incluye mejoras y corrección de errores. Esta actividad consume mucho tiempo de los recursos, incluso hasta un 60% de tiempo de los recursos que desarrollaron el sistema.

Considerando que todos los proyectos requieren documentación desde

el principio y que esta sufre muchos cambios, es necesario desde el principio disponer de una estrategia para mantener los documentos que se generen y las versiones de estos. Este proceso es denominado ñGestión de la Configuraciónò. Este proceso pese a estar definido en SYSGENSA no es totalmente respetado.

3.6. Planteamiento del Problema

En la actualidad se tiene gran cantidad de empresas dedicadas al desarrollo de software a nivel nacional, el principal problema que estas presentan es que no poseen la cultura de aplicación de metodologías de ingeniería de software desde el inicio del ciclo de desarrollo, generando problemas posteriores que recaen en la etapa de pruebas, al verificar que los requerimientos no han sido levantados de forma eficiente ni completa y que por tanto el producto no cumple con los objetivos del cliente. Adicionalmente, la etapa de pruebas no tiene definido un proceso formal que cuente con el grado de especialidad que se necesita, sino que más bien se adopta pruebas con los mismos desarrolladores y analistas de negocio que se han visto involucrados en la formulación de requerimientos e implementación de los mismos, ocasionándose incluso que las pruebas no tengan el nivel de detalle necesario, dejándose de lado aspectos que subjetivamente sean considerados como irrelevantes.

Gran parte de los errores e inconsistencias en el software se evidencian en la etapa de implementación, lo cual hace que el producto final tome mayor tiempo y cueste más e incluso que se refleje en la calidad del mismo, generando insatisfacción y desconfianza en el cliente.

SYSGENSA al ser una empresa de desarrollo de software, presenta

problemas identificados en todo el sector, en cuanto a la falta de aplicación de una metodología pese a tenerla definida; los principales problemas recaen en levantamiento de requerimientos incompletos, no se siguen los procedimientos previamente establecidos, no se realizan las pruebas formalmente. Todo esto se torna más complejo aún, cuando se trata de proyectos que tienen que ver con el sector público. Considerando, que en el sector público se maneja una perspectiva diferente del sector privado, en cuanto seguir procedimientos previamente establecidos que muchas veces se presentan como trabas en el avance de los mismos, Adicionalmente, se evidencia falta de compromiso en cuanto al establecimiento de las necesidades reales para plasmar en una aplicación de software,

Este problema ha sido históricamente común en empresas de desarrollo a nivel mundial. Tal como lo reporta un autor importante como Roger Pressman, (Software Engineering, a Practitioner Approach, Library of Congress, 1992), la falta de disciplina metodológica es la causa de incumplimiento de cronogramas y presupuestos de manera dramática. Un día que no se dedique al análisis puede requerir hasta mil días para corregir errores en tiempo de producción. Un dólar que no se invierta en el análisis puede significar cien dólares de gasto para corrección y reacción a cambios de requerimientos en producción, tal como lo muestran las Figuras 10 y 11.

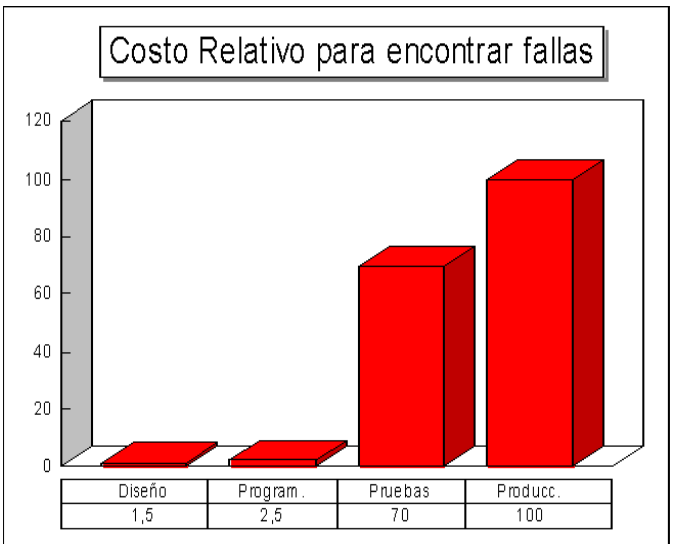

Figura 10. Relación Costo vs Fallas de Sistemas de Software

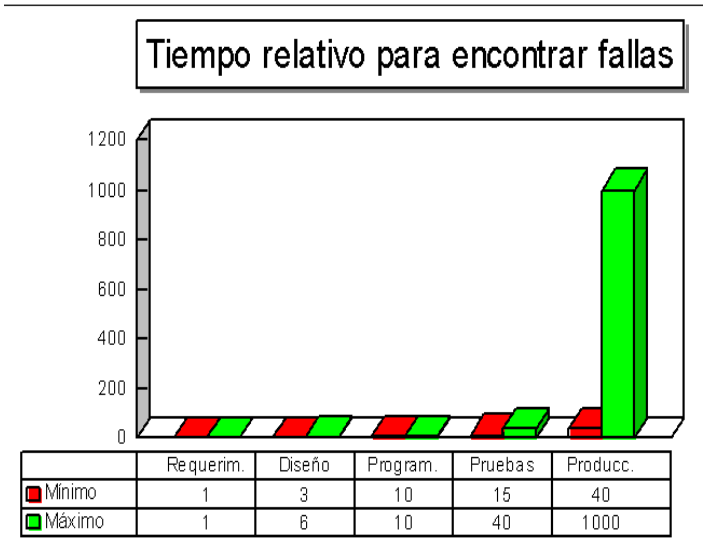

Figura 11. Tiempo Relativo para encontrar fallas

Estas premisas son validadas con datos tomados de ñhttp://www.agilemodeling.com/essays/costOfChange.htmò La Figura 12 presenta el costo de manejar cambios en el año 2002. El ritmo de incremento del costo es exponencial:

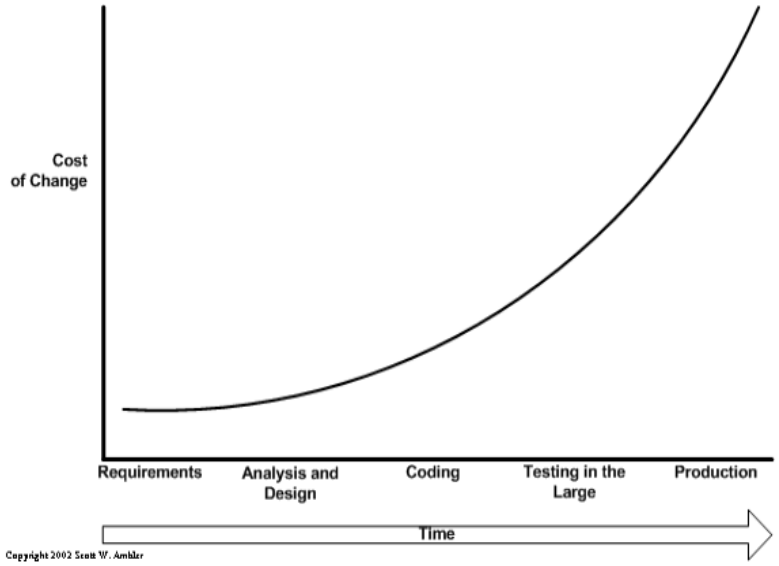

Figura 12. Costo de Manejo de Cambios

Para el año 2006, los mismos autores presentan el problema contrastado mediante diversos enfoques según muestra la Figura 13.

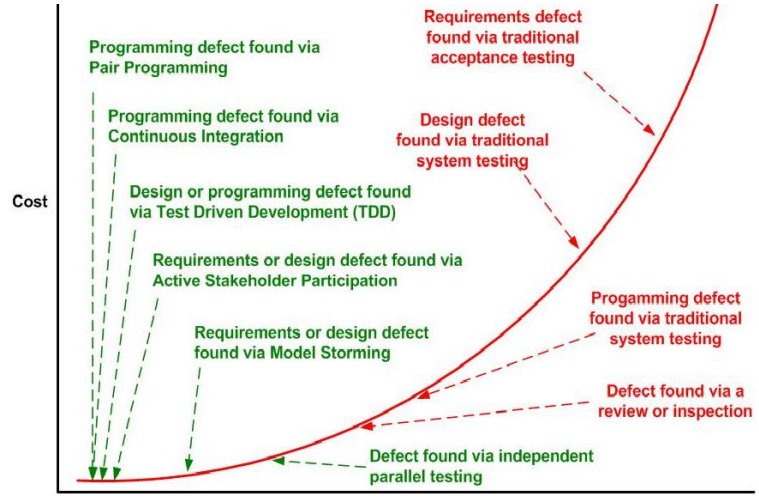

Figura 13. Duración del lazo de realimentación

Según el artículo ñMeasuring the Impact of Changing Requirements on Software Project Cost: An Empirical Investigationó[17], el problema de manejar cambios de requerimientos persiste a la fecha, tal como lo muestra la Figura

[17] Bushra Sharif 1, Dr. Shoab A. Khan, Muhammad Wasim Bhatti del Department of Computer Engineering, College of Electrical & Mechanical Engineering, National University of Sciences &Technology (NUST), y Engineering Management Department CASE, Centre for Advanced Studies in Engineering, Islamabad, Pakistan, publicado en el IJCSI International Journal of Computer Science Issues, Vol. 9, Issue 3, No 1, May 2012, en www.IJCSI.org

14.

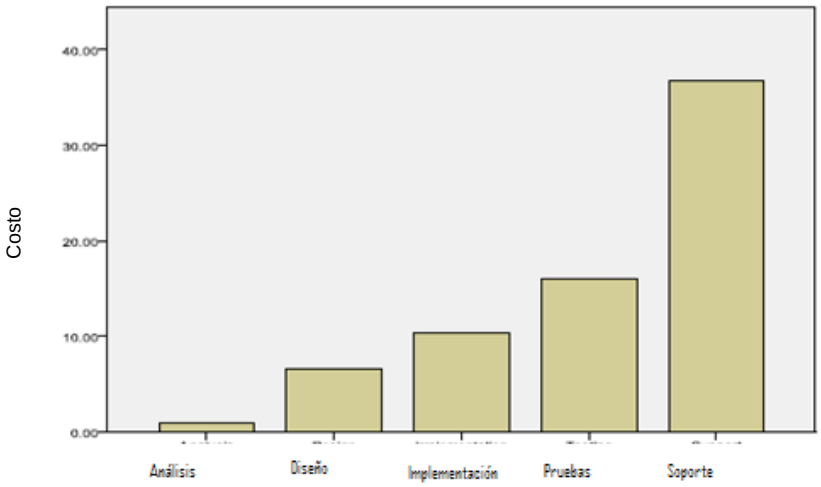

Figura 14. Fases y costo asociado a Cambios

Con este antecedente se han identificado problemas en todas las disciplinas que se aplican en los proyectos, así:

- Definición incompleta de requerimientos en los casos de uso
- Las revisiones de casos con los desarrolladores no garantizan que el equipo de comprenda el proceso a diseñar e implementar
- no se cumplen planes de pruebas
- gran número de novedades (defectos y fallos)
- Tiempos de corrección de novedades altos
- Gestión de cambios insuficiente

3.6.1. Análisis de información gestión de cambios

La empresa utiliza una herramienta para seguimiento de novedades conocida como Mantis, cuyos datos son útiles para cuantificar los defectos y tiempos de solución y pruebas.

La ejecución de la fase de pruebas se realiza de forma manual, no se apoya de ningún software para automatizarlas.

Se realizó el seguimiento de las novedades de dos sistemas que forman parte de un solo proyecto a cargo de la empresa. Los datos de las novedades se tomaron en una fase inicial de análisis por un período de 4 semanas, con información generada de las pruebas de 6 casos de uso del primer sistema (Tesorería) situado en la etapa de pruebas, y el segundo sistema (Administrativo-Becas) con un total 13 casos de uso, de los cuales 7 completada la fase de implementación y listos para paso a pruebas y los restantes 6 completada la etapa de levantamiento de requerimientos.

La Figuras 15, muestra las novedades reportadas por semana durante la fase de pruebas de un universo de 13 casos de uso de sistema que fueron entregados al equipo de desarrollo.

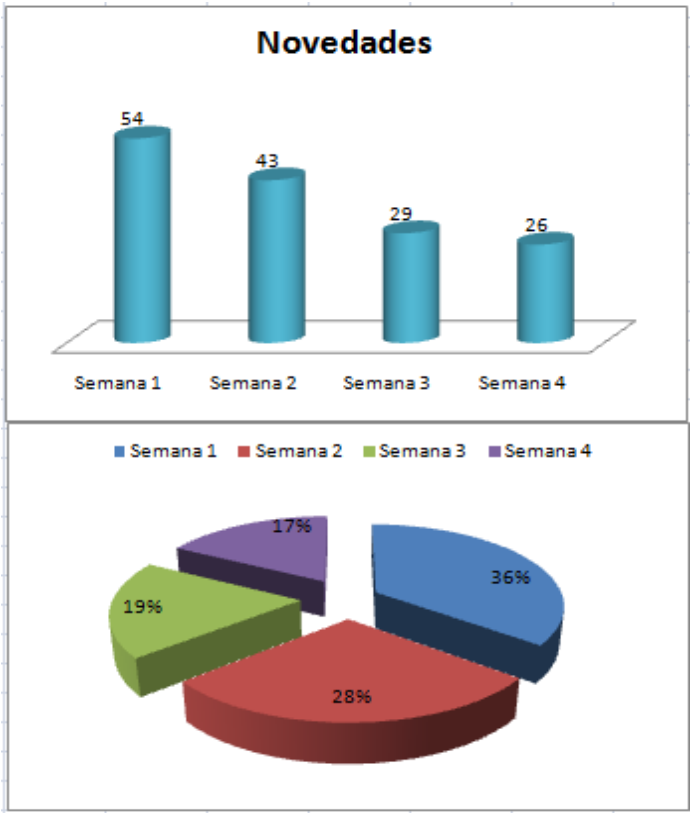

Figura 15. Novedades reportadas por período en etapa de pruebas

La figura 16, muestra una categorización de las novedades analizadas, identificándose las debilidades en las diferentes etapas del ciclo de desarrollo del software, estas novedades facilitaron la identificación del origen de las mismas a fin proponer mecanismos que apoyen a minimizar la generación de estas.

Entre las categorías de novedades reportadas se tienen: Ajustes de funcionalidad, De diseño, De programación, Implementación incompleta, De forma y falta de aplicación de estándares y De Compilación/versionamiento, las mismas que corresponde de manera global a las etapas de Gestión de requerimientos, Análisis y diseño, implementación y, despliegue y gestión y release. Estas novedades tienen un porcentaje promedio de afectación en los reportes de pruebas del 5, 4.3, 83,2, 4.4 y 7.5 %, respectivamente.

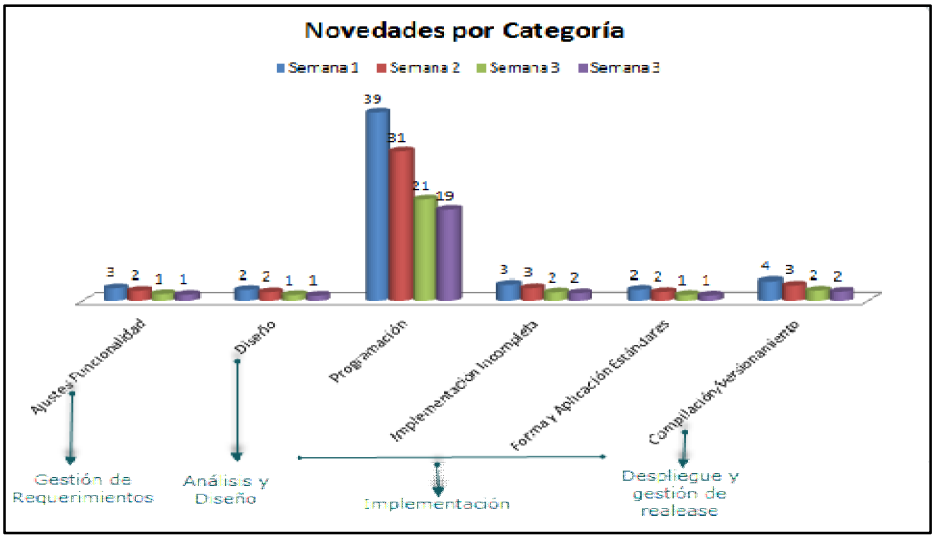

Figura 16. Novedades reportadas por Categoría

La Figura 17, muestra la atención que se dio a los registros de novedades encontradas durante el período. Se entiende como atención; la solución a estos, sus pruebas y respectivo cierre de las mismas.

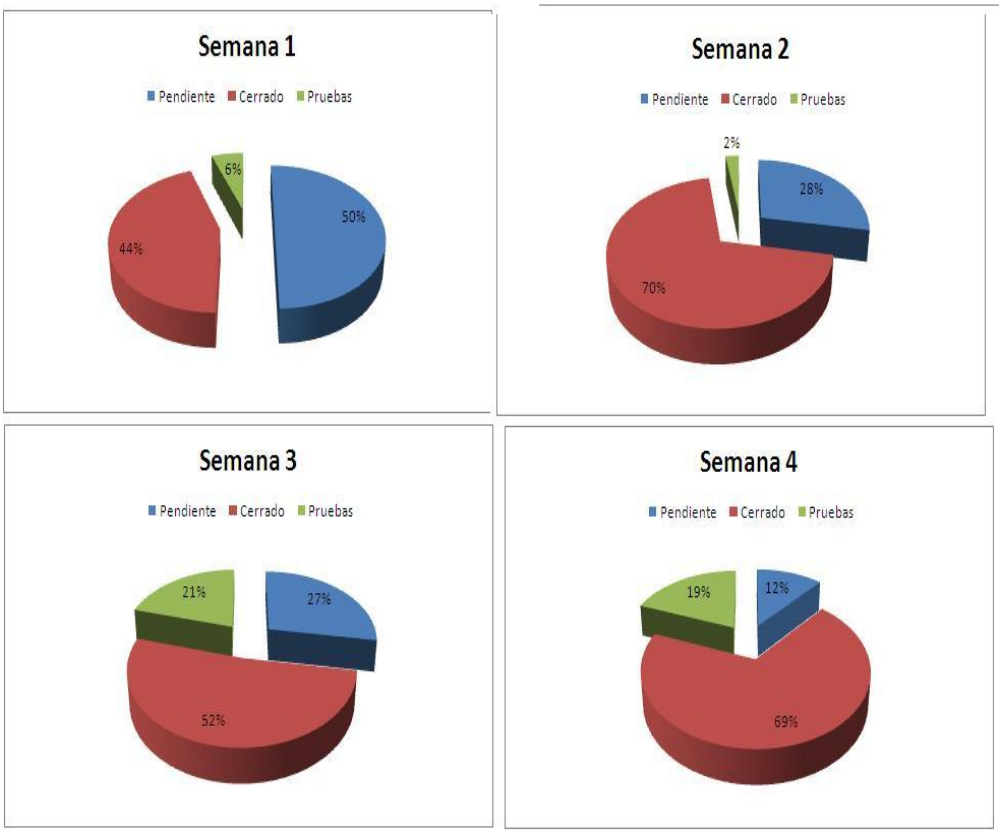

Figura 17. Atención de Novedades

La Figura 18, muestra el grado de complejidad de las novedades por cada semana.

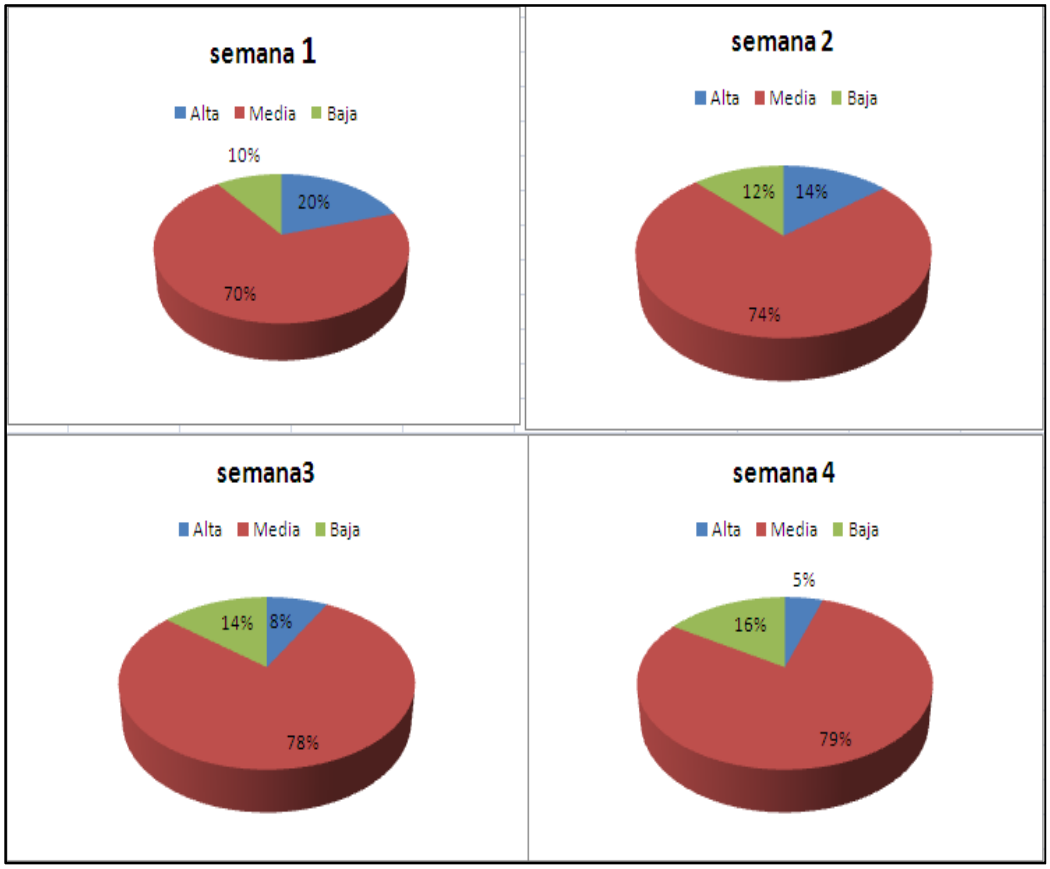

Figura 18. Complejidad de Novedades

La información mostrada, refleja los problemas que se presentan en todas las fases del ciclo de vida de desarrollo del software, la mayoría de novedades reportadas corresponden a errores de programación cuya complejidad es media; sin embargo, existen novedades relacionadas con cambios en la funcionalidad, debidos a que no se contemplaron en los requerimientos originales. Además, se identifica que las pruebas unitarias son incompletas y que muchos de los errores de complejidad baja pueden ser detectados por el mismo desarrollador en la etapa de implementación.

También se presentan errores de despliegue, a nivel de compilación de las versiones que se pasa a pruebas.

CAPITULO IV
PROPUESTA DE HERRAMIENTAS DE APOYO

4.1. Propuesta de Mejora del Proceso de Desarrollo

La propuesta de mejora para SYSGENSA, no solo debe contemplar a la etapa de pruebas, sino que se propone realizar ajustes y cambios en cada una de las etapas del ciclo de desarrollo; tomando en cuenta que los errores identificados no solo se generan en la implementación de las aplicaciones, sino que también están a nivel de falencias en la etapa de análisis de requerimientos, despliegue y gestión de la configuración y versiones. En cada fase, es necesario establecer controles para garantizar que las salidas del proceso sean productos de calidad. Esto asegurará que todo el ciclo de desarrollo del software este bajo el cumplimiento de la metodología establecida.

La utilización de la metodología RUP que actualmente se lleva en la empresa, debe ser fortalecida y respetada por todos los miembros del equipo de trabajo. Adicionalmente, se debe contemplar una reorganización de los roles necesarios que debe tener todo proyecto, así como sus respectivas funciones; se debe considerar políticas de capacitación permanente para el equipo de trabajo, de manera que se logre mayor compromiso con la empresa y a su vez se incentiven para hacer su trabajo de manera efectiva, satisfaciendo sus expectativas personales y profesionales.

Luego del análisis realizado y habiéndose determinado los aspectos potenciales de mejora, se proponen las siguientes modificaciones en el proceso:

- En la conceptualización del negocio debe establecerse reuniones formales entre los responsables tanto de parte del Cliente como de parte

de la Empresa, registrándose las debidas actas de reunión con la agenda desarrollada y los acuerdos alcanzados, además, la Empresa debe formalizar la entrega de artefactos (documentos) para su debida revisión y aprobación. Todo documento debe estar debidamente aceptado por tanto firmado por el o los responsables respectivos.

- Establecer el equipo de aseguramiento y control de calidad de la Empresa, con el objetivo de que participen transversalmente, es decir, en todas las etapas del proyecto, revisando, controlando y validando que se cumplan los procedimientos y políticas que sean definidos, para esto es necesario el establecimiento de un Plan de Aseguramiento y Control de Calidad.
- Incluir revisiones técnicas formales entre analistas conceptuales y analistas/diseñadores/desarrolladores de sistemas para la revisión de los casos de uso generados, con el objetivo de garantizar que sean adecuadamente comprendidos.
- Mejorar el esquema de comunicación a nivel de la Empresa; considerando aumentar la frecuencia de reuniones de trabajo y avance de los proyectos entre los involucrados en los mismos.
- Establecer como política básica previo el inicio de la construcción de los componentes de la aplicación de software; la revisión de casos de uso por parte de los desarrolladores.
- Ampliar la cobertura de las pruebas, iniciando obligatoriamente en las pruebas unitarias que deben ser solventadas y superadas por los implementadores de la aplicación. Se deben crear los planes de pruebas y casos de pruebas paralelamente con la definición de los

requerimientos o casos de uso.

- El despliegue de las aplicaciones de software que la empresa construya deben tener el plan de despliegue debidamente documentado.
- Los cambios que sufra cualquier elemento ó artefacto de la aplicación de software debe estar debidamente justificado y documentado, y por tanto debe ser parte de una versión ó release del proyecto.

En la Figura 19, se muestra el proceso de desarrollo de software propuesto para SYSGENSA.

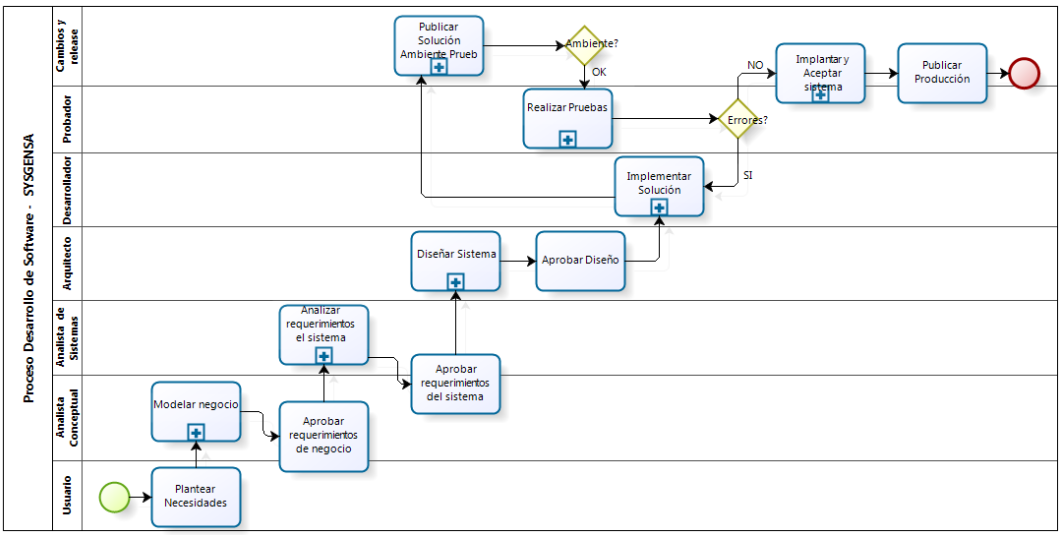

Figura 19. Proceso de Desarrollo de Software propuesto

4.1.1. Modelado de Negocio

La disciplina de modelado de negocio o también llamado modelado empresarial, es una disciplina en la que se debe definir los límites y alcances del negocio que será analizado y cubierto por la aplicación de software.

La propuesta para esta disciplina consiste en establecer reuniones de trabajo con los involucrados y expertos o conocedores del negocio por parte del Cliente y el especialista de negocio y/o analista conceptual de la Empresa. Los

acuerdos de estas reuniones deben estar plasmados en actas de reunión; adicionalmente, se debe generar mandatoriamente los artefactos básicos (documentos) que deben ser debidamente revisados y aprobados por los responsables por parte del Cliente.

Además, se propone la aplicación de listas de chequeo generales para la revisión y control de los artefactos generados, estas listas de chequeo deben ser realizadas por el equipo de aseguramiento y control de calidad de la empresa.

4.1.2. Gestión de Requerimientos

En la disciplina de gestión de requerimientos para la empresa, la propuesta es la de establecer la obligatoriedad del proceso formal y respetando la metodología adoptada. Se debe tener presente que para tener éxito en la construcción del software es básico que la especificación de requerimientos esté completa, para lo cual los analistas conceptuales y de sistemas deben ser seleccionados considerando habilidades y destrezas de comprensión, interpretación y comunicación, de tal forma; que la tarea de elaboración de casos de uso sea fluida, fácil y se evite la redundancia. Adicionalmente, es necesario que se profundice en cómo elaborar casos de uso, especialmente a los miembros del equipo que no han trabajado sobre estos artefactos.

Es necesario que en las plantillas de casos de uso se incluya una sección para indicar y mantener la trazabilidad de estos, lo cual permitirá medir impactos posteriores cuando se soliciten cambios o ajustes a los casos de uso.

Considerando que el análisis y especificación de requerimientos aparentemente carece de complejidad, y que sin embargo conlleva muchos

factores que deben considerarse para formular bien las necesidades del cliente y a su vez transmitirlas al equipo encargado del análisis e implementación del software. Se debe contemplar reuniones debidamente respaldadas con actas tanto a nivel de clientes, así como con el resto del equipo de desarrollo (análisis, diseño, implementación y pruebas) para revisiones de los casos de uso que se vayan a implementar, de tal forma que en caso de que no sean claros o sea necesario ampliarlos, se realice esta tarea previamente al iniciar el diseño e implementación.

4.1.3. Análisis y Diseño

En esta disciplina, se propone generar la documentación de análisis y diseño de la aplicación de acuerdo a plantillas que se muestran en el anexo. Los arquitectos de datos y aplicación deben trabajar conjuntamente para establecer la arquitectura candidata en función de los requerimientos críticos del negocio que deban ser tomados como ñcolumna vertebralò de la aplicación de software.

Es necesario, establecer mecanismos de comunicación fluida para dar a conocer y mantener informado al resto del equipo de trabajo, sobre las políticas para análisis y diseño que se aplicarán para la construcción del producto y los detalles que deben considerar los desarrolladores, de tal forma que el producto cumpla con lo estipulado durante todo el desarrollo de la aplicación.

Además, se debe garantizar que los responsables del diseño de la aplicación revisen y entiendan lo que se solicita en los casos de uso, de tal forma que se genere el respectivo artefacto de diseño para cada caso.

El equipo de aseguramiento y control de calidad debe realizar auditorías periódicas de cumplimiento de estos estándares.

La Figura 20, muestra el proceso de revisión de casos de uso entre analistas conceptuales de negocio, analistas de sistemas, desarrolladores y probadores.

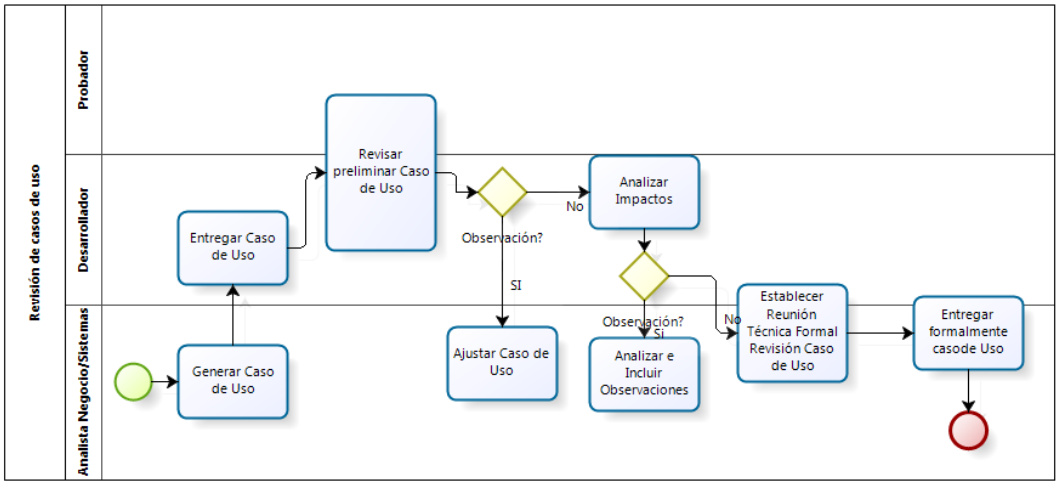

Figura 20. Propuesta del proceso de revisión de casos de uso entre el equipo conceptual, desarrollador y probador.

4.1.4.Implementación

En esta disciplina, se plantea la necesidad de que los desarrolladores revisen y entiendan los casos de uso, así como la documentación de diseño que debe estar previamente realizada, antes de iniciar la programación de la aplicación. Para verificar el cumplimiento de lo propuesto se debe establecer auditorías de revisión de casos de uso y documentos de diseño.

Además, es necesario realizar reuniones complementarias entre los analistas de sistemas que elaboran los casos de uso y el equipo de desarrolladores, para clarificar dudas que incluso pudieren exigir ajustes a los casos de uso entregados al equipo de desarrolladores.

Los desarrolladores, deben ser supervisados y guiados por los arquitectos y/o líderes desarrolladores, para que cumplan con los lineamientos que rigen a cada proyecto.

El equipo de control de calidad deberá revisar que los artefactos (código

y documentación técnica) generados cumplan con los requerimientos de acuerdo a los estándares que se sigan en cada proyecto.

4.1.5.Pruebas

Considerando que la empresa, en esta disciplina presenta la mayor cantidad de problemas, es necesario definir los tipos de pruebas que se deben incluir en cada proyecto. El equipo de control de calidad debe trabajar durante todo el proceso previo a las pruebas y poner mayor énfasis en las pruebas unitarias, mismas que deben realizarla los desarrolladores para garantizar que el producto que cada uno es responsable funcione correctamente, de tal forma que cuando se integre no se tengan problemas críticos que impliquen correcciones que retrasen los tiempos y por tanto la planificación de cada proyecto.

Las actividades a realizar se describen en la Tabla 5 conjuntamente con su cobertura o frecuencia:

Actividad	Cobertura o Frecuencia	Descripción
Verificación de cumplimiento de estándares y metodología de trabajo	Métodos públicos y privados. Documentación de modelado de negocio	Revisiones en base a los estándares de programación definidos para el Proyecto. Revisiones de documentación en base a la metodología propuesta
Pruebas Cruzadas	Todos los cambios	Cualquier cambio debe ser realizado en base al proceso de control de cambios definido para el Proyecto y será

		evaluado por otro desarrollador antes de ser enviado a Pruebas. Este procedimiento de pruebas se utilizará cuando ya se haya liberado la versión para garantizar la estabilidad de la aplicación.
Pruebas unitarias	Programa Desarrollado para cada caso de uso.	Se debe realizar una prueba funcional por cada caso de uso, verificando el cumplimiento del negocio. Se debe considerar las condiciones límites para cada argumento y se probarán ambos extremos para cada límite. Las pruebas deben de ser ejecutadas y superadas antes de que cada paquete de versión o release se compile y sea confirmado para pruebas formales por el equipo de pruebas
Pruebas Funcionales	Campos, pantallas y casos de uso	Equipo de Aseguramiento de Calidad debe generar y mantener un Protocolo de Pruebas para todo el sistema. Este plan será suficientemente detallado para que una persona pueda realizar repetidamente las pruebas.
Pruebas de	Se probará	Se validará el rendimiento del sistema

Estrés	tiempos de respuesta al ejecutar los programas concurrentemente	utilizando herramientas para monitoreo del Performance que utilizan scripts que lanzan a la vez llamados a funciones como si se tratase de más de 1000 usuarios, esto se ejecutará como parte de la fase de transición del proyecto
Pruebas de Integración	Al finalizar cada iteración	Por cada Iteración se debe ejecutar los planes de pruebas, relacionadas a los casos de uso desarrollados hasta la fecha, para garantizar que no se presenten errores que aparentemente ya se encontraban solucionados.
Pruebas de Aceptación	Pruebas con Usuarios Finales	Se involucrará a los delegados funcionales por parte del Cliente. Para el seguimiento y control el líder funcional filtrará los problemas encontrados.
Instrumentación y monitoreo en ambiente de producción	Supervisión de servidores de aplicación, presentación y Base de Datos	Se debe supervisar el desempeño de los servidores para detectar de forma automática, interrupciones en el servicio o degradación del rendimiento.
Reportes de fallas de campo durante la etapa	Reporte de fallas encontradas	Se debe solicitar a los usuarios que re-porten las fallas encontradas al líder funcional, quien filtrará los problemas y

Acompañamiento (Mantenimiento)		registrará en la herramienta de registro de fallas.
Documentación	Revisión de entregables por fase	Verificación de entregables del proyecto

Tabla 5. Actividades propuestas para la disciplina de pruebas

4.1.6. Despliegue

Esta disciplina se preocupa de la distribución, entrega e instalación de las partes que constituyen el sistema físico, donde se identificaron los siguientes problemas:

Aplicación de paquetes compilados incompletos en ambiente de producción, aplicación no operable después de liberaciones a producción, desorganización para armar paquetes de liberaciones, falta de ambientes de preproducción. Para lo cual se propone generar el plan de despliegue que respalde y justifique los componentes que deben tomarse en cuenta para la instalación del producto, además, de la creación de ambientes de preproducción previo liberaciones; de tal forma que se pueda probar la operatividad básica de la aplicación después de la aplicación de una liberación.

4.1.7. Gestión de la Configuración y Cambios

La gestión de cambios constituye una parte importante en proyectos de software, por lo que es necesario mantener el control, rastreo y monitoreo de los mismos de tal manera que se garantice el concepto de desarrollo iterativo. Además, es también una guía en la definición de los ambientes de trabajo de

cada desarrollador, controla los cambios de todos los elementos de software, tales como: modelos, código, documentos. Define la administración de releases de acuerdo a las iteraciones y fases que se requieran para alcanzar el software final.

Como parte de la gestión de la configuración se propone contemplar el establecimiento de líneas bases y generación de versiones y releases, por tanto esta disciplina al igual las otras, van estrechamente ligadas con la gestión del proyecto. Tomando en cuenta que en la empresa se ha identificado errores provocados por los cambios, confusión y mezcla de versiones, compilación y publicación de componentes; es necesario establecer actividades y controles para minimizarlos, así:

- Asociar los componentes de un cambio mediante una sola etiqueta (código) generada en la herramienta de gestión de cambios e incidentes al registrar una novedad o caso de uso.
- Elaborar documentos de solicitud de cambios por parte de los desarrolladores, indicando los componentes involucrados y detalles de configuración y compilación.
- Monitorear y verificar que las versiones y releases mantenidos contengan los cambios o requerimientos según la planificación de las iteraciones entregadas por la gestión del proyecto.
- Verificar que los cambios y/o requerimientos tengan la respectiva confirmación de pruebas y petición de empaquetamiento para despliegue.
- En caso de cambios simultáneos sobre un mismo componente de la aplicación de software, se debe contemplar la unificación del código con

los responsables de los cambios al componente.

- Establecer un número máximo de cambios pendientes de confirmación de pruebas.
- Establecer líneas bases para todos los artefactos que se mantengan en el proyecto.
- Centralizar organizadamente todos los artefactos que se generen en el proyecto con las respectivas versiones.
- Establecer políticas de inclusión de descripciones claras en todo cambio a cualquier artefacto del proyecto.

4.1.8. Gestión de Proyecto

Considerando que es en esta disciplina donde se determina lo siguiente:

- La visión general del proyecto en la cual se proporciona una descripción del propósito, alcance y objetivos del proyecto, estableciendo los artefactos que serán producidos y utilizados durante el mismo.
- La organización del proyecto, que describe la estructura organizacional del equipo de desarrollo.
- La gestión del proceso, que explica los costos y planificación estimada, define las fases e hitos del proyecto y describe cómo se realizará su seguimiento.
- Y, los planes y guías de aplicación, que proporcionan una vista global del proceso de desarrollo de software, incluyendo planes, métodos, herramientas y técnicas que serán utilizadas.

Se plantea la necesidad de establecer tiempos reales y reorganizar los equipos de trabajo considerando las fortalezas y debilidades de cada uno de los miembros de los mismos, de tal forma que esto permita cubrir con las

expectativas de los clientes en el logro del producto o aplicación se software requerido. Adicionalmente, se debe empoderar al recurso humano facilitando capacitación en los aspectos que se identifiquen necesarios y débiles a nivel de empresa; y de ser necesario, considerar la alternativa de contratar personal para cubrir temporalmente las falencias.

4.1.9. Entorno de Desarrollo

La disciplina de entorno de desarrollo, es la encargada de proveer la documentación básica para todos los equipos de trabajo del proyecto, además, de garantizar que los ambientes estén operativos durante todas las etapas de consecución del proyecto.

En este sentido, se propone establecer controles para verificar que los diferentes roles generen la documentación básica para que el equipo del proyecto tenga los insumos para avanzar con sus actividades.

La empresa debe generar políticas de comunicación semanal, los líderes de proyectos deben encargarse de que sus equipos de trabajo tengan conocimiento del ñEntornoò para desenvolvimiento del mismo.

Se debe alimentar el portal/repositorio común con toda la documentación que se vaya generando en los proyectos, manteniendo una organización amigable y de fácil ubicación, así como el versionamiento respectivo.

4.1.10. Control de Calidad

El control de Calidad, es una actividad transversal y de apoyo en todas las disciplinas y fases de los proyectos, en esta se debe considerar mecanismos que permitan asegurar la calidad del producto final. Se deben establecer métricas para cuantificar la calidad del producto al final de cada disciplina y fase, así como también, auditorías de calidad con mayor frecuencia.

Para alcanzar los objetivos de calidad del proyecto se proponen las siguientes estrategias de control y monitoreo:

- Revisiones e inspecciones conjuntas sobre los procedimientos y productos acorde a la metodología de trabajo.
- Utilización de planes de pruebas, los cuales deberán contener: guías de pruebas, enfoque de las pruebas a realizar y el cronograma por tipo de prueba.
- Utilización de planes de análisis y diseño de las aplicaciones
- Utilización de estándares de programación
- Auditorías de cumplimiento de procedimientos al equipo de la empresa
- Auditorias de generación de artefactos a cada rol participante en los proyectos de la empresa.
- Auditorías de los usuarios finales.
- Análisis de los resultados obtenidos de la aplicación de las métricas por parte del Cliente.
- Reuniones para determinar las soluciones a los riesgos encontrados.

Los instrumentos planteados en este capítulo para cada una de las disciplinas y aplicables a todas las fases del ciclo de desarrollo seguido por la empresa, generaran un impacto positivo; mismo que se verá reflejado en la etapa de pruebas de la solución de software que se construya. Para este análisis se tomó en cuenta las categorías de novedades presentadas en el capítulo III aplicadas a 13 de los 19 casos de uso objeto de la muestra para el análisis del proceso.

En la Figura 21, se indican las novedades que se hubieran generado sin la aplicación de las propuestas, mismas que se inducen de acuerdo a los factores obtenidos en función del análisis de los 13 casos de uso de la primera muestra. En la Figura 22, se presentan los resultados obtenidos del seguimiento de pruebas de los 6 requerimientos que estaban en la etapa de gestión de requerimientos al inicio y sobre los cuales se aplicaron algunas de las propuestas.

Dentro de las propuestas aplicadas sobre los 6 casos de uso restantes están: la de aplicación de listas de chequeo generales para revisión y control de artefactos y generación de actas de reunión en las disciplinas de: Modelado de negocio, Gestión de requerimientos, Implementación, Pruebas y Gestión de configuración/release y Gestión de proyectos, ajuste a plantilla de casos de uso para mantener registro de trazabilidad de requerimientos, Reuniones técnicas formales entre los equipos de análisis-diseño-implementación-pruebas; para la revisión de los casos de uso antes de iniciar con las tares que compete a cada uno, generación de planes de prueba globales y casos de pruebas, generación de solicitudes de cambio, la aplicación de pruebas unitarias y revisión de código a nivel de programadores, para lo cual se realizaron auditorías sobre el código implementado por parte de los líderes y revisiones de pares (entre programadores). Adicionalmente, como parte de la gestión del proyecto y entorno se reconfiguró el repositorio común para centralizar los artefactos generados y se reviso el esquema de versionamiento del proyecto a fin de trabajar con los líderes del proyecto y de los diferentes equipos sobre los impactos de cambios y aplicación del versionamiento a todos los artefactos incluyendo el código fuente y builds.

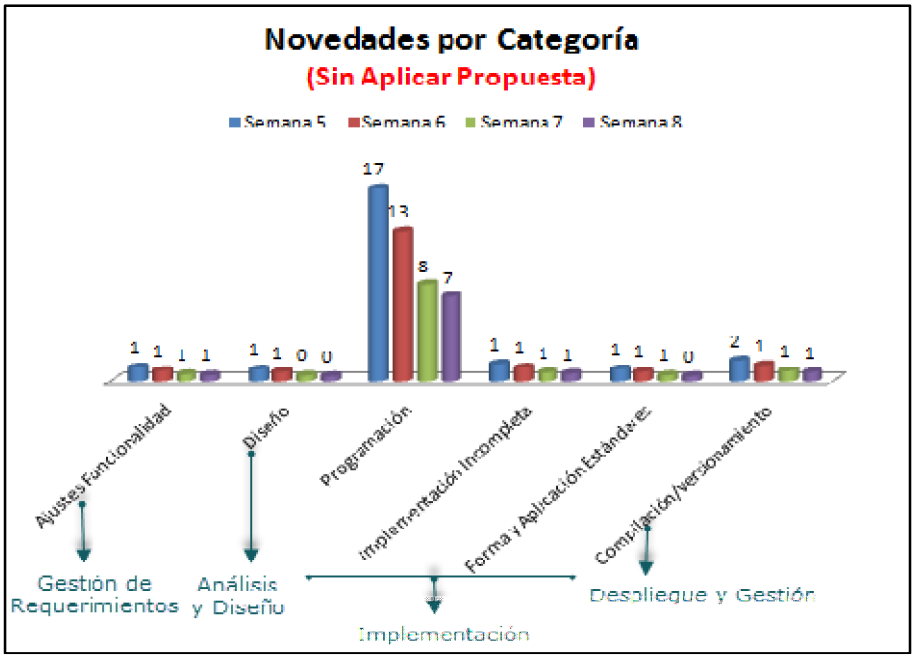

Figura 21. Novedades por categoría sin aplicación de propuesta

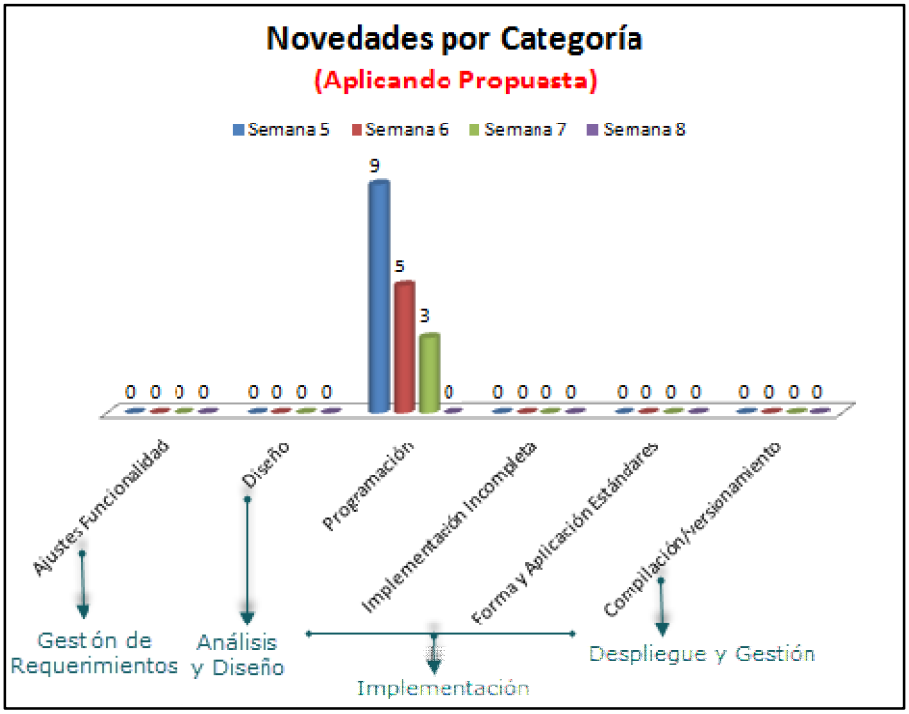

Figura 22. Novedades por categoría con aplicación de propuesta

Cabe recalcar, que el efecto posterior a la mejora del proceso aplicando la propuesta es positivo; sobre todo para la disciplina de implementación que es la que mayor cantidad de novedades generaba, es así que se mejora en un 43,05% en promedio, reduciendo los tiempos en la etapa de pruebas y por tanto solución a las novedades, aunque esto implique incrementar el tiempo de dedicación del responsable del control y auditorías, y del resto del equipo para la generación y cumplimiento de los artefactos y procedimientos establecidos en cada una de la disciplinas y fases de ciclo de desarrollo de SYSGENSA.

4.2. Entregables del proyecto

En la Tabla 6, se indican y describen los artefactos que deben ser ajustados y adoptados dentro de los proyectos y que forman parte de los entregables.

Nombre	Descripción	Observación
Plan de gestión de requerimientos	Describe los artefactos relacionados con los requerimientos funcionales, tipos de requerimientos y sus respectivos atributos, especificando la información que deben ser recolectada y los mecanismos de control a ser usados para la medición, evaluación y controlar los cambios a los requerimientos del producto. Este plan además, contendrá los diagramas de actividades y flujos a seguir para atender los requerimientos de	Este artefacto debe ser adoptado por la Empresa, en el Anexo 1 se muestra el formato con la información que debe contener.

	software.	
Casos de Uso	Un caso de uso es una técnica de modelado usada para describir lo que debería hacer un sistema (negocio/ software) nuevo o lo que hace un sistema que ya existe. Se describen bajo la forma de acciones y reacciones el comportamiento de un sistema desde el punto de vista de un usuario, permiten definir los límites del sistema y las relaciones entre el sistema y el entorno.	Este artefacto debe ser ajustado, se debe incluir una sección para gestión de requerimientos, en el Anexo 2 se muestra el formato con la información ajustada.
Plan de Aseguramiento de la Calidad	El contenido del plan de aseguramiento de calidad debe incluir aspectos tales como: - Estándares y normas a aplicar durante el proceso de desarrollo. - Procedimientos para informar, hacer seguimiento y resolver errores, identificando los responsables de llevarlos a cabo. - Mecanismos de modificación de productos estableciendo cómo se van a gestionar dichas modificaciones y el modo de comunicarlo a los implicados. - Métricas a utilizar, un resumen sobre los	Este artefacto debe ser adoptado por la Empresa, en el Anexo 3 se muestra el formato con la información que debe contener.

	resultados que se esperan obtener de estas métricas, incluyendo una descripción sobre cómo se utilizarán para poder controlar el avance del proyecto.	
Plan de Pruebas	La definición de las metas y objetivos de prueba dentro del alcance de cada iteración (fase o proyecto), los elementos identificados, la estrategia adoptada, los recursos requeridos y los entregables a ser producidos.	Este artefacto debe ser adoptado por la Empresa, en el Anexo 4 se muestra el formato con la información que debe contener.
Caso de Prueba	Cada prueba es especificada mediante un documento que establece las condiciones de ejecución, las entradas de la prueba y los resultados esperados. Estos casos de prueba son aplicados como pruebas de regresión en cada iteración. Cada caso de prueba llevará asociado un procedimiento de prueba con las instrucciones para realizar la prueba.	Este artefacto debe ser adoptado por la Empresa, en el Anexo 5 se muestra el formato con la información que debe contener.
Solicitud de Cambio	Los cambios propuestos para los artefactos se formalizan mediante este documento. Asimismo, se hace un seguimiento de los defectos detectados, solicitud de mejoras o cambios en los requisitos del producto. Así	Este artefacto debe ser adoptado por la Empresa, en el Anexo 6 se muestra el formato

	se provee un registro de decisiones de cambios, de su evaluación e impacto y se asegura que éstos sean conocidos por el equipo de desarrollo. Los cambios se establecen respecto de la última línea de base (el estado del conjunto de los artefactos en un momento determinado del proyecto) establecida.	con la información que debe contener.
Plan de Despliegue	El plan de despliegue describe el conjunto de tareas necesarias para instalar y probar el producto desarrollado de forma tal que pueda efectivamente ser desplegado a la comunidad de usuarios que harán uso del sistema.	Este artefacto debe ser adoptado por la Empresa, en el Anexo 7 se muestra el formato con la información que debe contener.

Tabla 6 Artefactos que deben ajustarse ó adoptarse en los proyectos de SYSGENSA.

Adicionalmente, en el Anexo 8 se resumen los artefactos que SYSGENSA genera actualmente, esto visto desde la perspectiva de artefactos planteada por la metodología RUP.

Es preciso destacar que de acuerdo a la filosofía de RUP (y de todo proceso iterativo e incremental), todos los artefactos son objeto de modificaciones a lo largo del proceso de desarrollo, con lo cual, sólo al término

del proceso se podrá tener una versión definitiva y completa de cada uno de ellos. Sin embargo, el resultado de cada fase y los hitos del proyecto están enfocados a conseguir un cierto grado de completitud y estabilidad de los artefactos. Esto deberá indicarse en los planes y objetivos de cada una de las fases e iteraciones propuestas en cada proyecto.

4.3. Herramientas Informáticas de Apoyo

Existen en el mercado herramientas informáticas para facilitar la mayoría de actividades involucradas en el ciclo de desarrollo de software, muchas de estas son gratuitas y otras tienen un costo. En el Anexo 9, se describen algunas herramientas que pueden ser tomadas en cuenta para apoyo en ciertas disciplinas del ciclo de vida del desarrollo de software en SYSGENSA.

CAPITULO V
CONCLUSIONES Y RECOMENDACIONES

5.1. CONCLUSIONES

Una vez desarrollado este trabajo y habiéndose realizado la investigación y el análisis respectivo se llegan a las siguientes conclusiones:

En las estadísticas de errores del proyecto analizado en la empresa, se identifican que la mayoría de novedades reportadas corresponden a errores de programación cuya complejidad es media; sin embargo, existen novedades relacionadas con cambios en la funcionalidad, debidos a que no se contemplaron en los requerimientos originales.

El sector de software en el Ecuador tiene un despunte muy importante en la economía del país, el apoyo que el Estado promulga para el 2013 es fundamental para que las empresas dedicadas a este sector persigan certificaciones de calidad conocidas internacionalmente; con el objetivo de ser más competitivas no solo a nivel interno sino también a nivel internacional.

El análisis de mercados de cualquier sector se puede llevar a cabo cuando existe disponibilidad de información a nivel general y detallada; el sector de software no tiene la suficiente apertura ni el nivel de detalle necesario en la información, es así que para el caso en que las entidades gubernamentales contraten los servicios de un empresa de software no tienen partidas presupuestaria claramente definidas o desglosadas que permitan realizar un seguimiento de la ejecución del presupuesto orientado a este sector.

Las empresas de desarrollo de software independientemente del cliente al que se orienten (privados o públicos), deben seguir una metodología que les permita garantizar el éxito de los proyectos, a lo largo del ciclo de vida del

mismo. Las certificaciones de calidad deben ser la meta a perseguir de cada una de estas empresas, ya que esto les abrirá puertas en el mercado no solo nacional sino internacional.

El uso de herramientas de apoyo en todo proceso de desarrollo de software, facilita la evolución de cada fase de manera efectiva, optimizando el uso del recurso humano y favoreciendo el recurso tiempo.

El talento humano constituye el factor más importante de toda organización, razón por la cual debe estar adecuadamente incentivado, esto permitirá alcanzar un grado de lealtad y compromiso que favorezca alcanzar con éxito todo proyecto que se emprenda. Bajo esta premisa es necesario que se contemplen planes de capacitación periódicos para el equipo de trabajo orientados al crecimiento profesional y personal de cada miembro del mismo.

El paradigma más grande para cualquier cambio; es la cultura de las personas y resistencia al cambio; y para el caso de esta empresa no es la excepción. Esto requiere de un proceso de concientización y aceptación de los problemas y de actitud positiva para enfrentarlos y corregirlos, siguiendo y respetando los procedimientos que se establezcan a todo nivel de la organización, sin justificación de premuras de tiempo y costo.

El ser humano trabaja mejor si se siente a gusto en su lugar de trabajo, si está rodeado de un ambiente de trabajo agradable, si su trabajo se ve, no solo recompensado económicamente, sino también valorado. La motivación no debe ser entendida únicamente como una compensación de índole económica o material, sino que se debe entender que muchas veces el reconocimiento del trabajo bien hecho, una felicitación por parte del cliente, una palabra de ánimo del Jefe cuando se encuentra agobiado y desanimando, genera un efecto

positivo y motivador en el trabajador.

En la Empresa SYSGENSA, se identificó que los errores en las aplicaciones desarrolladas no solo para el sector público sino para el privado radicaban en los problemas a nivel de procesos, sobre todo en la formalización de necesidades y acuerdos realizados con los clientes, lo cual generaba el no tener el debido respaldo ñescritoò para justificación de cambios a nivel de la gestión en general de los proyectos. Otro proceso crítico identificado radicaba en la pruebas del producto, mismo que evidentemente mejoró al elaborar los casos de pruebas paralelamente con los casos de uso con los respectivos planes de prueba, beneficiándose todo no solo el equipo de desarrollo sino también el proyecto en sí, puesto que el equipo de probadores podía cumplir con el objetivo de realizar pruebas bajo un guión y de acuerdo al plan de pruebas. Además, surge la necesidad de fortalecer el área de control de calidad no solo orientándolo hacia las pruebas de los productos, sino a todo el ciclo de desarrollo, para lo cual es necesario generar el plan de aseguramiento de calidad que debe ser difundido a todos los miembros del equipo del proyecto.

El establecimiento de reuniones para revisión de los casos de uso entre los responsables conceptuales, desarrolladores y probadores fue un factor que permitió implementar la funcionalidad deseada con menor cantidad de observaciones durante y después de su construcción, y en la fase de pruebas.

Es importante tomar en cuenta que el talento humano es lo más valioso de una empresa y que todo lo que represente gasto en capacitación e incentivos otorgados, a futuro se convierten en la mejor de las inversiones.

5.2. RECOMENDACIONES

Tomando en cuenta que en general, la cultura de las personas siempre tiende a dejar de hacer lo que le representa mayor inversión de tiempo, para el caso de SYSGENSA, si bien es cierto se adopto una metodología inicialmente claramente definida y que con el tiempo se dejó de seguirla y respetarla, se recomienda, mantener los cambios realizados al proceso de requerimientos y planificar las pruebas, así, como empoderar al equipo, realizar analizar periódicamente las habilidades y destrezas de cada miembro del equipo y fomentar planes de capacitación.

En la actualidad existen gran cantidad de herramientas de software que pueden apoyar la gestión del ciclo de desarrollo de software, se puede optar por algunas de ellas especialmente por aquellas que no tienen costo. La herramienta de seguimiento de incidentes ñMantisò, que mantiene la empresa debe ser explotada y utilizada por todo el equipo.

Se recomienda que las empresas de desarrollo de software en el Ecuador y en particular SYSGENSA consideren la alternativa de una Certificación de Calidad, tomando en cuenta que en la actualidad existe el apoyo para el sector de desarrollo de software por parte del Gobierno, para esto debe mantener y ampliar la cobertura de la metodología adoptada.

BIBLIOGRAFIA

- Evans ,James R. y Lindsay, William M., *Administración y Control de la Calidad*, México, International Thomson Editores S.A., 2005.
- Pande, Peter S., Newman, Robert P. y Cavanagh Roland R., *Las Claves Prácticas de Seis Sigma: Una Guía Dirigida a los Equipos de Mejora de Procesos*, España, McGraw-Hill, 2004.
- Pressman, Roger S, *Ingeniería del software - Un enfoque práctico*, Madrid , Mcgraw Hill, 1995, 2002.
- Senn, James, *Análisis y diseño de sistemas de información*, Mcgraw Hill, 1997.
- Laudon, Kenneth C., Laudon, Jane P., *Sistemas de Información Gerencial*, México, Pearson Prentice Hall, 2008.
- Sommerville , Ian, *Ingeniería del Software*, Madrid, Pearson Prentice Hall, 2005.
- Paulk, M., et al., *Capability Maturity Model for Software*, Software Engineering Institute, Carnegie Mellon University, Pittsburgh PA, 1993.
- IEEE*, Standards Collection: Software Engineering,IEEE* Standard 610.12-1990, 1993.
- Jacobson, I., Booch, G. y Rumbaugh, J. , *The Unified Software Developement Proccess*, Addison-Wesley, 1999.
- Weitzenfeld , Alfredo, *Ingeniería de Software orientada a objetos con UML, Java e Internet*, Thomson, 2005.
- Juran, Gryna y Bingham, *Manual de Control de la Calidad*, Edit.

Reverté S.A., 1990.

- Carot Alonso, Vicente, *Control estadístico de la calidad,* España, Ed. Servicio de Publicaciones , 1998.
- Esponda, Alfredo, *Hacia una calidad más robusta con ISO 9000:2000,* México, Ed. Panorama Editorial, 2005.
- Nava Carbellido, Víctor Manuel, *¿Qué es la calidad?: Conceptos, gurús y modelos fundamentales*, México, Limusa, 2005.
- Alcalde San Miguel, Pablo, *Calidad 2da Edición*, España, Ed. Paraninfo, 2010.
- IBM Corporation , IBM Rational Method Composer (Rational Unified Process Versión 7.0.1), 2005.
- Qualitrain Express, ñAseguramiento de la calidad de Softwareò, en http://www.qualitrain.com.mx/Aseguramiento-de-la-Calidad-de-Software.html
- Escalone, Fernanda, ñEstudio Comparativo de los Modelos y Estándares de Calidad del Softwareò, Universidad Tecnológica Nacional Facultad Regional Buenos Aires, Argentina, junio 2006 en http://laboratorios.fi.uba.ar/lsi/scalone-tesis-maestria-ingenieria-en-calidad.PDF, 2006.
- Harrington, H. James, *Mejoramiento de los Procesos de la Empresa*, Colombia, McGraw-Hill, 1997.
- Kruchten, Philippe, *The Rational Unified Process: An Introduction,* Addison-Wesley, 2000.
- López Guerrero, Hernando, Carrión Gordón, Hugo, *Las Tecnologías de la Información y las Comunicaciones en la Competitividad de*

Quito, Quito, ConQuito Corporación de Promoción Económica, Segunda Edición, 2007.

- Aesoft, ñEstudio de mercado del sector de software y hardware en Ecuadorò, quito, septiembre 2011, en http://www.aesoft.com.ec/www/index.php/noticias/183-estudio-de-mercado-2011.
- European Management Center, ñIndustria Local de Software exporta 30 millones en el 2011ò, noticias y medios, en http://www.eumcecuador.com/noticias_amp.php?id_noticia=14, 23 de diciembre del 2011.
- ProChile, ñEstudio de Mercado Servicio de Desarrollo de Software en Ecuadorò, Documento elaborado por la Oficina Comercial de ProChile en Ecuador en colaboración con la Universidad Casa Grande, Guayaquil, 2012, en http://rc.prochile.gob.cl/sites/rc.prochile.gob.cl/files/documentos/documento_11_19_12112936.pdf
- Prom Peru, ñPerfil del Mercado de Software en el Ecuadorò, 2011, en http://www.siicex.gob.pe/siicex/resources/sectoresproductivos/36471667rad3D22C.pdf.
- McDonald Landázuri, Bárbara A., ñDefinición de Perfiles en Herramientas de Gestión de Requisitosò, Facultad de Informática Universidad Politécnica de Madrid, Madrid: s.n., 2005, en http://www.dlsiis.fi.upm.es/docto_lsiis/Trabajos20042005/Mcdonald.pdf.
- Visure, ñVisure Requeriments 4.4 is now availableò, octubre 2012, en http://www.visuresolutions.com/noticia-completa/-/journal_content/56_INSTANCE_s6NZ/10826/246874?templateId=232653.

- Wiegers, Karl E., ñAutomating Requirements Managementò, en http://www.processimpact.com/articles/rm_tools.html.
- Garzas, Javier, ñUna lista de herramientas de calidad software imprescindiblesò, Marzo 2012, en http://www.javiergarzas.com/ 2012/03/herramientas-de-calidad-software.html.
- Fábricas de software, ñHerramientas de Desarrollo de Software ï Gestión de Requisitosò, en http://www.fabricasdesoftware.es/herra mientas.
- IBM, ñRational RequisitePròo, en http://www-01.ibm.com/software/ awdtools/reqpro/.
- IBM, ñDOORS familyò, en http://www-01.ibm.com/software/awdtools/ doors/ productline/.
- aNimble Platform, "Project Information", en http://nimble.sourceforge. net/.
- Requerimients Engineering Software - Visure Solutions, "Get to Know Visure Requirements Software" en http://www.visuresolutions.com/.
- remasysytem, "Project Information", en http://code.google.com/p/re masystem/.
- SPARX Systems, "Enterprise Architect - Herramienta de diseño UML", en http://www.sparxsystems.com.ar/products/ea.html.
- Fábricas de software, ñHerramientas de Desarrollo de Software ï Gestión de la Configuraciónò, en http://www.fabricasdesoftware.es/ herramientas.
- CVS - Open Source Version Control, "Introduction to CVS", en http://www.non gnu.org/cvs/.

- Aegis 4.24, "Aegis Propaganda", en http://aegis.sourceforge.net/propaganda/index.html.
- BITKEEPER, "Products", http://www.bitkeeper.com/Products.html.
- IBM, "Una Solucion de gestión de configuración de software", en http://www.ibm.com/software/products/es/es/clearcase/.
- Mantis Bug Tracker, "Feature List", en http://www.mantisbt.org/.
- Bazaar, "What is Bazzar?", en http://bazaar.canonical.com/.
- Tratando de entenderlo, "Herramientas de análisis de calidad del código", 2009, en http://tratandodeentenderlo.blogspot.com/2009/10/herramientas-de-analisis-de-calidad-del.html.
- Fábricas de software, "Herramientas de Desarrollo de Software ï Calidad, Verificación y Validación » SONAR", en http://www.fabricas desoftware.es /herramientas/calidad-verificacion-y-validacion/sonar/
- HP Quality Center, "Descripción General", en http://www8.hp.com/es/es/software-solutions/software.html? compURI=1172141.
- Arturo Fernández, "Comparativa de herramientas para pruebas automáticas", en http://www.globetesting.com/2012/03/comparativa-de-herramientas-para-pruebas-automaticas/.
- Fábricas de software, ñHerramientas de Desarrollo de Software ï Gestión de Proyectosò, en http://www.fabricasdesoftware.es/herramientas.
- Trac Integrated SCM & Project Management, "Trac Open Source Project", en http://trac.edgewall.org/.
- Abartia Team, "Gestión de Proyectos con dotProject", en http://www.abartiateam.com/dotproject.

- Atlassian, "Algunas de las maravilla de JIRA", en http://www.atlassian. com/ es/ software/jira.
- Microsoft, "Team Foundation Server - Detalles del Producto", en http://www.microsoft.com/visualstudio/esn/products/visual-studio-team-foundation-server-2012#product-edition-tfs-details.
- Archievo Org, "Achievo Flexible web-based project management", en http://www.achievo.org/.

ANEXOS

Anexo 1. Formato Plan de Gestión de Requerimientos

[Nombre del proyecto]

Plan de Gestión de Requerimientos

Versión [1.0]

[Este documento es la plantilla base para elaborar el documento Plan de Gestión de Requerimientos. Los textos que aparecen entre paréntesis rectos son explicaciones de que debe contener cada sección. Dichos textos se deben seleccionar y sustituir por el contenido que corresponda.]

Historia de revisiones

Fecha	Versión	Descripción	Autor
[dd/mm/aaaa]	[x.x]	[detalles]	[nombre]

1. Introducción

1.1. Propósito

[Esta sección debe contener el propósito y alcance del Plan de Gestión de Requerimientos.

Se debe especificar el uso que se le dará a los requerimientos que serán generados y se deben listar los elementos de estos que serán cubiertos por el Plan.

Además se debe especificar la porción del ciclo de vida del software que será cubierta por el Plan].

1.2. Generalidades

[Describir aspectos generales del contenido, uso y beneficios del documento]

2. Gestión

2.1. Organización , Responsabilidades y Actividades

[Descripción de cómo está organizado el equipo de trabajo, responsabilidad y actividades que deben realizar en cuanto a los requerimientos]

2.2. Herramientas

[Descripción de las herramientas de apoyo que se utilizarán para la gestión de los requerimientos, Ejm. plantillas de documentos.]

3. Programa de Administración de Requerimientos

[Se debe especificar los aspectos de definición de flujo del trabajo en la identificación de los requerimientos, el esquema de trazabilidad, los atributos, estados y beneficios de los requerimientos]

3.1. Trazabilidad

[Descripción de cómo ser generará la trazabilidad (relación) entre requerimientos.]

3.2. Atributos

[Descripción de los atributos de los requerimientos que se tomarán en cuenta.]

3.3. Estados

[Descripción de los estados que tendrán los requerimientos a fin de contemplarlos dentro de la definición de cambios.]

3.4. Importancia y Beneficios

[Descripción de los la importancia y beneficios de los requerimientos según los usuarios de negocio.]

4. Administración de Cambios de Requerimientos

[Se debe describir el flujo de proceso para los cambios en los requerimientos, con los responsables y acciones que deben realizar cada uno, Además, se debe definir los roles del comité de control de cambios]

4.1. Procesamiento y aprobación de las solicitudes de cambio

[Descripción del flujo de cambios y responsables.]

4.2. Comité de cambios

[Descripción de los roles y nombres de los conformantes del comité de cambios.]

4.3. Worflows y Actividades

[Diagramas de flujo para mostrar el proceso de administración de solicitudes de cambio.]

5. Capacitación y Recursos

[Se debe describir el mecanismo de difusión del plan]

5.1. Procesamiento y aprobación de las solicitudes de cambio

[Descripción del flujo de cambios y responsables.]

Anexo 2. Formato Caso de Uso

[Nombre del proyecto]

Caso de Uso: [**Código caso de uso - Nombre del caso de uso**]

Versión <1.0>

Código:	[Código del Caso de Uso]				
Nombre:	[Título del Caso de Uso]				
Elaborado Por:	[Nombre del responsable de la elaboración del CU]	**Revisado Por:**	[Nombre del responsable de la revisión del CU]	**Aprobado por:**	[Nombre del responsable de la aprobación del CU]
Fecha Elaboración:	dd-mmm-aa	**Fecha Revisión:**	dd-mmm-aa	**Fecha Aprobación:**	dd-mmm-aa

Historia de revisiones

Fecha	**Versión**	**Descripción**	**Autor**
[dd/mm/aaaa]	[x.x]	[detalles]	[nombre]

Objetivo:	[Indica el propósito que tiene la ejecución de caso de uso de negocio, el objetivo debe ser concreto]	
Descripción:	[Corresponde a la descripción conceptual del proceso al que representa el caso de uso. Es suficiente un solo párrafo para esta descripción.]	
Actores:	Externos	[Usuarios externos (Interesados, solicitantes, beneficiarios directos) que se benefician de lo descrito en el CU]
	Internos	[Usuarios del internos (propietarios, facilitadores o beneficiarios) que participan en el CU]
Precondiciones:	[Una precondición de un caso de uso es un estado previo que debe estar presente o debe cumplirse antes de que un caso de uso se ejecute.]	
Resultados	[Resultado(s) principal(es) que debería obtener el actor al terminar la ejecución de caso de uso]	
Flujo Normal de acciones:	[Este caso de uso comienza cuando el actor hace algo Un actor siempre inicia casos de uso. El caso de uso describe lo que el actor hace y que es lo que el negocio hace en respuesta. Está expresado como un diálogo entre el actor y el negocio/sistema. El caso de uso describe qué pasa dentro del negocio, pero no cómo ni por qué. Si se intercambia información, sea específico sobre qué es lo que se pasa hacia delante y	

	hacia atrás. Por ejemplo, no es muy claro decir que el actor ingresa información sobre el cliente. Es mejor decir que el actor ingresa el nombre y la dirección del cliente. Un Glosario de términos a menudo es útil para mantener manejable la complejidad del caso de uso ï puede querer definirse cosas como información del cliente allí para evitar que el caso de uso se ahogue en detalles. Las alternativas simples se pueden presentar dentro del texto del caso de uso. Si solo requiere unas pocas frases para describir que es lo que ocurre cuando hay una alternativa, hágalo directamente dentro de la sección Flujo de Eventos. Si el flujo alternativo es más complejo, use una sección separada para describirlo. Por ejemplo, una sub sección de Flujo Alternativo explica como describir alternativas más complejas. Un dibujo a veces vale mil palabras, a pesar de que no hay un sustituto para la prosa limpia y clara. Si mejora la claridad, siéntase libre de pegar descripciones gráficas de interfaces con el usuario, flujos de procesos u otras figuras dentro del caso de uso. Si un flujo grama es útil para presentar un proceso de decisión complejo, ¡no dude en usarlo! En forma similar para un comportamiento que depende de estados, un diagrama de transición de estados a menudo clarifica el comportamiento de un negocio mejor que páginas y páginas de texto. Use el medio de

	presentación correcto para su problema, pero sea cauteloso con el uso de terminología, notaciones o figuras que su audiencia pueda no entender. Recuerde que el propósito es clarificar, no oscurecer.] El actor, Acción 1. [Enumeración de la acción.] [Breve descripción de acción 1.] El actor, Acción 2. [Enumeración de la acción [Breve descripción de acción 2.]
Flujos alternos:	[Las alternativas más complejas se describen en una sección separada, a la cual se hace referencia en la sub Secuencia normal de acciones. Piense en las sub secciones de Flujos alternativas como comportamientos alternativas, cada flujo alternativa representa comportamiento alternativa debido a excepciones que ocurren en el flujo principal. Pueden ser tan largas como sea necesario para describir los eventos asociados con el comportamiento alternativa. Cuando un flujo alternativo termina, se continúa con los eventos del flujo de eventos principal a menos que se diga lo contrario.] 1.1. No cumple flujo normal de Acción 1 El actor, Acción Alternativa. [Breve descripción de acción] 1.2. No cumple flujo normal de Acción 1 El actor, Acción Alternativa. [Breve descripción de acción.]

	2.1. No cumple flujo normal de Acción 2 El actor, Acción alternativa [Breve descripción de acción.]
Postcondiciones:	[Una post condición de un caso de uso es una lista de posibles estados en los que puede quedar el negocio inmediatamente después de que el caso de uso termine.]
Restricciones Acceso	[Especificar restricciones a considerar para accesos al proceso de negocio/sistema descrito]
Impacto Funcional	[Especificar la afectación que tiene el proceso descrito en el caso de uso con respecto a los subsistemas/procesos/subprocesos/funciones a nivel del negocio/sistema en general]
Riesgos:	[Especificación de los riesgos de la ejecución y/o implementación del caso de uso de negocio/sistema]
Posibilidades:	[Descripción del potencial de mejora estimado del caso de uso de negocio/sistema]
Propietario del Proceso:	[Definición del quién es el responsable o propietario del proceso, la persona quien administra los cambios y planea los cambios de este.]
Notas y Comentarios:	[Datos adicionales que ayuden a aclarar los flujos de acciones o cualquier otro aspecto del casos de uso]
Diagramas:	[Diagramas UML desarrollados para el caso de uso] De Caso de Uso Código de Identificación - Nombre De Objetos

	Código de Identificación - Nombre
Requerimientos Especiales:	[Los requerimientos especiales del caso de uso de negocio/sistema son aquellos que no están cubiertos en los flujos básicos ni alternos, ni están contemplados dentro de las especificaciones suplementarias del negocio/sistema. Son requerimientos que se debe aplicar exclusivamente a este caso de uso de negocio/sistema]
Puntos de Extensión:	[Definición de los puntos de extensión en el flujo de acciones (normal o alterno)]
Referencias y Fuentes de Información:	

Gestión de Requerimientos

[En esta sección se debe llenar o marcar los atributos del requerimiento según corresponda]

Atributo	Valor
Estado	( ) Propuesto ( ) Aprobado ( ) Rechazado ( ) Incorporado
Beneficio	( ) Alto ( ) Medio ( ) Bajo
Complejidad	( ) Alta ( ) Media ( ) Baja
Riesgo	( ) Alto ( ) Medio ( ) Bajo
Estabilidad	( ) Alta ( ) Neutral ( ) Baja
Impacto sobre la Arquitectura	( ) Ninguno ( ) Extiende ( ) Modifica
Versión-release	

Anexo 3. Formato Plan de Aseguramiento de la Calidad

[Nombre del proyecto]

Plan de Aseguramiento de la Calidad (SQA)

Versión [1.0]

[Este documento es la plantilla base para elaborar el documento Plan de SQA. Los textos que aparecen entre paréntesis rectos son explicaciones de que debe contener cada sección. Dichos textos se deben seleccionar y sustituir por el contenido que corresponda.]

Historia de revisiones

Fecha	Versión	Descripción	Autor
[dd/mm/aaaa]	[x.x]	[detalles]	[nombre]

1. Propósito

[Esta sección debe contener el propósito y alcance del Plan de Calidad.

Se debe especificar el uso que se le dará al software que se está desarrollando y se deben listar los elementos del software que serán cubiertos por el Plan.

Además se debe especificar la porción del ciclo de vida del software que será cubierta por el Plan. (Ej.: Este Plan solo cubre la parte del ciclo de vida correspondiente al desarrollo del software pero no cubre la parte del ciclo de vida correspondiente al mantenimiento.)]

2. Referencias

[1]ANSI/IEEE Std 730.1-1989, IEEE Standard for Software Quality Assurance Plans.

3. Gestión

[Se debe especificar la organización, actividades y responsables.]

3.1 Organización

[Distinguir las líneas de trabajo dentro de la organización que tienen influencia y controlan la calidad del software.

Descripción de cómo está organizado el equipo de trabajo y de las dependencias o independencias de las líneas de trabajo antes mencionadas.]

3.2 Actividades

- **Ciclo de vida del software cubierto por el Plan**

[Esta sección debe contener las etapas más importantes del ciclo de vida del software que cubre el Plan. (Ej.: Etapa de Requerimientos y análisis)

Además debe contener una lista con todos los productos de proyecto que tendrán revisiones de calidad.]

- **Actividades de calidad a realizarse**

Las tareas a ser llevadas a cabo deberán reflejar las evaluaciones a realizar, los estándares a seguir, los productos a revisar, los procedimientos a seguir en la elaboración de los distintos productos y los procedimientos para informar de los defectos detectados a sus responsables y realizar el seguimiento de los mismos hasta su corrección.

Las actividades que se realizarán son:

- Revisar cada producto
- Revisar el ajuste al proceso
- Realizar Revisión Técnica Formal (RTF)
- Asegurar que las desviaciones son documentadas.

- **Revisar cada producto**

En esta actividad se revisan los productos que se definieron como claves para verificar en el Plan de calidad.

Se debe verificar que no queden correcciones sin resolver en los informes de revisión previos, si se encuentra alguna no resuelta, debe ser incluida en la siguiente revisión. Se revisan los productos contra los estándares, utilizando la checklist definida para el producto.

Se debe identificar, documentar y seguir la pista a las desviaciones encontradas y verificar que se hayan realizado las correcciones.

Como salida se obtiene el Informe de revisión de SQA, este informe debe ser distribuido a los responsables del producto y se debe asegurar de que son consientes de desviaciones o discrepancias encontradas.

- **Revisar el ajuste al proceso**

En esta actividad se revisan los productos que se definieron como claves para verificar el cumplimiento de las actividades definidas en el proceso. Con el fin de asegurar la calidad en el producto final del desarrollo, se deben llevar a cabo revisiones sobre los productos durante todo el ciclo de vida del software.

Se debe recoger la información necesaria de cada producto, buscando hacia atrás los productos previos que deberían haberse generado, para poder establecer los criterios de revisión y evaluar si el producto cumple con las especificaciones.

Esta información se obtiene de los siguientes documentos:

Plan del Proyecto, Plan de la iteración, Plan de Verificación.

Antes de comenzar, se debe verificar en los informes de revisión previos que todas las desviaciones fueron corregidas, si no es así, las faltantes se incluyen para ser evaluadas.

Como salida se obtiene el Informe de revisión de SQA correspondiente a la evaluación de ajuste al Proceso, este informe debe ser distribuido a los responsables de las actividades y se debe asegurar de que son concientes de desviaciones o discrepancias encontradas.

- **Realizar Revisión Técnica Formal (RTF)**

El objetivo de la RTF es descubrir errores en la función, la lógica ó la implementación de cualquier producto del software, verificar que satisface sus especificaciones, que se ajusta a los estándares establecidos, señalando las posibles desviaciones detectadas. Es un proceso de revisión riguroso, su objetivo es llegar a detectar lo antes posible, los posibles defectos o desviaciones en los productos que se van generando a lo largo del desarrollo. Por esta característica se adopta esta práctica para productos que son de especial importancia.

En la reunión participan el responsable de SQA e integrantes del equipo de desarrollo.

Se debe convocar a la reunión formalmente a los involucrados, informar del material que ellos deben preparar por adelantado, llevar una lista de preguntas y dudas que surgen del estudio del producto a ser revisado.

La duración de la reunión no debe ser mayor a dos horas.

Como salida se obtiene el Informe de RTF.

- **Asegurar que las desviaciones son documentadas**

Las desviaciones encontradas en las actividades y en los productos deben ser documentadas y ser manejadas de acuerdo a un procedimiento establecido.

Se debe chequear que los responsables de cada plan los modifiquen cada vez que sea necesario, basados en las desviaciones encontradas.

- **Relaciones entre las actividades de SQA y la planificación**

[En esta sección se incluye una lista con las actividades de calidad a realizarse durante el proyecto, especificando en que semana del proyecto se realizan.]

Actividad	**Semana cuando se realiza**
Actividad 1	[Semana]
[RTF de Estimaciones y Mediciones]	[Semana 6]

- **Responsables**

[Identificar los distintos responsables de cada actividad identificada.]

4. Documentación

4.1 Propósito

Identificación de la documentación relativa a desarrollo, Verificación & Validación, uso y mantenimiento del software.

Establecer como los documentos van a ser revisados para chequear consistencia: se confirman criterio e identificación de las revisiones.

4.2 Documentación mínima requerida

La documentación mínima es la requerida para asegurar que la implementación logrará satisfacer los requerimientos.

- **Especificación de requerimientos del software**

El documento de especificación de requerimientos deberá describir, de forma clara y precisa, cada uno de los requerimientos esenciales del software además de las interfaces externas.

El cliente deberá obtener como resultado del proyecto una especificación adecuada a sus necesidades en el área de alcance del proyecto, de acuerdo al compromiso inicial del trabajo y a los cambios que este haya sufrido a lo largo del proyecto, que cubra aquellos aspectos que se haya acordado detallar con el cliente.

La especificación debe:

- Ser completa :
    a. Externa, respecto al alcance acordado.
    b. Internamente, no deben existir elementos sin especificar.
- Ser consistente, no pueden haber elementos contradictorios.
- Ser no ambigua, todo término referido al área de aplicación debe estar definido en un glosario.
- Ser verificable, debe ser posible verificar siguiendo un método definido, si el producto final cumple o no con cada requerimiento.

- Estar acompañada de un detalle de los procedimientos adecuados para verificar si el producto cumple o no con los requerimientos.
- Incluir requerimientos de calidad del producto a construir.

Los requerimientos de calidad del producto a construir son considerados dentro de atributos específicos del software que tienen incidencia sobre la calidad en el usoôy se detallan a continuación:

Funcionalidad

a. adecuación a las necesidades

b. precisión de los resultados

c. interoperabilidad

d. seguridad de los datos

Confiabilidad

a. madurez

b. tolerancia a faltas

c. recuperabilidad (Ver si aplica)

Usabilidad

a. comprensible

b. aprendible

c. operable

d. atractivo

Eficiencia

a. comportamiento respecto al tiempo (Ver si aplica)

b. utilización de recursos

Mantenibilidad

a. analizable

b. modificable

c. estable, no se producen efectos inesperados luego de modificaciones

d. verificable

Portabilidad

a. adaptable (Ver si aplica)

b. instalable

c. co-existencia

d. reemplazante (Ver si aplica)

Cada uno de estos atributos debe cumplir con las normas y regulaciones aplicables a cada uno.

- **Descripción del diseño del software**

El documento de diseño especifica como el software será construido para satisfacer los requerimientos.

Deberá describir los componentes y subcomponentes del diseño del software, incluyendo interfaces internas. Este documento deberá ser elaborado primero como Preliminar y luego será gradualmente extendido hasta llegar a obtener el Detallado.

El cliente deberá obtener como resultado del proyecto el diseño de un producto de software que cubra aquellos aspectos que se haya acordado con el cliente incorporar al diseño, en función de la importancia que estos presenten y de sus conexiones lógicas.

El diseño debe:

- Corresponder a los requerimientos a incorporar:

  a. Todo elemento del diseño debe contribuir a algún requerimiento

  b. La implementación de todo requerimiento a incorporar debe estar contemplada en por lo menos un elemento del diseño.

- Ser consistente con la calidad del producto

- **Plan de Verificación & Validación**

El Plan de V & V deberá identificar y describir los métodos a ser utilizados en:

- La verificación de que:

a. los requerimientos descritos en el documento de requerimientos han sido aprobados por una autoridad apropiada. En este caso sería que cumplan con el acuerdo logrado entre el cliente y el equipo.

b. los requerimientos descritos en el documento de requerimientos son implementados en el diseño expresado en el documento de diseño.

c. el diseño expresado en el documento de diseño esta implementado en código.

- Validar que el código, cuando es ejecutado, se adecua a los requerimientos expresados en el documento de requerimientos.

- **Reportes de Verificación & Validación**

Estos documentos deben especificar los resultados de la ejecución de los procesos descritos en el Plan de V & V.

- **Documentación de usuario**

La documentación de usuario debe especificar y describir los datos y entradas de control requeridos, así como la secuencia de entradas, opciones,

limitaciones de programa y otros elementos necesarios para la ejecución exitosa del software.

Todos los errores deben ser identificados y las acciones correctivas descritas.

Como resultado del proyecto el cliente obtendrá una documentación para el usuario de acuerdo a los requerimientos específicos del proyecto.

- **Plan de Gestión de configuración**

El Plan de gestión de configuración debe contener métodos para identificar componentes de software, control e implementación de cambios, y registro y reporte del estado de los cambios implementados.

4.3 Otros documentos

[Esta sección puede contener otros documentos que se identifiquen de incidencia en la calidad del producto a desarrollar, por ejemplo:

- Plan de desarrollo
- Plan de proyecto
- Manual de estándares y procedimientos
- Etc...]

5. Estándares, prácticas, convenciones y métricas

[Esta sección deberá cumplir con las siguientes funciones:

- Identificar los estándares, prácticas, convenciones y métricas que serán aplicadas para la evaluación de Calidad.
- Indicar como será monitoreado y asegurado el cumplimiento con estos elementos.]

5.1 Estándar de documentación

Como estándares de documentación se definirán dos documentos:

- Estándar de documentación técnica y
- Estándar de documentación de usuario.

La documentación técnica del producto debe:

- Ser adecuada para que un grupo independiente del de desarrollo pueda encarar el mantenimiento del producto.
- Incluir fuentes, Modelos de Casos de Uso, Objetos

Para la escritura de documentos se han definido plantillas para ser utilizadas en la elaboración de entregables.

En estas plantillas se definen:

- encabezado y pie de página.
- fuente y tamaño de fuente para estilo normal
- fuente y tamaño de fuente para los títulos a utilizar
- datos mínimos que se deben incluir: fecha, versión y responsables.

[Esta sección debe incluir todos los estándares de documentación que se utilicen durante el desarrollo del proyecto.]

5.2 Estándar de verificación y prácticas

Se utilizan las prácticas definidas en el Plan de Verificación y Validación.

Como estándar se utiliza el documento de:

Std 1012-1986 IEEE Standard for Software Verification and Validation Plans.

[Esta sección debe incluir todos los estándares de verificación y prácticas que se utilicen durante el desarrollo del proyecto.]

5.3 Otros Estándares

[En esta sección se deberán definir otros estándares a utilizar.]

6. Revisiones y auditorías

6.1 Objetivo

Definición de las revisiones y auditorías técnicas y de gestión que se realizarán.

Especificación de cómo serán llevadas a cabo dichas revisiones y auditorías.

6.2 Requerimientos mínimos

[Se especifican las revisiones y auditorías que deben realizarse como mínimo, así como la agenda para la realización de las mismas.]

- **Revisión de requerimientos**

Esta revisión se realiza para asegurar que se cumplió con los requerimientos especificados por el Cliente.

- **Revisión de diseño preliminar**

Esta revisión se realiza para asegurar la consistencia y suficiencia técnica del diseño preliminar del software.

- **Revisión de diseño crítico**

Esta revisión se realiza para asegurar la consistencia del diseño detallado con la especificación de requerimientos.

- **Revisión del Plan de Verificación & Validación**

Esta revisión se realiza para asegurar la consistencia y completitud de los métodos especificados en el Plan de V & V.

- **Auditoría funcional**

Esta auditoría se realiza previa a la liberación del software, para verificar que todos los requerimientos especificados en el documento de requerimientos fueron cumplidos.

- **Auditoría física**

Esta revisión se realiza para verificar que el software y la documentación son consistentes y están aptos para la liberación.

- **Auditorías internas al proceso**

Estas auditorías son para verificar la consistencia: del código versus el documento de diseño, especificaciones de interfase, implementaciones de diseño versus requerimientos funcionales, requerimientos funcionales versus descripciones de testeo.

- **Revisiones de gestión**

Estas revisiones se realizan periódicamente para asegurar la ejecución de todas las actividades identificadas en este Plan. Deben realizarse por una persona ajena al grupo de trabajo (en caso de que sea posible).

- **Revisión del Plan de gestión de configuración**

Esta revisión se realiza para asegurar la consistencia y completitud de los métodos especificados en el Plan de gestión de configuración.

- **Revisión Post Mortem**

Esta revisión se realiza al concluir el proyecto para especificar las actividades de desarrollo implementadas durante el proyecto y para proveer recomendaciones.

- **Agenda**

[En esta sección se deberá especificar la agenda para las revisiones y auditorías detalladas anteriormente.]

6.3 Otras revisiones

- **Revisión de documentación de usuario**

Se revisa la completitud, claridad, correctitud y aplicación de uso.

7. Verificación

[Se debe identificar todas las verificaciones que no fueron identificadas en el Plan de V & V para el software y debe especificar los métodos a ser usados.]

8. Reporte de problemas y acciones correctivas

[Esta sección debe incluir: Descripción de las prácticas y procedimientos que se seguirán para el reporte, seguimiento, y resolución de los problemas surgidos en el desarrollo de software; especificar los responsables comprometidos con la implementación de estas acciones correctivas.]

9. Herramientas, técnicas y metodologías

[Se deben identificar herramientas de software, técnicas, y metodologías de soporte para las actividades de aseguramiento de calidad.

Ver sección 3.]

10. Gestión de riesgos

[Se deben especificar los métodos y procedimientos utilizados para especificar, monitorear, y controlar las áreas de riesgo durante el proyecto.

Los riesgos identificados, la estrategia de mitigación, monitoreo y plan de contingencia a ser llevados a cabo, serán descritos en el Documento de Gestión de Riesgos, con lo cual se podrá hacer referencia a él.]

Anexo 4. Plan de Pruebas

[Nombre del proyecto]

Plan de Pruebas

Versión [1.0]

[Este documento es la plantilla base para elaborar el documento Plan de Pruebas. Los textos que aparecen entre paréntesis rectos son explicaciones de que debe contener cada sección. Dichos textos se deben seleccionar y sustituir por el contenido que corresponda.]

Historia de revisiones

Fecha	Versión	Descripción	Autor
[dd/mm/aaaa]	[x.x]	[detalles]	[nombre]

1. Introducción

1.1 Propósito

[Esta sección debe contener el propósito y alcance del Plan de Pruebas.]

1.2 Generalidades

[Descripción de aspectos generales del contenido, uso y beneficios del documento]

2. Objetivos

[Descripción de los objetivos que se persigue con el plan de pruebas]

3. Detalle Plan de Pruebas

3.1 Fases de Pruebas

[Descripción de los tipos de pruebas, alcance y validación a aplicar por cada tipo]

3.2 Responsables de Pruebas

[Descripción de los responsables por cada fase de pruebas]

3.3 Alcance de Ejecución de Pruebas

[Detalle de los requerimientos (casos de uso) a probar, con la respectiva prioridad, se debe incluir el detalle de los escenarios básico y alternos]

3.4 Cronograma de Pruebas

[Definir los tiempos de las pruebas en detalle]

3.5 Generalidades para las pruebas

[Se debe detallar los ambientes en los que se deben realizar las pruebas, usuarios que probarán, accesos necesarios]

Anexo 5. Caso de Prueba

Caso de Prueba

Proyecto			Código incidente:	
Proceso/sistma			Nombre incidente:	
Versión/Release:			Ambiente Pruebas:	
Perifles de prueba				
Responsables prueba:				
Apoyo pruebas:				
Fecha Inicio Pruebas		Fecha Fin Pruebas		
Tipo de Prueba:				

Fecha	Tipo de elemento	Nombre de elemento	Descripción del proceso a probar	Descripción del resultado obtenido	Estado (Satisfactorio = OK, Insatisfactorio = X)	Observaciones y recomendaciones	Fecha solución
FLUJO NORMAL							
FLUJOS ALTERNOS							

Anexo 6. Solicitud de Cambio

Código:	[Código de la Solicitud de Cambio]
Nombre:	[Título del cambio]
Macro proceso /subproceso:	[Proceso al que afecta]
Tipo de Cambio:	( ) Nuevo ( ) Mejora ( ) Error
Solicitado por:	[Nombre de la persona que solicita el cambio]
Elaborado Por:	[Nombre de la persona elabora el requerimiento del cambio]
Fecha de registro solicitud:	[Fecha de generación de la solicitud de cambio]
Descripción del Problema:	[Descripción del problema que suscita el cambio]
Afecta a:	[Detalle de procesos y subprocesos que afecta el cambio] 1. 2. 3.
Alternativas de Solución:	[Descripción de alternativas de solución que se planteen] 1. 2. 3.
Código Caso(s) de Uso(s) :	[Códigos de requerimientos (casos de uso) relacionados al cambio]
Estado Solicitud :	[Estado de la solicitud de cambio]
Fecha Estado :	[Fecha del estado de la solicitud de cambio]

Responsable Aprobación /Rechazo de solicitud	[Responsable de estado de solicitud de cambio]

Anexo 7. Plan de Despliegue

[Nombre del proyecto]

Plan de Despliegue

Versión [1.0]

[Este documento es la plantilla base para elaborar el documento Plan de Despliegue. Los textos que aparecen entre paréntesis rectos son explicaciones de que debe contener cada sección. Dichos textos se deben seleccionar y sustituir por el contenido que corresponda.]

Historia de revisiones

Fecha	Versión	Descripción	Autor
[dd/mm/aaaa]	[x.x]	[detalles]	[nombre]

1. Introducción

1.1 Propósito

[Esta sección debe contener el propósito del plan de despliegue]

1.2 Alcance

[Describir el alcance del documento.]

2. Planificación del Despliegue

2.1 Responsabilidades

[Detallar las responsabilidades y compromisos a nivel del Cliente y de la Empresa contratada para lograr la instalación de la aplicación de software en el ambiente de producción.]

3. Recursos

3.1 Unidad de Despliegue

[Detallar el esquema de instalación, y entregables para dicho efecto, adicionalmente se debe describir cada componente de software de soporte que se requiera, incluyendo versiones y configuraciones que se debe aplicar , la documentación y personal de soporte para el despliegue en producción del producto final.]

4. Entrenamientos

[Detallar el esquema de capacitación que se realizara a los administradores y usuarios finales]

5. Configuración de la plataforma

[Incluir una guía de instalación y configuración de las plataformas a nivel de desarrollo y pruebas además de la de producción. Esta guía debe contemplar la Base de Datos, Front- Ent y Aplicación]

Anexo 8. Artefactos que SYSGENSA genera en sus proyectos

Nombre	Descripción
Plan de Desarrollo del Software	Describe el plan general que se utilizará para el desarrollo del proyecto. El detalle de las iteraciones individuales que formarán parte del software se describe en planes de iteración.
Lista de Riesgos	Este documento incluye una lista de los riesgos conocidos, vigentes y que se puedan presentar en el proyecto, organizados en orden decreciente de importancia y con acciones específicas de contingencia o para su mitigación.
Visión del negocio	El documento de visión del negocio contiene los objetivos de muy alto nivel del modelado del negocio. Sirve de insumo para el documento de visión y comunica los ñquéò y ñpor quéò fundamentales relacionados con el proyecto y sirve de referencia respecto a todas las decisiones futuras que deberían ser validadas.
Vista de procesos del negocio	Para conseguir los objetivos del negocio al cual se plantea apoyar con un producto de software, se debe organizar las actividades por medio de un conjunto de procesos de negocio. Cada uno de ellos debe caracterizarse por una colección de datos que son producidos y manipulados mediante un conjunto de tareas, en las que ciertos actores

	(por ejemplo, empleados, departamentos) participan de acuerdo a un flujo de trabajo determinado. Además, estos procesos se hallan sujetos a un conjunto de reglas de negocio, que determinan las políticas y la estructura de la información de la organización; por tanto, la finalidad del modelado del negocio es describir cada proceso del negocio, especificando sus datos, actividades (o tareas), roles (o actores, o agentes) y reglas de negocio. El primer paso del modelado del negocio consiste en capturar los procesos de negocio principales. La definición del conjunto de procesos del negocio es una tarea crucial, ya que define los límites del proceso de modelado posterior de los casos de uso de negocio (cada proceso de negocio se representa como un caso de uso de negocio y puede representarse textual y/o gráficamente). El artefacto o documento resultante de esta actividad se conoce como ñVista de procesos de negocioò
Glosario	Es un documento que define los principales términos usados en el proyecto y permite establecer una terminología consensuada para todos los involucrados.
Modelo de Casos de Uso de Negocio	Es un modelo de las metas y funciones de negocio del sistema al cual está enfocado el proyecto, vistas desde la perspectiva de los actores externos. Permite situar al

	sistema en el contexto organizacional haciendo énfasis en los objetivos en este ámbito, además de identificar roles y entregables en la organización.
Modelo de Casos de Uso de Sistema	El modelo de Casos de Uso presenta las funciones del sistema y los actores que hacen uso de ellas. Está constituido por todos los casos de uso que deberán ser implementados en el sistema y que pueden ser agrupados en paquetes de casos de uso. Este modelo se utiliza como insumo esencial para las actividades de análisis, diseño y pruebas; y se representa con un diagrama de casos de uso en notación UML. Un paquete de casos de uso es una colección de casos de uso, actores, relaciones, diagramas y otros paquetes; éste es utilizado para estructurar el modelo de casos de uso, dividiéndolo en partes más pequeñas. Para los casos de uso se realiza una descripción detallada utilizando una plantilla de documento, donde se incluyen: precondiciones, post-condiciones, flujo de eventos, requisitos no-funcionales asociados; también, para casos de uso cuyo flujo de eventos sea complejo podrá adjuntarse una representación gráfica mediante un Diagrama de Actividad.

Especificaciones Suplementarias	Este documento capturará todos los requisitos que no han sido incluidos como parte de los casos de uso y se refieren como requerimientos no-funcionales globales. Dichos requisitos incluyen: requisitos legales o normas, aplicación de estándares, requisitos de calidad del producto, tales como: confiabilidad, desempeño, etc., u otros requisitos de ambiente, tales como: sistema operativo, requisitos de compatibilidad, etc.
Visión	Este documento define la visión del producto (de software) desde la perspectiva del cliente, especificando las necesidades y características del producto. Constituye una base de acuerdo en cuanto a los requisitos del sistema.
Modelo de Diseño	Este modelo establece la realización de los casos de uso en clases y pasando desde una representación en términos de análisis (sin incluir aspectos de implementación) hacia una de diseño (incluyendo una orientación hacia el entorno de implementación), de acuerdo al avance del proyecto.
Modelo de Datos	La información del sistema será soportada por una base de datos relacional, este modelo debe describir la representación lógica de los datos usados por la aplicación. Para expresar este modelo se utilizará un

	diagrama de clases (donde se utiliza un perfil UML para Modelado de Datos, para conseguir la representación de tablas, claves, etc.) y posteriormente un diagrama entidad/relación, de acuerdo con el enfoque para modelado relacional de datos.
Arquitectura del Software	El documento de arquitectura del software ofrece un panorama arquitectónico comprensivo y completo del sistema, utilizando un sin número de vistas arquitectónicas que describen diferentes aspectos del sistema. Como parte de su representación mediante diagramas UML, está compuesta por un conjunto relevante de vistas arquitectónicas: Casos de uso, lógica, procesos, despliegue, implementación y datos.
Plan de Iteración	Es un conjunto de actividades y tareas ordenadas temporalmente, con recursos asignados, dependencias entre ellas. Se realiza para cada iteración y para todas las fases.
Evaluación de la Iteración	Este documento incluye la evaluación de los resultados de cada iteración, el grado en el cual se han conseguido los objetivos de la iteración, las lecciones aprendidas y los cambios a ser realizados.
Material de Apoyo al Usuario Final	Corresponde a un conjunto de documentos y facilidades de uso del sistema, incluyendo: Guías del Usuario, Guías

	de Operación, Guías de Mantenimiento y Sistema de Ayuda en Línea.
Builds	Un ñbuildò es una versión operacional del sistema que contiene un subconjunto de capacidades funcionales preliminares a la versión 1.0. Comprende uno o más elementos de implementación (a menudo ejecutable), cada uno construido a partir de otros elementos, usualmente por un proceso de compilación y encadenamiento del código fuente. Se espera contar con dos ñbuildsò preliminares, que corresponden a las versiones alfa y beta, resultantes de las dos iteraciones de construcción; y un ñbuildò final que corresponde a la versión 1.0 del sistema.
Producto de Software	El producto, al final de la fase de elaboración y a partir de la primera iteración de la fase de construcción es desarrollado de manera incremental e iterativa, obteniéndose un nuevo release al final de cada iteración. El producto de software incluye los archivos de código ejecutables, scripts de instalación, archivos de configuración empaquetados, con los mecanismos apropiados para facilitar su instalación (instrucciones de instalación). Una unidad de despliegue consiste de un ñbuildò (un grupo

	ejecutable de componentes), documentos (material de soporte del usuario final y notas de release) y artefactos de instalación. En este caso, por tratarse de un sistema que se despliega en una granja de servidores, la unidad de despliegue está asociada con conjuntos de componentes instalables en los servidores de cada una de las capas: Presentación, negocio, reportes, meta datos, datos.

Anexo 9. Herramientas de apoyo para ciclo de desarrollo de software

Gestión de Requerimientos[18]		
Nombre	**Características**	**Vendedor**
IBM Rational Requisite Pro	Es una herramienta centrada en documentos, que almacena los requisitos asociándolos a documentos. Ayuda especialmente en el control de cambio de requisitos, con trazabilidad para especificaciones de software y pruebas. Está muy ligado a MS Word. La herramienta permite el uso de Oracle sobre Unix o Windows como ñback-end databaseò y también soporta SQL Server sobre Windows[19].	IBM http://www-01.ibm.com/software/awdtools/reqpro/
CaliberRM	Esta herramienta es para sistemas grandes y complejos y proporciona una base de datos de requisitos con trazabilidad. La compañía ve a los requisitos como parte del proceso de gestión de la calidad del software, el cual es considerado también, las pruebas (testing) y el trazado de defectos (defect tracking). Caliber está basado en Internet y maneja referencia de documentos, responsabilidad de	Technology Builders Inc. http://www.tbi.com

[18] Fábricas de software, ñHerramientas de Desarrollo de Software ï Gestión de Requisitosò, en http://www.fabricasdesoftware.es/herramientas

[19] IBM, ñRational RequisiteProò, en http://www-01.ibm.com/software/awdtools/reqpro/

	usuario, trazabilidad, prioridad y estado entre otras características.	
IRQA(Integral Requisite Analizer)	Es una de las herramientas de Gestión de Requisitos más completas del mercado. Los requisitos que se capturan se almacenan en documentos Word y las descripciones de los mismos pueden referenciar a documentos externos como son tablas, gráficos y hojas de cálculo de Microsoft Excel. Permite establecer relaciones entre requisitos, además se puede integrar con Rational Rose.	Tcp Sistemas E Ingeniería S.L. http://www.visuresolutions.com/irqa-requirements-tool
IBM Rational DOORS	Es un sistema multiplataforma diseñado para la Gestión de Requisitos[20]. mediante la captura, trazabilidad, enlazado, análisis y manejo de los cambios que en ellos se realicen. Mediante el uso de Doors se puede realizar análisis de trazabilidad para identificar las áreas de riesgo y resulta fácil manejar los cambios que tengan lugar en los requisitos. Además permite gestionar un gran número de requisitos de forma eficiente mediante el uso de una base de datos sencilla, lo que se	IBM http://www-01.ibm.com/software/awdtools/doors/productline/

[20] IBM, ñDOORS familyò, en http://www-01.ibm.com/software/awdtools/doors/ productline/

	conoce como característica de gran escalabilidad	
REM (Requisite Management)	Es una herramienta de uso libre puede ser utilizada únicamente sobre Windows, ha sido utilizada con frecuencia para fines educacionales. REM permite generar un documento normalizado en el que se pueden incluir los requisitos necesarios para el desarrollo de sistemas de información. La documentación es generada en formato HTML.	Software Libre http://salasuach.googlecode.com/files/REM_1_2_2.rar
OSRMT (Open Source Requirements Management Tool)	Es una herramienta diseñada para dar cobertura a todo el ciclo de vida de desarrollo del software. Dispone de control de versiones, permite definir requerimientos derivados, es posible definir tanto casos de uso, como casos de prueba y brinda la posibilidad de definir atributos para los requisitos.	Software Libre http://sourceforge.net/projects/osrmt/
Nimble	Es una herramienta OpenSource para la gestión de requisitos, que permite la trazabilidad durante el ciclo de vida del	Software Libre http://nimble.sourceforge.net/

	producto. En desarrollo constante, se pretende que sea una herramienta web[21].	
Visure Requeriments	Es una herramienta para la ingeniería de requisitos, capaz de racionalizar los procesos de solución de los requisitos, lo que permite una colaboración más eficaz, aumentar la calidad, el apoyo a las necesidades de captura, análisis, especificación, validación y verificación, gestión y reutilización[22].	Headquarters http://www.visuresolutions.com/
HP Quality Center	Es una aplicación web que admite todos los aspectos esenciales de la gestión de pruebas. Ayuda en la gestión de requisitos de software, planificación y programación de pruebas, análisis de resultados y gestión de defectos.	HP https://h10078.www1.hp.com/cda/hpms/display/main/
Contour	Es una solución empresarial y colaborativa (a partir de su última versión 3.0) para la gestión de requisitos. Es una solución basada en Web y que permite capturar los requisitos en un solo lugar, con análisis de impacto y trazabilidad, control de cambios y manejo de equipos.	Jama http://www.jamasoftware.com/contour/contour3.php

[21] aNimble Platform, "Project Information", en http://nimble.sourceforge.net/

[22] Requirements Engineering Software - Visure Solutions, "Get to Know Visure Requirements Software" en http://www.visuresolutions.com/

Case Spec	Herramienta que facilita el seguimiento y la trazabilidad de los requisitos de un sistema, así como la gestión del ciclo de vida del mismo.	Goda Software http://analysttool.com/
Let's req	Let's req, es una aplicación web para gestionar proyectos y requisitos de sistemas software de forma sencilla.	Software Libre http://letsreq.com/
Rmtoo	Herramienta libre diseñada para manejar adecuadamente los requerimientos.	Software Libre http://www.flonatel.de/projekte/rmtoo/
Remas	Basado en eclipse utilizado para manejar los requerimientos[23].	Software Libre http://code.google.com/p/remasystem/downloads/list
Enterprise Arquitect	Es una herramienta comprensible de diseño y análisis UML, cubre el desarrollo de software desde el paso de los requerimientos a través de las etapas del análisis, modelos de diseño, pruebas y mantenimiento. EA es una herramienta multiusuario, basada en Windows, diseñada para ayudar a construir	Sparx Systems http://www.sparxsystems.com.ar/products/ea.html

[23] remasysytem, "Project Information", en http://code.google.com/p/remasystem/

	software robusto y fácil de mantener. Ofrece salida de documentación flexible y de alta calidad[24].	
Gestión de configuración, cambios y release[25]		
Nombre	**Características**	**Vendedor**
CVS (Concurrent Versioning System)	CVS es un almacén de ficheros donde se puede trazar la historia de los ficheros fuente que en él se depositan. Se pueden recuperar las versiones anteriores a la actual , también es capaz de mezclar las modificaciones que dos personas han hecho sobre un mismo módulo para evitar que una sobrescriba a la otra. Todo esto está soportado en un almacén centralizado. El tipo de proyectos sobre los que se puede usar CVS son los de pequeño y mediano tamaño[26].	Software Libre http://www.nongnu.org/cvs/
Subversion (SVN)	Es un sistema de control de versiones: maneja los archivos y las carpetas de un proyecto y sus modificaciones en el	Software Libre http://subversion.tigris.org/

[24] SPARX Systems,"Enterprise Architect - Herramienta de diseño UML", en http://www.sparxsystems.com.ar/products/ea.html

[25] Fábricas de software, ñHerramientas de Desarrollo de Software ï Gestión de la Configuraciónò, en http://www.fabricasdesoftware.es/herramientas

[26] CVS - Open Source Version Control, "Introduction to CVS", en http://www.nongnu.org/cvs/

	transcurso del tiempo.	
Aegis	Es un sistema de administración de configuración de software basado en transacciones. Proporciona un marco dentro del que un equipo de desarrolladores puede trabajar en muchos cambios de un programa independientemente, y Aegis coordina la integración de esos cambios dentro del código fuente maestro del programa, con el menor trastorno posible[27].	Software Libre http://aegis.sourceforge.net/
BitKeeper	Es un sistema de control de versiones distribuido para el código fuente de los programas[28].	Bitmover Inc. http://www.bitkeeper.com/
Rational Clear Case	Es una solución de gestión de configuración de software que proporciona control de versiones, gestión de espacios de trabajo, soporte al desarrollo paralelo y auditoría de compilaciones[29].	IBM http://www.ibm.com/software/products/es/es/clearcase/
Bugzilla	Es una herramienta basada en Web de	Software Libre

[27] Aegis 4.24, "Aegis Propaganda", en http://aegis.sourceforge.net/propaganda /index.html
[28] BITKEEPER, "Products", http://www.bitkeeper.com/Products.html
[29] IBM, "Una Solución de gestión de configuración de software", en http://www.ibm.com/software/products/es/es/clearcase/

	seguimiento de errores (Bug Traking System ó BTS), permite organizar en múltiples categorías los defectos de software, permitiendo seguimiento de múltiples productos con varias versiones, y a su vez compuestos por múltiples componentes. La categorización de los defectos del software se la aplica en función de su prioridad y severidad.	www.bugzilla.mozilla.org
Mantis Bug tracker	Es un sistema de gestión de incidentes, es muy popular y está basado en el control de errores web, Está escrito en PHP y trabaja con MySQL, MSSql y ProstgreSQL, y un servidor Web[30].	Software Libre http://www.mantisbt.org/
Bazaar	Es un sistema de control de versiones que ayuda a seguir la historia de un proyecto en el tiempo, facilita la colaboración de los desarrolladores indistintamente de su ubicación[31].	Software Libre http://bazaar.canonical.com/
Git	Software de control de versiones, está pensado en la eficiencia y confiabilidad del mantenimiento de versiones de	Software Libre http://git.or.cz/

[30] Mantis Bug Tracker, "Feature List", en http://www.mantisbt.org/
[31] Bazaar, "What is Bazzar?", en http://bazaar.canonical.com/

	aplicaciones cuando estas tienen un gran número de archivos de código fuente. Soporta Gestión Distribuida.	
Mercurial	Es un sistema de control de versiones distribuido que ofrece indexación cruzada de ficheros y conjuntos de cambios, protocolos de sincronización SSH y HTTP eficientes respecto al uso de CPU y ancho de banda, fusión entre ramas de desarrolladores, interfaz web autónoma integrada, aplicable para UNIX, MacOS X, y Windows".	Software Libre http://www.selenic.com/mercurial/
Microsoft Visual Soursafe	Es un sistema de control de versiones en el nivel de archivos, que permite del trabajo de distintas versiones de un proyecto al mismo tiempo, su funcionalidad es ventajosa en versiones de código paralelas, admite desarrollo multiplataforma.	Microsoft http://www.microsoft.com/

Control de Calidad - Pruebas[32]		
Nombre	**Características**	**Vendedor**
Sonar	Una herramienta que permite gestionar la calidad del código fuente. Al instalarla se puede recopilar, analizar, y visualizar métricas del código fuente. Sonar es básicamente la fusión de las siguientes herramientas Checkstyle y PMD, más otras como Findbugs, Clover y Cobertura. También realiza un histórico de todas las métricas del proyecto[33]. Permite visualizar informes con resúmenes de las métricas. Trabaja, principalmente, para Java. Aunque da soporte, gracias a la amplia librería de plugins , a los siguientes lenguajes: ABAP, C, Cobol, C#, Delphim Pascal, Flex/ActionScript, Groovy, JavaScript, Natural, PHP, PL/SQL, Visual Basic 6, Web y XML.	Software Libre http://sonar.codehaus.org/
PMD	Analizador estático de código que utiliza	Software Libre

[32] Tratando de entenderlo, "Herramientas de análisis de calidad del código", en http://tratandodeentenderlo.blogspot.com/2009/10/herramientas-de-analisis-de-calidad-del.html

[33] Fábricas de software, "Herramientas de Desarrollo de Software ï Calidad, Verificación y Validación » SONAR", en http://www.fabricasdesoftware.es/herramientas/calidad-verificacion-y-validacion/sonar/

	conjuntos de reglas para identificar problemas dentro del software. Detecta código duplicado, código muerto (variables, pará-metros o métodos sin usar), complejidad de métodos (if innecesarios, etc.,). Trabaja principalmente con lenguaje Java, aunque, con menos soporte, también posee conjuntos de reglas para JavaScript, xsl y ecmascript.	http://pmd.sourceforge.net/
Check Style	Herramienta de análisis estático de código que se utiliza para comprobar que el código analizado cumple con una serie de reglas de estilo. Ejemplo, analiza el código según el estándar ñSun Code Conventionsò (mira las cabeceras, importaciones de paquetes, Javadoc, etc.). Trabaja para Java.	Software Libre http://checkstyle.sourceforge.net/
Google CodePro Analytix	Otra de las herramientas de calidad software, ofrece un entorno para evaluación de código, métricas, análisis de dependencias, cobertura de código, generación de Test unitarios, etc. Mira las excepciones, refactorizaciones potenciales, convenios de JavaDoc, métricas, etc.	Software Libre http://code.google.com/intl/es-ES/javadevtools/codepro/doc/index.html

	Disponible como plugin de Eclipse. Trabaja para Java, concretamente en Eclipse.	
Simian	Herramienta para detectar código duplicado en desarrollos realizados con los lenguajes: Java, C#, C, C++, COBOL, Ruby, JSP, ASP, HTML, XML y Visual Basic. La licencia es libre si su uso está destinado a proyectos OpenSource.	http://www.redhillconsulting.com.au/products/simian/
Ndepend	Herramienta de Métricas para .Net . Permite obtener distintas métricas a distintos niveles (ensamblados, namespaces, tipos, etc.) y representa gráficamente varios aspectos de las aplicaciones analizadas (matriz y grafo de dependencias, tree maps de las métricas calculadas).	http://www.ndepend.com/
HP Unified Functional Testing	HP Unified Functional Testing era antes conocida como Quick Test Professional, QTP. Soporta una variedad muy extensa de tecnologías: Java, Sap, Siebel, Visual Basic .Net y Oracle entre muchas otras. Esta Herramienta se basa en el reconocimiento de objetos, aunque se	HP http://www8.hp.com

	puede utilizar el posicionamiento dentro de una pantalla para la realización de pruebas, así como reconocimiento de texto por OCR. Al poseer un interfaz amigable y un código de generación de scripts en visual basic se consigue que se aprenda más fácilmente, obteniendo una curva de aprendizaje muy alta desde el primer momento. Esta herramienta se suele integrar con todas las herramientas de la suite de HP como Quality Center (gestor de requisitos, casos de prueba y defectos). También hay que decir que al estar una empresa detrás como HP, el soporte es muy amplio y hay gran cantidad de ingenieros de calidad para su soporte dando fiabilidad y eficiencia[34].	
TestLink	TestLink permite fácilmente crear y gestionar casos de prueba, así como organizarlos en planes de pruebas. Estos planes de pruebas permitir a los miembros del equipo ejecutar casos de prueba y realizar un seguimiento de los resultados	Software Libre http://www.testlink.org

[34] HP Quality Center, "Descripción General", en http://www8.hp.com/es/es/software-solutions/software.html?compURI=1172141

	de la prueba en forma dinámica, generar informes, traza de requisitos de software, priorizar y asignar tareas. La herramienta tiene interfaz basada en web con PHP y base de datos MySQL, Postgres o MS-SQL. Colabora con conocidos sistemas de seguimiento de bugs como es Bugzilla, Mantis, etc.	
Jmeter	Herramienta para la realización de pruebas de rendimiento. Simula el ingreso de usuarios periódicos y permite hacer pruebas de carga y de stress. Lleva a cabo simulaciones sobre cualquier recurso de Software.	Software Libre http://jakarta.apache.org/jmeter
FITNESSE	Herramienta colaborativa para la realización de pruebas funcionales, orientadas a las pruebas de aceptación del cliente. Permite escribir las pruebas de aceptación en formato wiki o HTML y su posterior aplicación a la lógica de negocio del producto software.	Software Libre http://fitnesse.org
Microsoft Test Manager	Microsoft Test Manager (MTM) es la herramienta propiedad de Microsoft para la	Microsoft http://www.microsof

	gestión y automatización de pruebas. Esta herramienta está incluida en Microsoft Visual Studio Ultimate 2010 o en Visual Studio Test Professional 2010. El interfaz y el código generado en los scripts es bastante intuitivo, se debe de integrar con Team Foundation Server que almacena los casos de prueba y requerimientos entre otras cosas. El código generado se llama coded UI que graba operaciones de interfaz basado en Visual C#.NET. Además se pueden ejecutar las pruebas automáticas tanto en máquinas virtuales como físicas. Se instala en sistemas operativos Windows[35].	t.com/
TestComplete	Es una herramienta orientada a objetos que soporta una gran cantidad de tecnologías tales como Visual Basic, Delphi, C + + y otras herramientas de desarrollo. Se puede ejecutar en los navegadores Internet Explorer, Mozilla Firefox y Google Chrome en sus versiones de 32 y 64 bits y soporta	SmartBear software http://www.smartbear.com

[35] Arturo Fernández, "Comparativa de herramientas para pruebas automáticas", en http://www.globetesting.com/2012/03/comparativa-de-herramientas-para-pruebas-automaticas/

	flash y otros complementos. Por el momento sólo ofrece soporte en Windows.	
Junit	Es un framework de código abierto desarrollado especialmente para crear, ejecutar y hacer reportes de estado de conjuntos de Prueba Unitaria automatizadas hechos en lenguaje Java. JUnit es uno de los frameworks más populares en Java para realizar pruebas unitarias y llevar un desarrollo utilizando la práctica de Test Driven Development.	Software Libre http://github.com/kentbeck/junit/wiki
Gestión de Proyectos[36]		
Nombre	**Características**	**Vendedor**
Redmine	Es un gestor y planificador de proyectos con interfaz web, orientado a la coordinación de tareas, comunicación de participantes y que puede especializarse en proyectos de desarrollo gracias a herramientas como la integración en un repositorio de código.	Software Libre www.redmine.org

[36] Fábricas de software, ñHerramientas de Desarrollo de Software ï Gestión de Proyectosò, en http://www.fabricasdesoftware.es/herramientas

Trac	Es una herramienta para la gestión de proyectos y el seguimiento de errores, escrita en Python[37].	Software Libre http://trac.edgewall.org/
DotProject	Herramienta que permite gestionar las distintas fases y tareas que componen un proyecto. A menudo, esta gestión implica un control en recursos humanos, materiales, que hacen que esta labor se torne compleja y prácticamente inabordable sin la ayuda de determinadas herramientas que den soporte a esta tarea de planificación y gestión de proyectos[38]. Dotproject se perfila como una herramienta para trabajar en entornos colaborativos, permitiendo a los integrantes del equipo trabajar compartiendo información relativa a los proyectos Basado en plataforma web permite la participación online de los miembros de un proyecto.	Software Libre http://www.abartiateam.com/dotproject

[37] Trac Integrated SCM & Project Management, "Trac Open Source Project", en http://trac.edgewall.org/

[38] Abartia Team, "Gestión de Proyectos con DotProject", en http://www.abartiateam.com/dotproject

Jira	Jira es una aplicación para la administración de proyectos y actividades desarrollada para facilitar el trabajo de su equipo. Jira es una tecnología basada en el estándar J2EE. Facilita la evolución a un sistema de procesos modelable y estructurado[39].	Software Libre http://www.atlassian.com/es/software/jira
TFS + Microsoft Project	Team Foundation Server es un sistema de control de versiones de código fuente y de gestión del ciclo de vida de la aplicación, desde la fase de diseño hasta las pruebas, pasando por la integración continua o la calidad del código. TFS incorpora varios sistemas integrados: por un lado una base de datos en SQL Server que contiene no sólo el código fuente de nuestras aplicaciones sino los elementos de trabajo que posibilitan el seguimiento del desarrollo; los datos se integran en un Data Warehouse de SQL Server Analysis Services que proporciona información sobre el estado del proyecto	Microsoft http://www.microsoft.com

[39] Atlassian, "Algunas de las maravilla de JIRA", en http://www.atlassian.com/es/software/jira

	mediante informes en Reporting Services o Excel; el motor de compilación Team Build permite la compilación desatendida de los proyectos y genera informes de calidad de la compilación; y todo ello se puede integrar en portales de colaboración de Microsoft SharePoint para que todo el equipo pueda compartir información, documentos o calendarios asociados al proyecto[40].	
TeamWork	Es una sencilla herramienta que permite la gestión de proyectos y sus relaciones con los clientes.	Software Libre http://www.teamworkpm.net/
Achievo	Es una herramienta que cuenta con calendarios, gestión de tareas y división según el avance en su ejecución, estadísticas, plantillas de proyectos y notas. Se encuentra disponible en más de 20 idiomas[41].	Software Libre http://www.achievo.org/

[40] Microsoft, "Team Foundation Server - Detalles del Producto", en http://www.microsoft.com/visualstudio/esn/products/visual-studio-team-foundation-server-2012#product-edition-tfs-details

[41] Archievo Org, "Achievo Flexible web-based project management", en http://www.achievo.org/

Gantt Project	Es una herramienta de escritorio multiplataforma totalmente gratuito. Incluye diagramas de Gantt, asignación de las personas que trabajarán en el proyecto, y permite exportar los diagramas como imágenes, mientras genera informes en PDF y HTML. Permite interoperar con Microsoft Project, importando y exportándolos a sus formatos.	Software Libre http://www.ganttproject.biz/
TaskJuggler	Es un gestor de proyectos realmente potente y superior a otros que usan herramientas para editar diagramas de Gantt. Cubre todos los aspectos de desarrollo de un proyecto, desde la primera idea hasta su fin. Ayuda a medir su campo de alcance, asignación de recursos, esquema de costos y ganancias, riesgo y gestión de las comunicaciones.	Software Libre http://www.taskjuggler.org/