# Universidad de las Ciencias Informáticas
# Facultad 3


# Título: Propuesta de Mecanismo de Gestión de Calidad Interna para el proyecto Registro y Notarías

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

**Autores:** Lídice Delgado Alón

Heney Díaz Pérez

**Tutor:** Ing. Michael González Jorrín

**Junio 2007**

DECLARACIÓN DE AUTORÍA

Declaro que soy el único autor de este trabajo y autorizo al _____ de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

Autores:

Lídice Delgado Alón

_____

Heney Díaz Pérez

_____

Tutor:

Ing. Michael González Jorrín

_____

AGRADECIMIENTOS

Queremos agradecer en primer lugar por la oportunidad de haber cursado estudios universitarios en un programa de la Batalla de Ideas que impulsa nuestro Comandante y por el privilegio de pertenecer al Destacamento “Tropa de Futuro”. Agradecer por su sueño hecho realidad que es la UCI.

A nuestra familia por todo el apoyo y el sacrificio de estos cinco años… y por los que vienen.

A nuestros profesores todos, por contribuir a nuestra formación profesional.

A los amigos que nos apoyaron en todo momento, a los distantes, a los cercanos, a los de siempre.

## RESUMEN

La Propuesta de un Mecanismo de Gestión de Calidad Interna para el Proyecto Registro y Notarías está orientada a identificar el proceso de desarrollo de software establecido, determinando las actividades que se realizan, con los roles involucrados en la realización de cada una de esas actividades y definiendo las entradas y salidas, para la segunda fase de este proyecto. Una vez precisado el objeto sobre el que se centra la atención de la gestión, o sea, el proceso, se definen y describen las acciones concretas que permitirán actuar sobre el objeto y obtener resultados que permitan tener criterio valorativo sobre la calidad como característica. Para esto, se elabora un Plan de Aseguramiento de Calidad en el que se determinan, como paso esencial, los objetivos que se persiguen con la gestión de calidad interna en perfecta concordancia con los intereses de la dirección del proyecto, se refleja claramente la finalidad de la gestión y el radio de acción sobre el que se va a desarrollar la influencia. Se dedica también una parte importante a la organización del equipo gestor de calidad, que se encargará de desarrollar el mecanismo de gestión de calidad interna del proyecto, definiendo los roles, las responsabilidades y la base de conocimientos necesaria para su desempeño, proponiéndose además un Plan de Capacitación para la formación de dicho equipo.



## PALABRAS CLAVE

Calidad, Gestión, Procesos, Proyecto, Software, Plan

TABLA DE CONTENIDOS

ÍNDICE DE FIGURAS

# Introducción

La producción de software en la UCI es una actividad que centra la atención de todos en la comunidad universitaria. Constantemente se están contrayendo importantes acuerdos con clientes nacionales e internacionales que hacen que el compromiso para con la calidad del producto sea muy alto.

En este sentido, surge la necesidad de personal calificado en los temas de calidad, gestión e ingeniería de software y de investigar y automatizar procesos para el control de calidad y dar seguimiento a procesos de desarrollo de software. La universidad se dio a la tarea de controlar la calidad del producto de software. Surgen así el Grupo de Investigación de Calidad e Ingeniería de Software de la UCI, la Dirección de Calidad de la Infraestructura Productiva (IP) y el Laboratorio de Certificación.

El grupo de investigación se dedica a indagar sobre temas referentes a los procesos de desarrollo de software, metodologías y calidad. A partir de los resultados de la investigación que se realizan en el grupo, la Dirección de Calidad de la IP dicta las políticas por la que se deben regir los proyectos que tienen como eje fundamental las distintas facultades. Al mismo tiempo, la Dirección de Calidad se encarga de dirigir y llevar a cabo la planificación y ejecución del control de la calidad a que deben ser sometidos todos los proyectos de desarrollo de software y aplica los principios de las revisiones e inspecciones con vistas a la mejora continua de los productos a obtener, ya sea de la aplicación o de toda la documentación asociada al producto a través del trabajo del Laboratorio de Certificación.

A pesar de estos esfuerzos, en los proyectos productivos de la universidad se siguen presentando problemas con la calidad. En el proyecto Registro y Notarías, a pesar de estar definido en su línea base de la arquitectura, no se encontraba activo un equipo gestor de calidad. Al no existir un aparato que programe y organice el control de calidad sistemático, el desarrollador no se ve comprometido con la calidad y por tanto, en las tareas implementadas son abundantes los errores desde el análisis de la solución, que se traduce luego en defectos que atentan contra la calidad del sistema. Como consecuencia de esto, durante las Pruebas de Aceptación del Cliente se detectaron un número significativo de defectos. En la corrección de estos defectos, se empleó un esfuerzo considerable no sólo del equipo del proyecto sino también del Laboratorio de Certificación. El tratamiento de la calidad en el proyecto se redujo a resolver los errores, certificar los productos por el laboratorio y someterlos a pruebas de aceptación. Esta actividad involucró mucho tiempo y provocó un fuerte retraso en la planificación.

Teniendo en cuenta la situación anterior, se define el siguiente problema: La ausencia de una adecuada gestión de calidad interna del proyecto Registro y Notarías provoca que desde el análisis mismo de la solución sean abundantes los errores que se traducen luego en defectos, que atentan contra la calidad del sistema.

El objeto de estudio que se plantea es la gestión de calidad en los proyectos de desarrollo de software.

El campo de acción que se propone es la gestión de calidad interna del proyecto Registros y Notarías.

El objetivo que se persigue es desarrollar un mecanismo de gestión de la calidad interna para el proyecto Registros y Notarías.

La hipótesis de la que se parte es la siguiente: Con una adecuada gestión de calidad interna del proyecto Registro y Notarías, se reducirían los problemas con la calidad del software.

Entre las tareas que se propone desarrollar para el cumplimiento del objetivo se señalan:

- Estudiar sobre las normas y estándares internacionales de calidad de software y su adaptación a las realidades concretas del proyecto Registro y Notarías.
- Estudiar sobre los roles y responsabilidades que propone la metodología RUP (Racional Unified Process) para el control de calidad a lo largo del proyecto de desarrollo de software.
- Realizar un análisis de cómo se controlaba la calidad del desarrollo de software hasta la fecha en el proyecto.
- Analizar la propuesta de desarrollo de software para el proyecto Registro y Notarías en su segunda fase.
- Definir la organización del proyecto Registro y Notarías en su segunda fase para la gestión de la calidad.
- Desarrollar un plan de calidad que conciba las actividades a realizar para garantizar la gestión de calidad interna del proyecto Registro y Notarías en su segunda fase.

- Analizar el cronograma de desarrollo del proyecto Registro y Notarías en su segunda fase para planificar las pruebas y las revisiones.
- Proponer un mecanismo para llevar a cabo un control y seguimiento de defectos detectados en las pruebas y revisiones.
- Proponer una organización para el equipo de gestión de calidad interna.
- Proponer una capacitación para los miembros del equipo de calidad para lograr una nivelación de los conocimientos con vistas a desempeñar las tareas y actividades de gestión.
- Definir las actividades de gestión de calidad partiendo de la propuesta de desarrollo de software.
- Elaborar y proponer un mecanismo de gestión de calidad interna para el proyecto Registro y Notarías en su segunda fase.

En el Capítulo 1 se realiza un análisis del concepto de calidad partiendo de sus antecedentes y centrándonos en la industria de software. Se realiza una síntesis sobre los modelos y estándares de calidad aplicados al desarrollo de software. Se concluye con un análisis de la gestión de calidad a diferentes niveles haciendo énfasis en la organización de un grupo de calidad, así como la planificación y el control de la calidad del software.

En el Capítulo 2 se describen las características de la solución que se propone, partiendo de la identificación del proceso de desarrollo de software en la industria de modo general y analizando las peculiaridades de dicho proceso en el proyecto Registro y Notarías hasta el momento. Posteriormente se realiza una propuesta de identificación de los objetivos de la gestión de calidad interna, así como las actividades y tareas enfocadas a lograr dichos objetivos. Se dedica además una parte importante a la organización del equipo de calidad interna del proyecto. En otro momento se analiza la propuesta preliminar de organización del proceso de desarrollo para Registro y Notarías en su segunda fase y se tiene en cuenta los lineamientos propuestos por la Dirección de Calidad de la IP para los proyectos productivos en las facultades.

En el Capítulo 3 se realiza la propuesta de solución, básicamente centrada en el Plan de Aseguramiento de Calidad donde se formulan todos los documentos y todos los elementos importantes que tributan a la gestión de calidad interna del proyecto.

Entre los Métodos de trabajo científico utilizados se destacan los siguientes:

Métodos teóricos:

- Histórico - Lógico: Para determinar las necesidades históricas y tendencias actuales de la gestión de la calidad en la elaboración de software.
- Análisis y Síntesis: Permitió, profundizar en la esencia del fenómeno objeto de estudio, así como realizar una valoración acerca de los modelos, estándares y procedimientos de gestión de calidad existentes que nos permitiese ajustar de una forma coherente nuestra propuesta de solución.
- Inducción-deducción: Posibilitó llegar a conclusiones acerca del problema investigado, a partir del estudio de la bibliografía consultada y del resultado de los instrumentos utilizados.

Métodos empíricos:

- Entrevista: Se realizó a miembros de la dirección del proyecto con el objetivo de dilucidar las peculiaridades del proceso de desarrollo y la selección de métricas y guías para una gestión de la calidad interna enfocada a las necesidades concretas del proyecto.

# Capítulo 1: Fundamentación Teórica

## *1.1. Introducción*

En el presente capítulo se pretende sentar las bases que fundamentan el desarrollo de la tesis. Se parte de un análisis del concepto de Calidad de Software que va desde sus antecedentes hasta su definición. Dicho análisis se ve desde una perspectiva histórica estudiando cada una de las etapas que caracterizan su desarrollo. Se estudia además el concepto planteado por múltiples autores que constituye el soporte para la determinación de la definición afín al objeto de estudio propuesto.

Se realiza además un acercamiento al conocimiento de los Modelos y Estándares de Calidad a nivel de proceso y de producto, útiles para el posterior análisis de la Gestión de Calidad del Software, así como su planificación y control, haciendo énfasis en la organización de un grupo de calidad.

## *1.2. Calidad de Software. Antecedentes. Definición.*

Mirando la Calidad como concepto desde una perspectiva histórica, podemos decir según (2) que es un concepto cuya evolución viene marcada por varias etapas. La primera Etapa, la del control de la calidad mediante la Inspección (Siglo XIX), se caracterizó por la detección y solución de los problemas generados por la falta de uniformidad del producto.

Durante esta fase, se consideró que la inspección era la única manera de asegurar la calidad, reflejándose esto en el pensamiento y la literatura técnica de la época. La ejecución de la práctica se orientó a tareas tales como la selección y clasificación de los productos, el rescate de productos de lotes dañados, reprocesamiento, la ejecución de mezclas para salvar materias primas con daños leves, la toma de acciones correctivas y la búsqueda de las fuentes de no conformidad.

La segunda Etapa, definida por la era del control estadístico del proceso (década de los 30’s) estaba enfocada al control de los procesos y a la aparición de métodos estadísticos para el mismo fin y la reducción de los niveles de inspección. En esta fase, según (3), existe ya un método de calidad, siendo la inspección una parte del Control de Calidad. La filosofía y la práctica del Control de Calidad se orienta al desarrollo de manuales de calidad, la recolección de información sobre el comportamiento de los procesos, utilización de la estadística básica en control de calidad, ejecución del autocontrol, análisis y

ensayos de materias primas, de productos en proceso y productos terminados, se establecen los procedimientos para la elaboración, control y difusión de informes. Aparece una planificación básica de control de calidad.

La tercera Etapa, la del aseguramiento de la calidad (década de los 50's), está dada por el surgimiento de la necesidad de involucrar a todos los departamentos de la organización en el diseño, plantación y ejecución de políticas de calidad. En esta época de Aseguramiento de la Calidad, la filosofía y la práctica de la calidad cambian notablemente y, es la primera vez, que el enfoque no es sólo hacia la inspección y control de calidad, sino que ahora se concentra en que los mismos productos cumplan con sus especificaciones, a través de un sistema de calidad definido, y una planificación orientada a la calidad y utilización de los costos de calidad. Aparecen manuales de calidad comprensibles, hay un control estadístico del proceso, y se inicia la participación de algunas operaciones de no producción y del análisis de causa y efecto.

La cuarta Etapa está determinada por la administración estratégica de la calidad total (década de los 90's) donde se hace hincapié en el mercado y en las necesidades del consumidor, reconociendo el efecto estratégico de la calidad, como una oportunidad de competitividad. Es llamada también Gerencia de la Calidad Total o el TQM (Total Quality Management). Es una práctica gerencial para el mejoramiento continuo de los resultados en cada área de actividad de la empresa y en cada uno de los niveles funcionales, utilizando todos los recursos disponibles y al menor costo. El proceso de mejoramiento se orienta hacia la satisfacción completa del consumidor, considerándose al recurso humano como el más importante de la organización.

Según (2), esta etapa se extiende por la reingeniería de procesos, donde el avance tecnológico y de sistemas administrativos propone un mejoramiento radical, empezar de nuevo, cambiar toda la organización, rearquitectura de la empresa y rompimiento de las estructuras del mercado (a finales del siglo XX y principios del XXI), donde se propone que el conocimiento es la base de los negocios actuales.

La palabra calidad tiene varios significados, aunque dentro de la Ingeniería del Software podemos adoptar la definición de la norma ISO-8402, que luego se repite en otras (por ejemplo en ISO-14598): "La totalidad de aspectos y características de un producto o servicio que tienen que ver con su habilidad para satisfacer las necesidades declaradas o implícitas". La calidad es un objetivo importante para cualquier

producto pero no debemos olvidar que los productos (incluidos los software) tienen como finalidad ser usados, por tanto, el principal objetivo de un producto es satisfacer necesidades de un usuario. Es decir, la calidad no es el objetivo final cuando elaboramos un producto, sino satisfacer las necesidades de un cliente. También es importante señalar que esto implica que la calidad de un producto software no se puede referir únicamente a obtener un producto sin errores. La especificación de la calidad del software debe ser más detallada y exacta.

Para el acercamiento a este complejo tema se ha querido referir el interesante enfoque de Garvín que esboza la visión de la calidad desde cinco perspectivas: (I) la visión trascendental que puede ser reconocida pero no definida, (II) la visión del usuario como la adecuación al propósito del usuario, (III) la visión del productor como conformidad con la especificación, (IV) la visión del producto, basada en las características observables del producto, y (V) la visión basada en el valor que el cliente está dispuesto a pagar.

La calidad del software debe ser considerada en todos sus estados de evolución, es importante diferenciar entre la calidad del producto software y la calidad del proceso de desarrollo. Las metas que se tracen en relación a la calidad deseada del producto final definirán los niveles de exigencia necesarios para implementar un proceso de desarrollo de software con calidad. Sin un buen proceso de desarrollo es casi imposible obtener un buen producto.

Es importante subrayar que el aseguramiento de la calidad del software no se puede dar si no existe un control que garantice el cumplimiento de las fases o etapas que se han definido previamente en una metodología, y sin una constante supervisión de todo el proceso de desarrollo; esto solo podrá lograrse con la continua evaluación de la calidad que se alcance en cada etapa del proceso antes de continuar adelante.

Los principales problemas a los que hay que enfrentarse al hablar de la calidad de un producto de software están dados por la definición misma de la calidad del software, la comprobación de la calidad del software y la mejora de la calidad del software.

Para definir la calidad del software surgen los *Modelos de Calidad*, donde la calidad se puntualiza de forma jerárquica y se resuelve la complejidad mediante la descomposición (4). Para la comprobación

de la calidad aparece el concepto de *Control de la Calidad*, el que define las técnicas y actividades de carácter operacional para satisfacer los requisitos relativos a la calidad. Se orienta a mantener bajo control los procesos y a eliminar las causas que generan comportamientos insatisfactorios en etapas importantes del ciclo de calidad para conseguir mejores resultados económicos. (3). La *mejora de la calidad* del software está determinada por cómo utilizar la información disponible acerca de la calidad del producto de software para mejorar su calidad a lo largo del ciclo de vida. Es posible "construir" la calidad y en esa dirección surge el concepto de *Gestión de Calidad* y el de *Garantía o Aseguramiento de la Calidad*. La Gestión de Calidad es aquel aspecto de función general de la gestión de una organización que define y aplica la política de calidad. La gestión de la calidad incluye la planificación, las asignaciones de recursos y otras actividades sistemáticas, tales como los planes de calidad. Por su parte, la Garantía o Aseguramiento de la Calidad son todas aquellas acciones planificadas y sistemáticas necesarias para proporcionar la confianza adecuada de que un producto o servicio satisface los requisitos de calidad establecidos. Para que sea efectivo, el aseguramiento de la calidad requiere, generalmente, una evaluación permanente de aquellos factores que influyen en la adecuación del diseño y de las especificaciones según las aplicaciones previstas, así como también verificaciones y auditorías a las operaciones de producción, instalación e inspección. Dentro de una organización, el aseguramiento de la calidad sirve como una herramienta de la gestión. (3)

## *1.3. Modelos y Estándares de Calidad.*

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.

La ventaja de los Modelos y/o Estándares de Calidad es que la calidad se convierte en algo concreto, que se puede definir, que se puede medir y, sobre todo, que se puede planificar. Los Modelos

y/o Estándares de Calidad ayudan también a comprender las relaciones que existen entre las diferentes características de un producto de software. Una desventaja es que aún no ha sido demostrada la validez absoluta de ninguno de estos Modelos o Estándares. Las conexiones que se establecen entre características, atributos y métricas se derivan de la experiencia. Esto origina que existan múltiples Modelos y Estándares de Calidad. En este epígrafe se hace referencia a algunos de estos basándonos en el compendio realizado en el trabajo investigativo “Estudio comparativo de los Modelos y Estándares de calidad del software” (1)

*Modelos de Calidad del Software a Nivel de Proceso.*

*El Modelo de Madurez de Capacidad para el Software (CMM, siglas en inglés)*

Es un marco que describe los elementos clave de un proceso de software eficaz, describe una trayectoria de mejora evolutiva hasta llegar a un proceso maduro y disciplinado; incluye prácticas de planificación, ingeniería y gestión del desarrollo y mantenimiento de software. El CMM también puede utilizarse por una organización para planificar mejoras en su proceso de software. Está formado por cinco niveles, cada uno se orienta a un aspecto específico y explica las áreas clave del proceso que se deben considerar.

Estructura del CMM.

CMM está compuesto por cinco niveles de madurez que a excepción del Nivel 1, cada nivel de madurez está compuesto por diversas áreas claves del proceso; estas áreas claves del proceso están organizadas en secciones denominadas características comunes que las prácticas especifican que cumplen los objetivos del área clave de proceso.

Niveles de madurez del CMM.

A medida que las organizaciones establecen y mejoran los procesos de software mediante los cuales desarrollan y mantienen sus productos de software, van avanzando a través de los niveles de madurez. La siguiente figura muestra los cinco niveles de madurez del CMM.

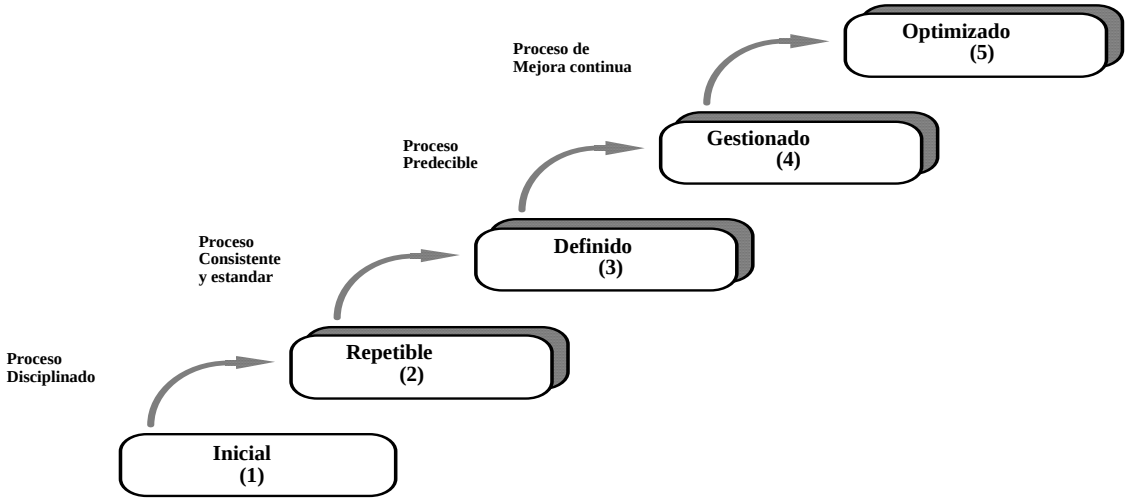



**FIG. 1 NIVELES DE MADUREZ DE CMM**

Nivel 1 (El nivel inicial).

En el Nivel Inicial, la organización no suele proporcionar un entorno, se estable para el desarrollo y el mantenimiento del software. El éxito depende exclusivamente de disponer de un director excepcional y de un equipo software sensato y eficaz. La capacidad de proceso de software en las organizaciones de Nivel 1 es impredecible, ya que el proceso software es constantemente cambiado o modificado a medida que avanza el trabajo. Los calendarios, los presupuestos, la funcionalidad y la calidad del producto son impredecibles. El rendimiento depende de las capacidades de los individuos y varía según sus habilidades naturales, conocimientos y motivaciones.

Nivel 2 (El nivel repetible).

En el Nivel Repetible, se establecen las políticas para la gestión del proyecto y los procedimientos para implementarlas. Uno de los objetivos de este nivel es institucionalizar unos procesos de gestión que permitan a las organizaciones repetir las prácticas desarrolladas con éxito en proyectos precedentes; se identifican los problemas que surgen para cumplir los compromisos adquiridos; se establece una línea base de los requisitos software y de los productos desarrollados para satisfacerlos, controlándose su integridad; se definen los estándares del proyecto de software.

Nivel 3 (El nivel definido).

En el Nivel Definido, se documenta el proceso estándar de desarrollo. Los proyectos adaptan el proceso software estándar de la organización para desarrollar su propio proceso software definido. Un proceso bien definido se caracteriza por incluir criterios de disponibilidad, entradas, estándares y procedimientos para realizar el trabajo, mecanismos de verificación, salidas y criterios de terminación. Dentro de las líneas de producto establecidas, el coste, el calendario y la funcionalidad se encuentran bajo control y se realiza un seguimiento de la calidad del software.

Nivel 4 (El nivel gestionado).

En el Nivel Gestionado, la organización establece objetivos cuantitativos de calidad para los productos y procesos software. Se miden la productividad y la calidad de las actividades importantes del proceso software parte de un programa de medidas propio de la organización; se utiliza una base de datos del proceso software que abarque a toda la organización para recoger y analizar los datos disponibles a partir de los procesos software definidos de los proyectos. Este nivel permite predecir las tendencias en el proceso y en la calidad del producto. Se usan métricas para gestionar la organización.

Nivel 5 (El nivel optimizado).

En el Nivel Optimizado, toda la organización se centra en la mejora continua del proceso; se evalúan los procesos software para impedir que se produzcan los tipos de defectos conocidos, y las lecciones aprendidas se difunden a otros proyectos. La capacidad del proceso software de las organizaciones del Nivel 5 se puede caracterizar como de mejora continua. Este nivel incluye mejoras incrementales e innovadoras de los procesos que mediante métricas son identificadas, evaluadas y puestas en práctica.

Desventajas del modelo.

Este modelo presenta algunas desventajas:

- Es un modelo extranjero, no internacional.
- No es fácil de entender (inglés, 220 páginas).
- No es fácil de aplicar (pensado para organizaciones grandes).
- La mejora no está enfocada directamente a los objetivos de negocio.
- La evaluación es costosa y no tiene periodo de vigencia.
- Se está abandonando a favor de CMMI.

*Integración del Modelo de Madurez de Capacidad para el Software (CMMI, siglas en inglés).*

CMMI se desarrolló para facilitar y simplificar la adopción de varios modelos de forma simultánea integrando a sus predecesores: CMM-SW (CMM for Software), SE-CMM (Systems Engineering Capability Maturity Model), IPD-CMM (Integrated Product Development).

CMMI es el resultado de la publicación del modelo CMM con dos representaciones: contínua y escalonada, ambas son equivalentes. La visión continua de una organización mostrará la representación de nivel de capacidad de cada una de las áreas del proceso del modelo. La visión escalonada definirá a la organización dándole en su conjunto un nivel de madurez del 1 al 5 (niveles).

Áreas de proceso.

CMMI tiene 22 áreas de procesos que ayudan a evaluar el desarrollo de software de ingeniería de sistemas (CMMI-SE/SW) y 25 en la que cubre también integración de producto (CMMI-SE/SW/IPPD). La siguiente tabla muestra las áreas de proceso de CMMI.

| **Áreas de proceso de CMMI (Capability Maturity Model Integration)** | | |
|---|---|---|
| **Área de proceso** | **Categoría** | **Nivel de Madurez** |
| Análisis y resolución de problemas. | Soporte | 5 |
| Gestión de la configuración. | Soporte | 2 |
| Análisis y resolución de decisiones. | Soporte | 3 |
| Gestión integral de proyecto. | Gestión de proyectos | 3 |
| Gestión integral de proveedores. | Gestión de proyectos | 3 |
| Gestión de equipos. | Gestión de proyectos | 3 |
| Medición y análisis. | Soporte | 2 |
| Entorno organizativo para integración. | Soporte | 3 |
| Innovación y desarrollo. | Gestión de procesos | 5 |
| Definición de procesos. | Gestión de procesos | 3 |
| Procesos orientados a la organización. | Gestión de procesos | 3 |
| Rendimiento de los procesos de la organización. | Gestión de procesos | 4 |

| | | |
|---|---|---|
| Formación. | Gestión de procesos | 3 |
| Integración de producto. | Ingeniería | 3 |
| Monitorización y control de proyecto. | Gestión de proyectos | 2 |
| Planificación de proyecto. | Gestión de proyectos | 2 |
| Gestión calidad procesos y productos. | Soporte | 2 |
| Gestión cuantitativa de proyectos. | Gestión de proyectos | 4 |
| Desarrollo de requisitos. | Ingeniería | 3 |
| Gestión de requisitos. | Ingeniería | 2 |
| Gestión de riesgos. | Gestión de proyectos | 3 |
| Gestión y acuerdo con proveedores. | Gestión de proyectos | 2 |
| Solución técnica. | Ingeniería | 3 |
| Validación. | Ingeniería | 3 |
| Verificación. | Ingeniería | 3 |

Las mejoras que aporta el modelo CMMI son:

- Desarrolla un marco de actuación para permitir el crecimiento de otras disciplinas.
- Nuevo énfasis sobre el producto, así como sobre los procesos, incluyendo las interacciones con el cliente.
- Mayor importancia, desde las fases iniciales, del análisis y la medición de los procesos empresariales.
- Cobertura de servicios, así como de sistemas.
- Especial énfasis sobre la capacidad de los procesos y madurez de la organización en su conjunto (no exclusivamente en el área de ingeniería del software).
- Mejor cobertura de la gestión de ingeniería integrada.
- Énfasis sobre las mejoras medibles y cuantificables para alcanzar los objetivos del negocio empresarial.
- Existe un nuevo enfoque de la formación. La educación y el entrenamiento adecuado para la mejora de la eficacia y de la eficiencia.
- Favorece el establecimiento de un ambiente adecuado para la gestión de los cambios dentro de la organización.

- Proporciona compatibilidad con los principios, requisitos y recomendaciones de la norma ISO 9000:2000.
- Sienta las bases para que las organizaciones del sector de desarrollo del software se encaminen hacia el ciclo de la mejora continua.

### *TSP (Team Software Process)*

El objetivo del TSP (Team Software Process) es construir y guiar a los equipos. Los objetivos de TSP son: 1) ayudar a los equipos de Ingeniería de Software a elaborar productos de calidad dentro de los costos y tiempos establecidos, 2) tener equipos rápidos y confiables; y 3) optimizar el performance del equipo durante todo el proyecto.

TSP es una manera de guiar a los Ingenieros y a sus Gerentes en la utilización de métodos de trabajo en equipos efectivos. En TSP, el principal énfasis de la calidad está en el manejo de defectos.

Para administrar la calidad, los equipos deben: 1) establecer medidas de calidad, 2) determinar objetivos de calidad, 3) establecer planes para alcanzar los objetivos, 4) medir el progreso de los planes y 5) efectuar una acción de recuperación cuando no se alcanzan los objetivos. Los elementos del manejo de la calidad en TSP son: 1) realizar un plan de calidad, 2) identificar los problemas de calidad; y 3) encontrar y prevenir los problemas de calidad.

### *PSP (Personal Software Process)*

El Personal Software Process (PSP) es un proceso de software definido para ser usado individual por medio de un Ingeniero de Software. Es una tecnología de SEI (Software Engineering Institute) que trae disciplina a las prácticas de los Ingenieros de Software, mejorando la calidad del producto, aumentando los costos y reduciendo el tiempo del ciclo de desarrollo del software. PSP está basado en los principios de mejoramiento del proceso. CMM está enfocado respecto del mejoramiento de la capacidad organizacional.

El proceso de PSP consiste de un conjunto de métodos, formularios y escrituras que muestran a los Ingenieros de Software cómo planificar, medir y administrar su trabajo. El PSP provee una estructura

acerca del proceso de software y un punto de partida con el cual desarrollar sus propios procesos personales.

### *Modelos de Calidad del Software a Nivel de Producto:*

#### *Modelo de Gilb*

El modelo de Gilb plantea la creación de una especificación de requisitos de calidad para cada proyecto que deben escribir conjuntamente el usuario y el analista. Es un modelo que permite determinar una lista de características que definen la calidad de la aplicación. Puede ser de 2 tipos: 1) Originales y 2) de modelos tradicionales.

Las características se pueden medir mediante varias subcaracterísticas o métricas detalladas. Para cada una de ellas, se deben especificar los siguientes conceptos: 1) nombre y definición de la característica, 2) Escala o unidades de medición, 3) recopilación de datos o prueba, 4) valor previsto, 5) valor óptimo, 6) valor en el sistema actual y 7) comentarios.

En la década del ochenta, se comenzaron a usar modelos particulares de evaluación para cada empresa o proyecto, implantándose el concepto de calidad relativa. Este enfoque se ha asociado a la filosofía QFD (Quality Function Deployment), o el despliegue de la función de la calidad, que se aplica al ámbito de la gestión de la calidad industrial y en el que se han basado modelos posteriores.

#### *McCall*

El modelo de McCall se basa en 11 factores de calidad, que se organizan de la siguiente forma teniendo en cuenta la visión del usuario:

| **Visión del usuario** | **Factores de Calidad según McCall** |
|---|---|
| Operación del producto | Facilidad de Uso – Integridad – Corrección – Confiabilidad – Eficiencia |
| Revisión del producto | Facilidad de mantenimiento – Facilidad de prueba – Flexibilidad |
| Transición del producto | Reusabilidad – Interoperabilidad - Portabilidad |

Características Operativas

Corrección: Es el grado en que un programa satisface sus especificaciones y consigue los objetivos pedidos por el cliente.

Confiabilidad: Es el grado en que se puede esperar que un programa lleve a cabo sus funciones esperadas con la precisión requerida.

Eficiencia: Es la cantidad de recursos de computadoras y de código requeridos por un programa para llevar a cabo sus funciones.

Integridad: Consiste en la capacidad de un sistema para resistir ataques contra su seguridad.

Facilidad de Uso: Es un intento de cuantificar “lo amigable que puede ser con el usuario”.

Capacidad de Soportar Cambios

Facilidad de Mantenimiento: Es el esfuerzo requerido para localizar y arreglar un error en un programa.

Flexibilidad: Es el esfuerzo requerido para modificar un programa que ya está en funcionamiento.

Facilidad de Prueba: Es el esfuerzo requerido para probar un programa de forma que se asegure que realiza su función requerida.

Adaptabilidad de nuevos entornos

Portabilidad: Es el esfuerzo requerido para transferir el programa desde un hardware y/o un entorno de sistema de software a otro.

Reusabilidad: Es el grado en que un programa (o partes de este) se puede reusar en otras aplicaciones, en relación al empaquetamiento y alcance de las funciones que realiza el programa.

Interoperabilidad: Es el esfuerzo requerido para acoplar un sistema con otro.

*Estándares de Calidad del Software a Nivel de Procesos:*

ISO 90003:2004

La aplicación de ISO 90003:2004 es apropiada para un software que: 1) Forma parte de un contrato comercial con otra organización, 2) Es un producto disponible para un sector del mercado, 3) Es usado para soportar los procesos de una organización y 4) Está relacionado a servicios de software.

ISO/IEC 12207:1995

ISO/IEC 12207 puede ser usado para: 1) Adquirir, suministrar, desarrollar, operar y mantener software, 2) Soportar las funciones de aseguramiento de calidad, administración de la configuración, revisiones conjuntas, auditorias, verificación, validación, resolución de problemas y documentación, 3) Administrar y mejorar tanto al personal como a los procesos de la organización, 4) Establecer la administración del software y los ambientes de Ingeniería basados en los procesos de ciclo de vida que se adapten para servir a las necesidades del negocio, 5) Ayudar a un mejor entendimiento entre clientes y proveedores y entre las partes involucradas en el ciclo de vida de un producto de software, 6) Facilitar la comercialización global del software.

Esta norma tiene un proceso de aseguramiento de la calidad: Permite asegurar que el software cumple con los requisitos especificados de calidad. Las actividades de este proceso son: 1) Implementación del proceso, 2) Aseguramiento del producto, 3) Aseguramiento del proceso y 4) Aseguramiento de los sistemas de calidad. Además cuenta con los siguientes procesos:

Proceso de Verificación: Permite determinar si los requisitos están completos y correctos. Define las actividades a realizar por el adquiriente, proveedor o tercera parte independiente para verificar la conformidad de los productos y proyectos con sus especificaciones. Las actividades de este proceso son: 1) Implementación del proceso y 2) Verificación.

Proceso de Validación: Permite determinar si el software cumple con los requisitos previstos para su uso. Define las actividades a realizar por el adquiriente, el proveedor o una tercera parte independiente, para validar si el uso de los productos o servicios del proyecto satisface a los adquirientes. Las actividades de este proceso son: 1) Implementación del proceso y 2) Validación.

Proceso de Revisión: Permite evaluar el estado del software en cada etapa del ciclo de vida. Las actividades de este proceso son: 1) Implementación del proceso, 2) Revisión de la administración del proyecto y 3) Revisiones técnicas.

SPICE (ISO/IEC 15504).

Es el estándar internacional para la evaluación del proceso de software.

Niveles de capacidad en el modelo.

- Nivel 0, No Realizado: Es un fracaso general el tratar de utilizar las prácticas bases a los procesos. Ya que no es fácil identificar las salidas de los procesos o el trabajo de los productos.
- Nivel 1, Realizado Informalmente: Las prácticas de los procesos son ejecutados generalmente. La ejecución de las prácticas dependerá del conocimiento y esfuerzo personal. Se identifica algunos procesos.
- Nivel 2, Planificado y Seguido: La ejecución de las prácticas bases en los procesos son planificadas y seguidas. La primera distinción entre el nivel 1 y el 2 es que la ejecución de los procesos está planificada y administrada y progresan hacia un proceso bien definido.
- Nivel 3, Bien Definido: Las prácticas bases son ejecutadas de acuerdo a una versión adaptada del estándar, procesos aprobados bien definidos y documentados.
- Nivel 4, Cuantitativamente Controlado: Mediciones detalladas de rendimiento o ejecución son alcanzadas y evaluadas. Es conocido el rendimiento de los procesos y es posible su predicción. La calidad de las prácticas es cuantitativamente conocida.
- Nivel 5, Mejoramiento Continuo: Son establecidos en forma cuantitativa procesos y metas eficientes, basados en los objetivos de la organización. En forma continua los procesos se van mejorando mediante la retroalimentación (feedback) obtenida por los resultados de procesos definidos y por ideas pilotos y tecnologías novedosas.

Beneficios y desventajas de SPICE.

Algunos de los beneficios que brinda SPICE:

- Reducción dramática de:
  - Interferencia a los proveedores.
  - Costo a los clientes.

- Combinar la mejor experiencia disponible en mejora del proceso.
- Avanzar el estado del arte utilizando los mejores atributos de todos los métodos existentes.
- Armonizar los esquemas existentes de valoración.

Algunas de las desventajas que tiene es que:

- No es práctico ni fácil de aplicar.
- Tiene solamente lineamientos para un mecanismo de evaluación.

*Estándares de Calidad del Software a Nivel de Producto:*

ISO/IEC 9126-1:2001 – Quality Model.

Esta parte de la ISO 9126 describe el modelo de calidad del producto de software. La primera parte del modelo especifica 6 características de calidad interna y externa, las cuales están divididas en subcaracterísticas, son manifestadas externamente cuando el software es utilizado como parte de un sistema, y son un resultado de atributos internos del software.

La calidad externa evalúa que el software satisfaga las necesidades del usuario teniendo en cuenta las condiciones especificadas. Esta calidad es medible en el comportamiento del producto. La calidad interna evalúa el total de atributos que un software debe satisfacer teniendo en cuenta condiciones especificadas. Esta calidad es medible a partir de las características intrínsecas.

Esta Norma permite especificar y evaluar la calidad del software desde distintas perspectivas, las cuales están asociadas a la adquisición, requerimientos, desarrollo, uso, evaluación, soporte, mantenimiento, aseguramiento de la calidad, y auditoria del software. Puede ser usada por desarrolladores, evaluadores independientes y grupos de aseguramiento de la calidad responsable de especificar y evaluar la calidad del software.

ISO/IEC 25000:2005 – Square.

SQuaRE (Software Quality Requirements and Evaluation) es una nueva serie de normas que se basa en ISO 9126 y en ISO 14598 (Evaluación del software). Uno de los principales objetivos de la serie SQuaRE es la coordinación y armonización del contenido de ISO 9126 y de ISO 15939:2002

(Measurement Information Model). ISO 15939 tiene un modelo de información que ayuda a determinar que se debe especificar durante la planificación, performance y evaluación de la medición.

SQuaRE incluye un estándar de requerimientos de calidad. Está compuesto por 14 documentos agrupados en 5 tópicos: 1) Administración de la Calidad, 2) Modelo de Calidad, 3) Medidas de Calidad 4) Requerimientos de Calidad y 5) Evaluación de la Calidad.

Los beneficios de utilizar SQuaRE son:

1. El modelo representa la calidad esperada del producto de software.
2. Planteo del desdoblamiento de las necesidades o expectativas en calidad en uso, calidad externa y calidad interna.
3. Permite una mayor eficacia en la definición del software.
4. Plantea la evaluación de productos intermedios.
5. Propone una calidad final a través de las evaluaciones intermedias.
6. Permite efectuar un rastreo entre las expectativas, requisitos y medidas de evaluación.
7. Mejora la calidad del producto.

## *1.4. Gestión de la Calidad del Software.*

La Gestión de Calidad, también conocida como Administración de la Calidad, no es más que el conjunto de actividades encaminadas a obtener un producto final con altos índices de calidad y competitividad en el mercado. En el caso específico de la Gestión de la Calidad de Software, algunos autores la definen como el conjunto de actividades de la dirección que determinan la calidad, los objetivos y las responsabilidades, durante el desarrollo de la producción de software. Se basa en la determinación y aplicación de las políticas de calidad de la empresa (objetivos y directrices generales). (1) Se aplica normalmente a nivel de empresa aunque también puede haber una gestión de calidad dentro de la gestión de cada proyecto.

Otros autores utilizan un término muy similar al de Gestión de Calidad; Sistema de Calidad. Plantean que este último es el que define como implementar la Garantía de Calidad y que es un marco en

el que se establecen las diferentes estrategias, actividades y herramientas de garantía de calidad que se van a utilizar. (2)

Para abordar el campo de acción se plantea que la Gestión de la Calidad de Software no es más que el conjunto de actividades planificadas, aprobadas por la dirección y por cada uno de los miembros del equipo, en función de desarrollar un proceso de desarrollo con calidad a fin de obtener un producto de software con un número de defectos reducidos.

La bibliografía consultada refiere también diferentes niveles a los que se puede desarrollar la Gestión de la Calidad de Software. A nivel de entidad u organización o a nivel de proyecto. (1) Otras fuentes definen un nivel más de la siguiente manera: el nivel de organización que es el nivel al que normalmente se establece un sistema de calidad, el nivel de proyecto y un tercer nivel que es el nivel de fase de desarrollo. (2)

*Gestión de la Calidad del Software. El nivel de entidad u organización.*

En este nivel se trata de crear una infraestructura que fomente la calidad de los productos de software mediante la adecuación y mejora de las actividades y procesos involucrados en su producción, e incluso, en su comercialización y en la interacción con los clientes.

Las fuentes consultadas plantean que las organizaciones de software, dentro de este nivel, han seguido una línea marcada por las entidades internacionales de estandarización para todas las organizaciones de producción o servicios. Principalmente se han impuesto la práctica de las directrices marcadas por ISO (Organization for International Standardization) a través de unas normas ISO 9000 para la gestión de calidad. En el caso del software es principalmente aplicable la norma ISO 9001. El sector de software difiere por la naturaleza del producto tanto del resto de los sectores productivos que ha sido necesario crear una guía específica para su aplicación a este sector: ISO 9000-3.

El mundo del software ha creado sus propias líneas para la gestión de la calidad trabajando sobre los procesos, y es entonces cuanto el SEI (Software Engineering Institute) propone un modelo de clasificación y mejora de los procesos empleados por las organizaciones de software denominado CMM.

*Gestión de la Calidad del Software. El nivel de proyecto.*

Es en este nivel donde las guías que la infraestructura productiva prevé para las distintas actividades y mantenimiento del software deben ser adaptadas a las características concretas del proyecto y de su entorno para ser aplicadas a la práctica.

La calidad de software se diseña al mismo tiempo que se planifica el sistema, nunca al final. En los sistemas de calidad se observa que a mayor calidad mayor son los costos, pero mayores son también los beneficios obtenidos en la fase de mantenimiento del software.

El aseguramiento de la Calidad de Software, según (3) engloba un enfoque de gestión de calidad, tecnología de ingeniería de software efectiva (métricas y herramientas), revisiones técnicas formales durante el proceso de desarrollo del software, una estrategia de prueba multiescalada, el control de la documentación del software y de los cambios realizados, un procedimiento que asegure un ajuste a los estándares de desarrollo del software (cuando sea posible) y mecanismos de medición y de generación de informes.

Garantizar la calidad de software a través de la gestión, comprende una gran variedad de tareas, asociadas con dos constitutivos diferentes: los ingenieros de software, que realizan el trabajo técnico y un grupo de aseguramiento de la calidad de software, que tiene la responsabilidad de la planificación de garantía de la calidad. (4) Y no sólo de la planificación sino de la ejecución de las actividades planificadas y de su sistematicidad. Otros autores plantean que la Gestión de la Calidad del Software está formada por cuatro partes, las cuales son: Planificación de la Calidad de Software, Control de la Calidad de Software, Aseguramiento de la Calidad de Software y Mejora de la Calidad de Software.

*Grupo de Garantía de Calidad del Software.*

Los ingenieros de software afrontan la calidad aplicando métodos técnicos sólidos y medidas, realizando revisiones técnicas formales y llevando a cabo pruebas de software bien panificadas. Por su parte, el grupo de aseguramiento de la calidad de software, responsable de la planificación de garantía de la calidad, se basa en un grupo de reglas que tratan de ayudar al equipo de ingeniería del software en la consecución de un producto final de alta calidad. El Instituto de Ingeniería del Software de Estados Unidos, SEI, recomienda un conjunto de actividades de garantía de calidad de software que se enfrentan

con la planificación de garantía de calidad, supervisión, mantenimiento de registros, análisis e informes. Estás son las actividades que realizan o facilitan un grupo independiente de garantía de calidad de software: (5)

Establecimiento de un plan de garantía de calidad de software. El plan se desarrolla durante una planificación del proyecto y es revisado por todas las partes interesadas. Las actividades de garantía de calidad realizadas por el equipo de ingeniería del software y el grupo de garantía de calidad del software son gobernadas por el plan. El mismo identifica:

- Evaluaciones a realizar,
- Auditorías y revisiones a realizar,
- Estándares que se pueden aplicar al proyecto,
- Procedimientos para información y seguimiento de errores,
- Documentos producidos por el grupo de garantía de calidad del software,
- Retroalimentación de información proporcionada al equipo de proyecto del software.

Participación en el desarrollo de la descripción del proceso de software de proyecto. El equipo de ingeniería del software selecciona un proceso para el trabajo que se va a realizar. El grupo de garantía de la calidad del software revisa la descripción del proceso para ajustarse a la política de la empresa, los estándares internos del software, los estándares impuestos externamente (por ejemplo: ISO 9001), y a otras partes del plan de proyecto de software.

Revisión de las actividades de ingeniería del software para verificar su ajuste al proceso de software definido. El grupo de garantía de calidad del software identifica, documenta y sigue la pista de las desviaciones desde el proceso y verifica que se han hecho las correcciones.

Auditoría de los productos de software designados para verificar el ajuste con los definidos como parte del proceso del software. El grupo de garantía de calidad del software revisa los productos seleccionados; identifica, documenta y sigue la pista de las desviaciones; verifica que se han hecho las correcciones, e informa periódicamente de los resultados de su trabajo al gestor del proyecto.

Asegurar que las desviaciones del trabajo y los productos del software se documentan y se manejan de acuerdo con un procedimiento establecido. Las desviaciones se pueden encontrar en el plan del proyecto, en la descripción del proceso, en los estándares aplicables o en los productos técnicos.

Registrar lo que no se ajuste a los requisitos e informar a sus superiores. Los elementos que no se ajustan a los requisitos están bajo seguimiento hasta que se resuelven.

Además de estas actividades, el grupo de garantía de la calidad de software coordina el control y la gestión de cambios y ayuda a recopilar y a analizar las métricas del software.

*Planificación de la Calidad del Software.*

La Planificación de la Calidad de Software, según la bibliografía consultada, es la parte de la Gestión de la Calidad encargada de realizar el proceso administrativo de desarrollar y mantener una relación entre los objetivos y recursos de la organización; y las oportunidades cambiantes del mercado. El objetivo es modelar los negocios y productos de la empresa, de manera que se combinen para producir un desarrollo y utilidades satisfactorias.

Los aspectos a considerar en la Planificación de la Calidad de Software son: Modelos/Estándares de Calidad de Software a utilizar, Costos de la Calidad de Software, Recursos humanos y materiales necesarios, etc. Los factores que determinan el Modelo o Estándar de Calidad de Software a elegir son: La complejidad del proceso de diseño, La madurez del diseño, La complejidad del proceso de producción, Las características del producto o servicio, La seguridad del producto o servicio, y Económico.

Según la norma ISO/IEC 90003:2004 se puede decir que:

“La planificación de la calidad facilita el modo de adaptar la planificación del sistema de gestión de la calidad a un proyecto específico, producto o contrato. La planificación de la calidad puede incluir referencias genéricas y/o proyecto / producto / contrato específico de procedimientos, como apropiados. La planificación de la calidad debería ser revisada de nuevo junto con el progreso del diseño y desarrollo, y los elementos, en cada fase, deberían ser completamente definidos al comienzo de dicha fase” (5)

Según (1), la Planificación de la Calidad de Software a nivel de proyectos debería considerar lo siguiente:

1. Inclusión de los planes de desarrollo.
2. Los requisitos de calidad relacionados con los productos y/o procesos.
3. Los sistemas de gestión de calidad adaptando y/o identificando los procesos e instrucciones específicos, apropiados para el ámbito del manual de calidad y algunas exclusiones expuestas.
4. Los procesos de proyectos específicos e instrucciones, tales como, especificación de pruebas del software detallando los planes, diseños, casos de pruebas y procesos para la unidad, integración, sistemas y pruebas de aceptación.
5. Los métodos, modelos, herramientas, convenios de lenguajes de programación, bibliotecas, marcos de trabajo, y otros componentes reutilizables para ser usados en los proyectos.
6. Los criterios para el comienzo y el final de cada fase o etapa del proyecto.
7. Los tipos de análisis y otras verificaciones y actividades de validación para ser llevadas a cabo.
8. Los procesos de gestión de la configuración para ser llevados a cabo.
9. Las actividades de seguimiento y las medidas para ser llevadas a cabo.
10. Las personas responsables de aprobar los procesos de salida para su uso posterior.
11. La formación necesaria para el uso de herramientas y técnicas, y la organización de la formación previa a la habilidad necesaria.
12. Los registros para ser mantenidos.
13. La gestión de cambios, como por ejemplo, para recursos, escalas de tiempo y cambios de contrato.

La planificación de la calidad, por otra parte, abreviada, es particularmente útil para limitar los objetivos de calidad para el software, siendo designados para un propósito limitado.

Según Humphrey (1989) un plan de calidad puede tener la siguiente estructura:

1- Introducción al producto: una descripción del producto, su objetivo en el mercado y expectativas de calidad del producto.
2- Planes del producto: fechas críticas de acuerdo a la liberación del producto y responsabilidades del producto respecto de su distribución y servicio.
3- Descripciones del proceso: Procesos de desarrollo y servicios que serían utilizados en el desarrollo y en la administración.
4- Objetivos de Calidad: objetivos y planes de calidad del producto, los cuales incluyen la identificación de los atributos de calidad del producto.

5- Manejo del riesgo: principales riesgos que pueden afectar la calidad del producto.

Esta información es fundamentalmente recogida en un plan de calidad. El Plan de Calidad define los atributos de calidad más importantes del producto a ser desarrollado y define el proceso de evaluación de la calidad.

El IEEE [IEEE94] ha recomendado un estándar para los planes de garantía de calidad del software. Las secciones iniciales describen el propósito y el alcance del documento e indican aquellas actividades del proceso del software cubiertas por la garantía de calidad. Se listan todos los documentos señalados en el plan de garantía de calidad de software y se destacan todos los estándares aplicables. La sección de Gestión del plan describe la situación de la garantía de calidad del software dentro de la estructura organizativa; las tareas y las actividades de garantía de calidad del software y su emplazamiento a lo largo del proceso del software; así como los papeles y responsabilidades organizativas relativas a la calidad del producto. (5)

La sección de Documentación describe (por referencia) cada uno de los productos de trabajo producidos como parte del proceso de software. Entre estos se incluyen:

- Documentos del proyecto (por ejemplo: plan del proyecto),
- Modelos (por ejemplo: Diagramas de Entidad-Relación, jerarquías de clases),
- Documentos técnicos (por ejemplo: especificaciones, planes de prueba),
- Documentos de usuario (por ejemplo: archivos de ayuda).

Además, aquí se definen el conjunto mínimo de productos de trabajo que se pueden aceptar para lograr alta calidad.

Los estándares, prácticas y convenciones muestran todos los estándares/prácticas que se aplican durante el proceso de software (por ejemplo: estándares de documentos, estándares de codificación y directrices de revisión). Además, se listan todos los proyectos, procesos (en algunos casos) métricas de producto que se van a recoger como parte del trabajo de ingeniería del software.

La sección Revisiones y Auditorías del plan, identifica las revisiones y auditorías que se van a llevar a cabo por el equipo de ingeniería de software, el grupo de garantía de calidad del software y el cliente. Proporciona una visión general del enfoque de cada revisión y auditoría.

La sección Prueba hace referencia al Plan y Procedimiento de Pruebas del Software. También define los requisitos de mantenimiento de registros de pruebas. La información sobre los problemas y acción correctiva define procedimientos para informar, hacer seguimiento y resolver errores y defectos, e identifica las responsabilidades organizativas para estas actividades.

El resto del plan de garantía de calidad del software identifica las herramientas y métodos que soportan actividades y tareas de garantía; hace referencia a los procedimientos de gestión de configuración del software para controlar el cambio; define un enfoque de gestión de contratos; establece métodos para reunir, salvaguardar y mantener todos los registros; identifica la formación que se requiere para cumplir las necesidades del plan y define métodos para identificar, evaluar, supervisar y controlar riesgos. (5)

En la Planificación de la Calidad de Software se debe determinar, según (1), el Rol de la Planificación, los Requerimientos de la Calidad del Software, la Preparación de un Plan de Calidad del Software, la Implementación de un Plan de Calidad del Software y preparar un Manual de Calidad.

*Control de la Calidad del Software.*

Según la norma ISO 9000:2000, el control de calidad es la parte de la gestión de la calidad orientada al cumplimiento de los requisitos de la calidad. El control de la calidad del software, son las técnicas y actividades de carácter operativo, utilizadas para satisfacer los requisitos relativos a la calidad, centradas en dos objetivos fundamentales: mantener bajo control un proceso y eliminar las causas de los defectos en las diferentes fases del ciclo de vida. Está formado por actividades que permiten evaluar la calidad de los productos de software desarrollados. El aspecto a considerar en el Control de la Calidad del Software es la “Prueba de Software”. (1)

El Control de Calidad, según Pressman (5), es una serie de inspecciones, revisiones y pruebas utilizadas a lo largo del proceso del software para asegurar que cada producto cumple con los requisitos que le han sido asignados. El Control de Calidad incluye un bucle de realimentación (feedback) del

proceso que creó el producto. La combinación de medición y realimentación permite afinar el proceso cuando los productos de trabajo creados fallan al cumplir sus expectativas. Este enfoque ve el Control de Calidad como parte del proceso de fabricación.

Según algunos autores, se puede clasificar las actividades de control de calidad de dos categorías: controles estáticos y controles dinámicos. Los primeros analizan el objeto sin necesidad de ejecutarlo mientras que los segundos requieren la ejecución del objeto que está siendo probado. (2) La barrera entre controles estáticos y dinámicos no es totalmente estricta. Cualquier forma de control dinámico requiere un cierto grado de análisis estático. Además, hay algunas técnicas, como la verificación formal y la ejecución simbólica, consideradas como estáticas, que “ejecutan” el código, aunque en un entorno no real.

Otros autores consideran que las actividades de Control de Calidad pueden ser manuales, completamente automáticas o una combinación de herramientas automáticas e interacción humana. Definen que un concepto clave del Control de Calidad, lo constituye el hecho de que se hayan definido todos los productos y las especificaciones mensurables en las que se puedan comparar los resultados de cada proceso. El bucle de realimentación es esencial para reducir los defectos producidos. (5)

De los tipos de controles que relaciona la bibliografía, vamos a referirnos a los controles estáticos y dinámicos y centrarnos en los dinámicos, que son básicamente, las “Pruebas de Software”.

Los controles estáticos se clasifican en Manuales y Automáticos. Los Manuales, a su vez se clasifican en Informales y Disciplinados.

***Controles estáticos manuales informales.***

Estas actividades las realizan los propios autores de los objetos a comprobar, o personas de su misma categoría y ubicación.

*Comprobación de escritorio (desk checking):*

Consiste en examinar a mano e individualmente el objeto que se acaba de desarrollar. Es el método más tradicional para analizar un programa. Se debe aplicar a los requisitos, especificaciones de

diseño y código según se van desarrollando. Debe ser cuidadoso y concienzudo para que sea efectivo. Es más efectivo si se hace intercambiando el objeto a examinar con otro compañero.

*Revisión por pares o iguales (peer review):*

Consiste en la revisión del código de un programador por otros programadores (sus pares). Se puede poner en práctica creando un panel que se encarga de revisar periódicamente muestras de código.

***Controles estáticos manuales disciplinados.***

Las revisiones y auditorías son la evolución natural de la Comprobación de Escritorio, pero a diferencia de aquella pasan a ser técnicas de grupo. Su misión principal es conseguir que la responsabilidad del Control de Calidad no recaiga sólo sobre el propio desarrollador.

*Auditorías*

Una auditoría consiste en realizar una investigación para determinar:

- El grado de cumplimiento y la adecuación de los procedimientos, instrucciones, especificaciones, códigos, estándares u otros requisitos de tipo contractual, establecidos y aplicables.
- La efectividad y adecuación de la implementación realizada.
- Se pueden considerar tres tipos de auditorías:
- Auditoría del producto: el objetivo es cuantificar el grado de conformidad del producto con las características requeridas. Las auditorías del producto software más comunes son la auditoría Funcional y la auditoría Física.
- Auditoría del proceso: el objetivo es evaluar el proceso de desarrollo o de gestión, y evaluar su completitud y efectividad, determinando dónde se puede mejorar. En el desarrollo de software se suelen realizar dos tipos de auditorías del proceso:

- Prueba de aceptación. (La prueba de aceptación se realiza una vez que el sistema se ha implantado en su entorno real de funcionamiento, y su objetivo es demostrar al usuario que el sistema satisface sus necesidades)
- Auditorías de gestión de proyecto: cuyo objetivo es evaluar la efectividad de las prácticas de gestión realizadas y la organización del proyecto.

- Auditoría del sistema de calidad: el objetivo es evaluar la completitud y efectividad del propio sistema de calidad establecido.

*Revisiones*

Se puede definir una revisión como una reunión formal en la que se presenta el estado actual de los resultados de un proyecto a un usuario, cliente u otro tipo de persona interesada, y se realiza un análisis estructurado de los mismos.

Uno de los objetivos fundamentales de las revisiones técnicas es ofrecer a los gestores información fiable acerca de los aspectos técnicos del proceso de desarrollo de software, para que con esta información puedan tomar decisiones adecuadas para dirigir con éxito el proyecto.

Con las revisiones se consigue que el peso de la evaluación técnica no recaiga sobre las mismas personas involucradas en la producción del software, que por la posición que ocupan no pueden ser totalmente objetivas, sino en otras personas técnicamente competentes y objetivas.

Según la bibliografía, las revisiones son, hoy en día, el único método de control de calidad eficaz en las fases iniciales del desarrollo a la hora de identificar desviaciones con respecto a las especificaciones de calidad. Las revisiones redundan en una mejora directa de la calidad del objeto que se examina y provocan, indirectamente, una mejora de la calidad del proceso de desarrollo, al facilitar la comunicación entre los miembros del equipo de desarrollo. Al mismo tiempo facilitan el control del coste y el tiempo.

En este momento se clasificarán las revisiones de acuerdo al material bibliográfico consultado. Posteriormente, cuando se describan las actividades de Aseguramiento de la Calidad del Software, se

hará énfasis en las Revisiones Técnicas Formales, teniendo en cuenta que se han definido como la actividad fundamental de aseguramiento de la calidad.

Hay dos tipos fundamentales de revisiones, las inspecciones y los walkthrough. En las inspecciones, los participantes van leyendo el documento, paso a paso, guiados por el autor del mismo, y comprobando en cada paso el cumplimiento de los criterios de una lista de comprobación. Por su parte, en los walkthrough o visitas guiadas, se demuestra la funcionalidad del objeto revisado mediante la simulación de su funcionamiento con casos de prueba y ejemplos. Se introducen al objeto los casos de prueba y se van registrando los resultados intermedios. Aunque estos son los tipos ideales, en la vida real hay otras muchas variantes intermedias, desde las revisiones sin disciplina alguna hasta cualquier tipo de mezcla entre inspecciones y walkthrough.

Las revisiones se pueden clasificar en formales e informales. Son formales cuando constituye un evento público, se informa por escrito los resultados e involucra a todos los participantes como responsables de la calidad de la revisión. La ventaja de las revisiones formales radica en que los informes que se generan sirven como hitos para el proyecto, y el hecho de ser algo público promueve una mejor preparación por parte de los participantes.

Por otro lado, según el objetivo que se revise, las revisiones se pueden diferenciar entre revisiones con orientación técnica y las revisiones orientadas a la gestión. Las revisiones técnicas más comunes son: Revisión de especificación de requisitos, Revisión del diseño, Revisión del código, Revisión de las pruebas, Revisión del manual de usuario.

En otro orden, con las revisiones de gestión o de proyecto se persigue controlar la progresión del proyecto; evaluar los riesgos asociados a los mismos, con relación al coste, escala de tiempo, recursos utilizados y calidad del producto; y evaluación general del producto. Para esto, es necesario que exista un plan de desarrollo bien estructurado, con hitos bien definidos, que permitan evaluar la progresión del proyecto y que los resultados del proyecto se encuentren bien documentados y hayan sido ya examinados en una revisión técnica.

***Controles estáticos automáticos.***

Dentro de esta categoría tenemos el análisis estático automático y la verificación formal de programas. La mayor parte del análisis estático automático del código lo realizan los compiladores, que pueden detectar desde expresiones estáticamente incorrectas hasta incompatibilidades de tipo y otros errores de tipo semántico. Otras técnicas de análisis estático automático de programas son: análisis de flujo y ejecución simbólica.

El análisis de flujo se basa en una ejecución básica. Se usan grafos en los que los nodos representan sentencias o segmentos de programas y los arcos, posibles transiciones de control desde un segmento a otro. Estos grafos se pueden utilizar para identificar caminos, para analizar el comportamiento del programa, para situar puntos de ruptura y para otras técnicas de análisis estático. La técnica consiste en trazar el comportamiento de las variables del programa desde su inicialización hasta que termina la ejecución del programa.

La ejecución simbólica consiste en la ejecución figurada de ciertos caminos dentro del programa, durante la cual se verifican ciertas expresiones simbólicas con respecto a ciertas condiciones y afirmaciones preestablecidas. El resultado de la ejecución simbólica se puede entonces comparar con la especificación externa del programa y ver si están de acuerdo una con otra. Para su análisis, la mejor forma es representando el resultado descompuesto en una estructura de árbol donde cada hoja representa un camino de ejecución y cada ramificación representa un punto de decisión en el programa.

La verificación formal, por su parte, consiste en demostrar matemáticamente la corrección de un programa con respecto a sus especificaciones. Para ellos se considera un programa como un objeto formal, es decir, como una cadena de un lenguaje formal, con una sintaxis y una semántica formal. Es necesario que la especificación se haya escrito en un lenguaje formal. Por eso no siempre es posible realizar este tipo de verificación.

***Controles dinámicos.***

Se llama controles dinámicos a aquellos que requieren la ejecución del objeto que se está probando o de un modelo del mismo. Según se plantea en la bibliografía consultada, hasta la fecha no se ha desarrollado ninguna teoría universalmente aceptada acerca de la prueba de software. Lo único que

hay es un conjunto de aproximaciones metodológicas que facilitan y hacen más eficiente el proceso de prueba.

Se llama prueba de software, según Pressman (5) al proceso en el que se ejecuta un sistema con el objetivo de detectar fallos. Es un proceso destructivo que determina el diseño de los casos de prueba y la asignación de responsabilidades.

*Tipos de pruebas*

El proceso de prueba conlleva la realización de un conjunto de tareas a lo largo del ciclo de vida del sistema. De acuerdo con el estándar IEEE 1012-1986 el conjunto mínimo de pruebas que se deben realizar son:

- Prueba modular, prueba unitaria o prueba de componentes. (Consiste en la prueba de cada módulo aislado del sistema)
- Prueba de integración. (Se realiza a medida que los diferentes módulos del sistema se integran en el mismo. El objetivo fundamental de esta prueba es comprobar que las interfaces entre los distintos módulos son correctas.)
- Prueba del sistema. (Se realiza cuando se han integrado todos los módulos, y su objetivo es comprobar que el sistema satisface los requisitos del usuario, tanto los funcionales como los no funcionales)
- Prueba de aceptación. (La prueba de aceptación se realiza una vez que el sistema se ha implantado en su entorno real de funcionamiento, y su objetivo es demostrar al usuario que el sistema satisface sus necesidades)
- También se suele realizar otro tipo de prueba llamada prueba de regresión. (Esta tiene como objetivo comprobar que toda nueva versión de un producto software es de no menos calidad que la versión anterior, es decir, que al introducir cambios no se ha reducido la valoración de ninguna de las características de calidad que tenía el producto)

*Métodos de prueba*

Los métodos de prueba se clasifican en Métodos de Caja Negra y Métodos de Caja Blanca.

En el método de caja negra la elección de los casos de prueba no se va a basar en el conocimiento que se tenga acerca de la estructura del objeto, sino en el conocimiento acerca de la funcionalidad deseada. A la prueba de caja negra también se le llama prueba funcional o prueba orientada al diseño. Una prueba de caja negra exhaustiva requeriría la generación de un caso de prueba por cada combinación posible (válida o no válida) de los valores de entrada, lo cual resulta imposible en la mayor parte de los casos por producirse una explosión combinatoria. Por eso se utilizan diferentes criterios a la hora de restringir el conjunto de casos de prueba.

Los métodos de selección del conjunto de casos de prueba más usuales son:

- Método de clases de equivalencia:
  Consiste en dividir las posibles entradas al sistema en clases de equivalencia, de tal forma que todos los miembros de una misma clase de equivalencia prueben las mismas propiedades en el sistema, por lo que sólo va a ser necesario seleccionar un elemento de cada clase de equivalencia.
- Análisis de valores frontera o valores límite Calidad del Software
  Consiste en seleccionar como casos de prueba aquellos valores de entrada que caen en la frontera de las clases de equivalencia (justo a un lado, justo al otro y justo en la frontera).
- Grafos causa/efecto y tablas de decisión
  Consiste en crear un grafo causa/efecto a partir de las especificaciones, y seleccionar suficientes casos de prueba como para asegurar la cobertura del grafo. Se llama causas a las características de los datos de entrada y efectos a las clases de salidas que puede proporcionar el programa. A partir del grafo causa/efecto se construye una tabla de decisión que refleje las dependencias entre causas y efectos. Lo que se hace entonces es reducir la tabla de decisión y seleccionar sólo un caso de prueba para todas las causas que producen el mismo efecto, o para cada columna de la tabla de decisión.
- Adivinación de errores
  Consiste en tratar de imaginar cuáles son los errores que se pueden haber cometido con mayor probabilidad, y generar casos de prueba para comprobar dichos errores.

En el método de caja blanca la elección de los casos de prueba se va a basar en el conocimiento que se tenga acerca de la estructura del objeto (diseño detallado, diagramas de flujo de datos y de control, código fuente). A la prueba de caja blanca también se le llama prueba estructural.

Los métodos de caja blanca se pueden clasificar, a su vez, en dos grupos, los basados en métricas de cobertura y los basados en métricas de complejidad.

Métodos basados en métricas de cobertura:

Todo programa se puede representar mediante un grafo de flujo de control, donde cada nodo es una sentencia o una secuencia de sentencias. Los arcos dirigidos en el grafo representan el flujo de control.

Para cada conjunto de datos de entrada el programa se ejecutará a través de un camino concreto dentro de este grafo. Cuando el programa incluye estructuras iterativas, el número de posibles caminos en el grafo puede ser infinito.

Una prueba de caja blanca exhaustiva requeriría la generación de un caso de prueba por cada posible camino. Como esto no es posible, por lo general, se utilizan métricas que dan una indicación de la calidad de un determinado conjunto de casos de prueba en función del grado de cobertura del grafo que consiguen. Las métricas más utilizadas son:

- Cobertura de sentencias.
- Cobertura de segmentos entre decisiones.
- Cobertura de decisiones de ramificación.
- Cobertura de condiciones.
- Cobertura de todas las combinaciones de condiciones.
- Cobertura de caminos.

Métodos basados en métricas de complejidad:

Las métricas de complejidad más utilizadas en la generación de casos de prueba son las de MacCabe:

- Complejidad ciclomática (arcos - nodos + 2 * número de componentes conexos)
- Complejidad esencial (complejidad ciclomática - número de subgrafos propios de entrada y salida única)
- Complejidad real (número de caminos ejecutados)

*Metodología de prueba*

Cada uno de los diferentes tipos de prueba implica la realización de un conjunto de actividades estándar, así como la producción de un conjunto de salidas estándar.

| **Actividades estándar de prueba** | **Salidas estándar asociadas** |
|---|---|
| Planificación de la prueba | Plan de pruebas |
| Diseño de la prueba | Documento de diseño de la prueba |
| Determinación de los casos de prueba | Especificación de los casos de prueba |
| Planificación del procedimiento de prueba | Especificación del procedimiento de prueba |
| Ejecución de la prueba | Informe de los casos de prueba |
| Análisis y evaluación de la prueba | Informe de la prueba |

- Planificación de la prueba: Esta actividad consiste en la creación de un plan de pruebas en el que se registra:

- El objetivo del proceso de prueba.
- Los objetos que hay que probar.
- Las características que se van a probar y las que no.
- El método de prueba a utilizar.
- Los recursos que se van a emplear.
- El plan de tiempos.
- Los productos a generar durante las pruebas
- El reparto de las responsabilidades.

- Diseño de la prueba: Esta actividad consiste en dar instrucciones detalladas acerca de:

- Cómo llevar a cabo la prueba para alcanzar los objetivos deseados.
- De qué forma se van a utilizar los métodos de prueba.

- Qué objetos se van a probar en cada una de las pruebas.
- Qué criterios se van a utilizar para determinar si el objeto pasa o no pasa la prueba.

  - Determinación de los casos de prueba: Esta actividad consiste en especificar el conjunto de casos de prueba a utilizar en función del diseño realizado para la prueba.

Para cada caso de prueba habrá que especificar:

- Qué objetos se van a probar.
- Qué entradas se les van a dar.
- Cuáles son las salidas esperadas.

  - Planificación del procedimiento de prueba: Esta actividad consiste en fijar un conjunto de pasos para la ejecución de la prueba. Se especifica detalladamente:

- La secuencia exacta de ejecución de los distintos casos de prueba.
- Los requisitos que hay que cumplir para la ejecución de cada caso.
- Las condiciones de terminación de cada uno de ellos.

  - Ejecución de la prueba: Esta actividad consiste en ejecutar cada caso de prueba, según el procedimiento especificado en el paso anterior, y registrar los incidentes o problemas encontrados durante la misma.
  - Análisis y evaluación de la prueba: Se examinan los resultados de la prueba y se decide si se han alcanzado los objetivos propuestos o se debe repetir la prueba.

# Capítulo 2: Características de la Solución.

## *2.1 Introducción.*

Para gestionar la calidad de cualquier proceso es preciso identificarlo. No se puede pretender gestionar calidad sin conocer el objeto de esa gestión: el proceso, las actividades o eventos que se realizan o suceden con un determinado fin.

Según se estudió en la gestión de calidad del software a nivel de entidad u organización, en el proyecto, para elaborar el mecanismo de gestión de la calidad interna, es preciso conocer primero el proceso de producción de software; identificar las actividades que se realizan, los roles involucrados en la realización de cada una de esas actividades y definir las entradas y salidas que se mueven en las mismas.

Una vez conocido el proceso de producción entonces se define un proceso para gestionar la calidad de ese proceso. Se definen igualmente las actividades de gestión de la calidad conjuntamente con los roles que las desempeñan junto a su perfil de competencias. Se establece además los resultados que deben obtenerse en cada una de las actividades de control de calidad y las entradas de las que se parte.

Una parte importante de la gestión de la calidad en el proyecto de desarrollo de software es definir los objetivos de la calidad, de la misma manera que se planteó en el capítulo anterior, relacionado con la planificación de la calidad del software. Si ya se tiene identificado el proceso de producción, pues se necesita identificar el interés de la dirección del proyecto respecto a la calidad. Es el momento de enfocar los objetivos de la gestión de la calidad a los objetivos de la dirección del proyecto. De esta manera se consigue el compromiso de la dirección con la calidad y se involucra al equipo de desarrollo.

Definidos los objetivos, se procede a identificar las actividades a realizar en cada disciplina del proceso de desarrollo. En cada una de las actividades se definen tareas, responsable, precondiciones y poscondiciones que no son más que las entradas y salidas sobre las que actúa el proceso de gestión de calidad interna. Se determina además, el flujo de trabajo que guiará el proceso de gestión.

Resulta muy difícil gestionar la calidad de un proceso sin contar con las personas encargadas de realizar esta gestión. En dependencia del tamaño y del alcance del proyecto será la cantidad de personas necesarias. No resulta para nada conveniente que toda esta responsabilidad caiga sobre una persona, por

lo que es preciso también determinar la organización del equipo de calidad interna, con la descripción de la responsabilidad de cada miembro y los conocimientos mínimos para desempeñarse en su rol. Es oportuno además, definir un plan de capacitación para lograr un mejor desempeño en la gestión. El plan de capacitación debe incluir contenido afín a las actividades o a la responsabilidad de cada miembro, de modo que se logre uniformidad en el conocimiento que se consolide luego con la práctica.

Otro elemento importante, y que constituye a su vez un punto de partida importante a tener en cuenta para la gestión interna de la calidad en un proceso de desarrollo de software, lo constituyen las pautas o lineamientos definidos por una dirección de calidad central en la empresa u organización. La gestión de calidad interna no debe romper con dichos lineamientos sino que debe trabajar también en función de responder a ellos de la manera más factible.

## *2.2 Identificación del Proceso de Desarrollo de Software.*

*El proceso de desarrollo de Ingeniería de Software.*

El proceso de desarrollo de software es una descripción de la construcción del software que contiene actividades organizadas, orientadas a ese fin. No existe una definición estándar de estas actividades y muchos autores le dan importancia a algunas más que a otras. Generalmente, podemos clasificar estas actividades en:

- Ingeniería de Requerimientos.
- Diseño del Sistema.
- Implementación del Software.
- Prueba o Validación del Software.
- Mantenimiento.

En la Ingeniería de Requerimientos, es necesario conocer la naturaleza del sistema, entender lo que desean los clientes, delimitar el alcance del sistema teniendo en cuenta el tiempo disponible, el presupuesto y el personal asignado. El producto generado de esta etapa, es un documento de

especificación de requerimientos en donde se encuentran definidos todos los servicios requeridos del sistema y las restricciones sobre las que debe operar.

El Diseño del Sistema toma como base el documento de especificación de requerimientos, agrega detalles a cada requerimiento, identifica los subsistemas, describe la estructura interna de los datos y las interfaces entre los componentes del sistema. El Diseño del Sistema utiliza varios modelos para la representación del sistema desde diferentes perspectivas y niveles de abstracción. El resultado de esta etapa es la especificación precisa de los algoritmos y estructuras de datos que van a implementarse.

En la etapa de Implementación se lleva a cabo la codificación del sistema, se deben satisfacer los requerimientos de la manera que especifica el diseño detallado. El programador examina los documentos generados en las etapas anteriores para evitar inconsistencias entre los documentos. Se toman en cuenta los estándares de programación, así como los lenguajes de programación.

Las Pruebas y la Validación se utilizan para demostrar que el sistema cumple con su especificación y satisface las necesidades del usuario. De cualquier forma, es necesario llevar a cabo un proceso de verificación por medio de inspecciones y revisiones desde la especificación de los requerimientos hasta la puesta en marcha del sistema. El propósito de realizar pruebas no es demostrar que una aplicación es satisfactoria, sino determinar con firmeza en qué parte no lo es. Las pruebas nos permiten mostrar la presencia de defectos en el sistema.

El Mantenimiento de Software consiste en las actividades realizadas sobre el sistema una vez entregado y puesto en marcha. Una definición es (Ian Sommerville) (6) “El proceso de modificar un sistema o componente de software entregado para corregir defectos, mejorar el desempeño, mejorar algún atributo, o adaptarlo al cambio de entorno”.

Como se observa, el resultado de la Ingeniería de Software consiste en mucho más que el código del sistema, es decir, incluye planes, informes, documentos e inclusive programas llamados prototipos que son desechados después de lograr el objetivo por lo que fueron creados.

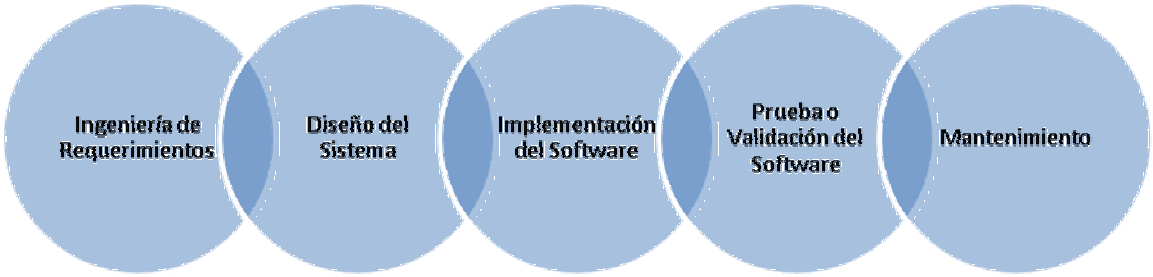



**FIG. 2 CLASIFICACIÓN GENERAL DE LAS ACTIVIDADES DEL PROCESO DE DESARROLLO DE INGENIERÍA DE SOFTWARE**

## *2.3 Identificación de los Objetivos de Gestión de la Calidad Interna.*

Una vez que se conoce el proceso de desarrollo de software, se definen las actividades que se desarrollan en las diferentes disciplinas así como los responsables, las entradas y los resultados. Partiendo de la definición y análisis de la gestión de calidad del software, estudiada en el capítulo anterior, se puede dar otro paso de avance en búsqueda de la gestión de la calidad interna.

Ahora bien, gestionar la calidad interna del proyecto requiere tener claridad en cuanto a cómo se desea gestionar, o sea, como se ejercerá la influencia para obtener los resultados satisfactorios de calidad. Una vez ya determinado el objeto sobre el que se centra la atención de la gestión, o sea, el proceso, entonces resta otro paso importante que es el de identificar y describir las acciones concretas que nos van a permitir actuar sobre el objeto y obtener resultados que nos permitan tener criterio valorativo sobre la calidad como característica y poder tener argumentos para responder por ella.

Para esto, es importante determinar los objetivos que se persiguen con la gestión de calidad interna, reflejar claramente la finalidad de la gestión y el radio de acción sobre el que va a desarrollar la influencia. Este radio de acción puede ser tan amplio como sea necesario, pues la gestión no sólo se enmarca dentro del proceso de desarrollo de software, o sea, dentro del proceso netamente productivo. También existen recursos materiales, recursos humanos que pueden accionar positiva o negativamente sobre la producción de software, como se estudió anteriormente en el epígrafe 1.2, relacionado con el grupo de garantía de la calidad del software.

Un momento importante para la determinación de los objetivos de la gestión de la calidad es la identificación de los intereses de la dirección del proyecto. Los objetivos tienen que estar enfocados a los objetivos del proyecto y deben responder a la dirección.

Una manera práctica de lograr que los objetivos de la gestión de la calidad interna respondan a los intereses de la dirección es determinando en cada disciplina las tareas y actividades que se van a desarrollar que respondan a lograr índices aceptables de calidad y enfocados en los intereses del proyecto.

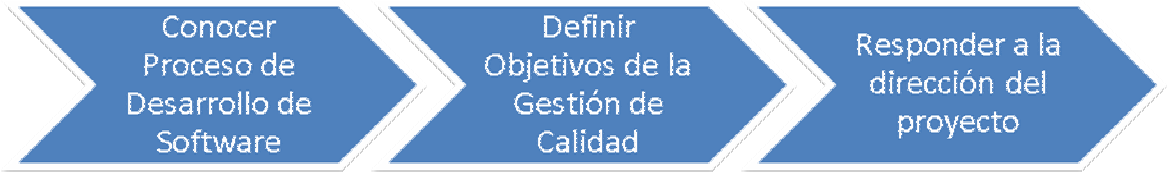



**FIG. 3 IDENTIFICACIÓN DE LOS OBJETIVOS DE LA CALIDAD INTERNA**

*Actividades y tareas enfocadas a lograr los objetivos de la gestión de calidad interna*

Como primer paso previo a la determinación de los objetivos se ha identificado el proceso de desarrollo de software. Se han identificado las actividades, entradas y resultados en cada disciplina, respaldados por el desempeño de un rol en el equipo de proyecto.

Por cada uno de estos elementos, el equipo de calidad interna se propone realizar un conjunto de actividades y tareas que garanticen la gestión de calidad. Es el momento de pensar en el control de la calidad de software, partiendo de lo reflejado al respecto en la fundamentación teórica.

Para orientar los objetivos de la gestión de las actividades que se realizan en cada disciplina, se propone primeramente gestionar el flujo de trabajo dentro de esa actividad. Es preciso primeramente conocer cómo se lleva a cabo dicha actividad para tener una base que permita posteriormente determinar

cómo se va a realizar la gestión, partiendo del principio de que para gestionar debemos conocer el objeto de la gestión.

Identificado el flujo de trabajo de realización de esa actividad en una disciplina determinada, se está apto para ejercer un control lo más estricto posible para lograr que ese flujo de trabajo se realice de acuerdo a como ha sido definido. Esto no quiere decir que dicho flujo no esté sujeto a variaciones, pues depende mucho de las características del entorno, las características del momento, para que el orden de las actividades pueda ser alterado o no.

Por tal motivo es importante que el grupo de calidad se trace también la tarea de gestionar las variaciones que sufre el flujo de trabajo de la actividad de cada disciplina. Tener conocimiento del comportamiento del flujo es la base para analizar posibles mejoras que se traducen posteriormente en mejoras para el proceso de desarrollo de software. En tal sentido se enfoca además la última tarea propuesta que es la del registro de los resultados de todos los puntos descritos anteriormente.

Otro de los momentos importantes detectados en la identificación del proceso de desarrollo de software lo constituyen las entradas en cada disciplina. Gestionar dichas entradas también nos tributa a la gestión de calidad y en este sentido de deben proponer actividades que permitan el conocimiento y la gestión de las mismas.

Conocidas las entradas es preciso entonces controlar la integridad y completitud de las mismas. Si las entradas o el punto de partida para la realización de una actividad dentro de una disciplina del proceso de desarrollo de software son incompletas pues dicho defecto atenta contra el buen desempeño de la actividad en cuestión. Por eso es preciso tener conocimiento y controlar esas entradas para tener garantía para el desarrollo del flujo de trabajo de la disciplina.

De la misma manera que en las actividades, las entradas a dichas actividades no están exentas de sufrir variaciones dadas por variaciones en el entorno o cualquier otro motivo. Por eso es importante gestionar dichas variaciones, conocerlas, tener claridad en cuanto al tipo de variación que se produce con las entradas para tener una valoración de su carácter y de las ventajas o desventajas que provee al proceso.

Documentar los resultados, registrarlos, también constituye una base sólida para la gestión de la calidad. Nos permite tener constancia del desarrollo histórico de la gestión de calidad y que constituye cantera para análisis más profundos.

El momento más importante del proceso de desarrollo del software es la obtención de resultados de las actividades de cada una de las disciplinas. Dichos resultados no son más que las salidas de cada uno de los subprocesos de desarrollo de software, que a su vez, constituyen un elemento que nos permite evaluar acorde a los criterios de calidad.

Por supuesto que de la misma manera, para gestionar estos resultados, hay que conocer cuáles son los que deben obtenerse como salida de cada disciplina. Sólo una vez identificados se puede actuar sobre ellos a través del control. Este control sobre los resultados está encaminado a determinar su nivel de correspondencia con el resultado esperado. Son múltiples los artefactos que tributan a garantizar el control de estas salidas.

Ahora bien, una vez chequeados los resultados, pues lo más importante resulta ser entonces el registro del resultado del control de los mismos, la constancia de la gestión de la calidad de los resultados de cada disciplina. Pero no todo queda allí. Este es un punto importante del proceso de desarrollo de software que centra la atención de la dirección y por qué no, también del cliente. Por tanto, la garantía de la calidad de estos resultados reporta cierta seguridad tangible del desarrollo del proceso de software.

Es por eso entonces que se hace necesario tener en cuenta la manera de informar los resultados de la revisión en este momento y dar seguimiento en cierta medida a los mismos. En el orden en que la dirección conozca el criterio evaluativo de calidad interna pues contará con argumentos para tomar decisiones para futuras acciones del desarrollo del proceso de software. Por eso es muy importante que el equipo de calidad interna se trace tareas en función de lograr que los resultados sean informados y gestionados.

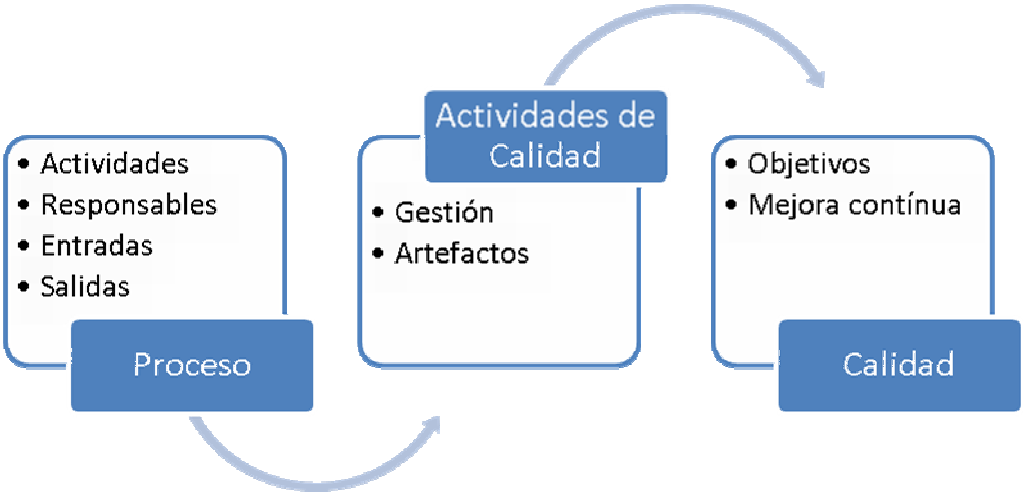



**FIG. 4 ENTORNO DE LAS ACTIVIDADES DE CALIDAD**

Hasta este momento se ha estado hablando de actividades en función de gestionar la calidad en cada momento del flujo de trabajo que definen cada disciplina. Ahora, para cada momento se relacionan artefactos, métodos y formas de lograr el cumplimiento de esas actividades por el equipo de calidad interna.

En la gestión del flujo de trabajo de la actividad de cada disciplina se pueden elaborar diagramas de flujo de actividades para conocer el subproceso. Para controlar dicha actividad, conocido el subproceso, se pueden realizar inspecciones para posteriormente informar los resultados de la inspección, registrarlos y tener propuestas a partir de criterios de variación y de una evaluación del desarrollo del flujo de trabajo acorde a como está definido.

En la gestión de las entradas de la actividad de cada disciplina se puede organizar un listado o relación de dichas entradas para su determinación y conocimiento. Se establecen parámetros en dependencia de sus características particulares, de modo que puedan ser evaluadas y tener criterio sobre las deficiencias que se puedan presentar. En un informe de deficiencias se recogen todas las dificultades presentadas y se establecen también acciones correctivas en función de las entradas, de modo que se garantice la gestión de variación. Las acciones correctivas también deben quedar reflejadas en un documento del equipo de calidad interno.

Para la gestión de las salidas existen varios procedimientos que se deben tener en cuenta para lograr un resultado óptimo. En primer lugar se deben realizar Revisiones Técnicas Formales a resultados específicos, software o documentación, indistintamente. Otro momento importante de la gestión de calidad interna sobre las salidas de los subprocesos, lo constituyen las pruebas internas de software. Previamente planificadas garantizan la detección de defectos en el producto desde pequeñas unidades constituyentes hasta un producto final y a lo largo del proceso de producción.

La discusión y aprobación por parte de la dirección del proyecto es un momento importante en los inicios de la gestión de la calidad interna en un proceso de desarrollo de software. La administración del equipo de calidad interna debe exponer a la dirección las tareas y actividades enfocadas en cada subproceso del proceso de desarrollo de software, de modo que la dirección pueda valorar y emitir criterios al respecto. Dichos criterios son valorados por la administración del equipo de calidad de modo que sirva para mejorar la propuesta en pos de responder a los intereses de la dirección. De esta manera se llega al momento de aprobar la propuesta a partir del cual se le da a conocer al equipo de desarrollo y a todos los involucrados para su conocimiento.

### *Organización del equipo de gestión de calidad interna del proyecto*

Como se ilustró en el capítulo anterior, la organización del equipo de calidad que se encargará de desarrollar el mecanismo de gestión de calidad interna del proyecto, es un punto relevante. Ahora, ¿cómo organizar el equipo? Pues bien, lo primero a tener en cuenta son las necesidades de la gestión de calidad para valorar la cantidad de personas que se necesiten para el proyecto. Las necesidades de gestión de calidad del proyecto están dadas por la definición de roles que van a estar delimitados por las actividades específicas que van a desempeñar y la base de conocimientos de la que deben partir. Se parte de determinar los roles de acuerdo a las actividades que va a desarrollar calidad interna para la gestión. Una vez determinados los roles se definen pues la base de conocimientos de la que debe partir. A partir de aquí y de acuerdo al volumen del proyecto, se determina entonces la cantidad de personas necesarias para desarrollar cada uno de los roles.

Se considera que no debe faltar el rol de Administrador de la Calidad o Responsable de la Calidad. Esta persona es la encargada de responder por la gestión de calidad ante la dirección del proyecto de acuerdo también a la estructura organizacional del proyecto. Desde el punto de vista de su papel en el

grupo de calidad, además de guiar el proceso de gestión, el Responsable de la calidad es una persona orientada al detalle que lleva la visión de todo el proceso. Asegura que la aplicación producida se ajusta a las especificaciones y está razonablemente libre de errores. Proporciona una metodología para realizar las pruebas, coordina las pruebas de calidad interna y evalúa los resultados que se obtienen en las pruebas de calidad. Vela por el cumplimiento de la calidad planificada.

El Responsable de la Calidad debe tener conocimientos básicos de la metodología de desarrollo RUP y el lenguaje de modelado UML (Unified Modeling Language), avanzados conocimientos de calidad e ingeniería de software y que posea además conocimientos básicos del negocio que se está modelando en el proceso de software.

Otro rol importante a considerar es el de Diseñador de Pruebas, encargado de diseñar los casos de prueba que se van a aplicar en el momento de las pruebas. Evalúa y documenta el resultado de las pruebas realizadas al software y además tiene la responsabilidad de definir las listas de chequeo que se deben aplicar. Sus conocimientos básicos deben estar orientados hacia la metodología RUP y el lenguaje de modelado UML, además de ser docto en los temas de pruebas de software.

Del mismo modo que una persona se especializa en el tema de las pruebas debe existir alguien que se encargue de los temas de las revisiones técnicas. Es entonces que se considera necesario se tenga en cuenta la definición del rol de Revisor Técnico que se encargará de chequear que los artefactos generados se ajusten a las pautas y lineamientos establecidos para su confección. Es el responsable de las revisiones técnicas y de toda la gestión respecto a las revisiones que se realicen en el proceso de desarrollo del software. Sus conocimientos estarán basados fundamentalmente en la metodología que propone RUP y el lenguaje de modelado UML.

Trabajadores como tal dentro de la gestión de calidad interna se pueden considerar a los que efectúan las pruebas. Dichas personas se desenvuelven por el rol de Probador con la responsabilidad de ejecutar las pruebas diseñadas y anotar los resultados obtenidos con el fin de realizar una gestión de los resultados en función de tratarlos de la manera más óptima posible. Se precisa para esta labor el conocimiento del negocio y habilidades básicas a la hora de operar con una computadora. En este papel se puede considerar además la presencia de un especialista en el negocio por nuestra parte que interpreta las necesidades del cliente y aporta mucho al equipo de desarrollo.

Una vez que se han definido los roles que intervendrán en la gestión de calidad interna del proyecto es preciso definir un plan de entrenamiento para lograr nivelar los conocimientos básicos necesario para realizar las actividades de gestión. La planificación del entrenamiento debe ser desarrollada por el Responsable de la Calidad teniendo en cuenta los conocimientos básicos requeridos por cada rol para su desempeño. El tiempo en que se planifique debe ser óptimo de manera que tribute al éxito de las actividades de gestión y pueden desarrollarse en la medida de las necesidades.

Se considera que entre los contenidos impartidos no deben faltar los temas relacionados con Evaluación de Software, Pruebas de Software y Métricas basados en normas y estándares internacionales, como propuesta inicial. A medida que se desarrolla el proyecto y en función de las necesidades, se podrán ir tramitando otros temas de capacitación relacionados con el negocio que se modela.

## *2.4 Organización del proceso de desarrollo para Registro y Notarías en su Segunda Fase.*

*Peculiaridades del proceso de desarrollo de Ingeniería de Software en Registro y Notarías previo a su segunda fase.*

En el proyecto Registro y Notarías, el proceso de desarrollo de software ha tenido sus peculiaridades. Luego de la ingeniería de requerimientos que se desarrolló en la etapa inicial de levantamiento de requisitos, se procedió directamente a la implementación de cada uno de los requerimientos sin el previo análisis. A medida que se iba implementando se documentaba, para luego pasar a la prueba y validación del producto.

Así, de manera iterativa, se refinaban los requisitos, se implementaban los cambios y se documentaba. En las pruebas se encontraban un número de defectos que tributaban a una nueva iteración de implementación y documentación.

Este proceso de desarrollo de Ingeniería de Software no se encontraba bien definido, resultado de la naturaleza cambiante del proyecto que se reflejaba de forma más relevante en las constantes

variaciones en las especificaciones del cliente y en características internas del equipo de desarrollo como fue la gran cantidad de estudiantes que participan y su necesidad de formación a la par de la producción, así como la poca experiencia del personal de dirección.

Inicialmente se concebía el desarrollo aquí en Cuba con todo el equipo concentrado. Posterior a esto surgió la necesidad de mover el equipo a Venezuela para trabajar de manera directa con el cliente, dando paso a una etapa en la que se documentaba allá, se implementaba en Cuba y se enviaban actualizaciones para ser probadas y así de manera repetida. El desarrollo del software transitó por todas las variantes y modalidades, atendiendo al lugar, organización del equipo de desarrollo y comunicación entre las partes. El proceso ha sido básicamente un proceso inestable.

*Organización del proceso de desarrollo de software en Registro y Notarías para su segunda fase.*

En la organización del proceso de desarrollo para Registro y Notarías en su segunda fase se definieron varios subprocesos con los artefactos o los resultados que se obtenían de cada uno y divididos por disciplinas del proceso y de soporte. Para las disciplinas de proceso se definieron sus objetivos, las actividades, los involucrados y los artefactos tanto de entradas como de salidas.

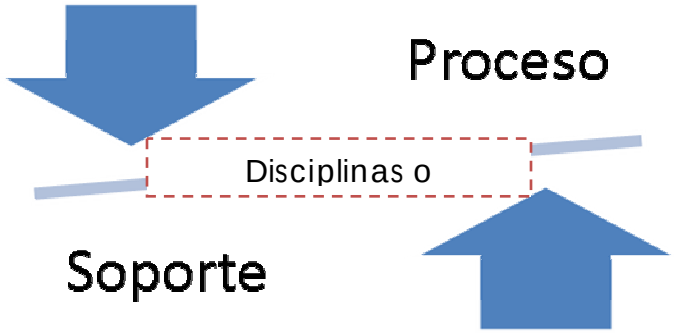



**FIG. 5 CLASIFICACIÓN DE LAS DISCIPLINAS: DE PROCESO Y DE SOPORTE**

Como primera disciplina del proceso se denominó el *Modelo de Negocio*, que tiene el objetivo de comprender el funcionamiento de la organización; garantizar que los clientes, usuarios finales, desarrolladores y las otras partes interesadas tengan un entendimiento común de la organización y derivar los requisitos de software necesarios para apoyar los procesos de la organización.

En el *Modelo de Negocio*, se definen una serie de actividades con el fin de cumplir con los objetivos de esta disciplina. Es en este subproceso que se realiza la recepción de la documentación de procesos, se actualiza el cronograma base, se elabora un plan de contactos con el cliente y se realiza la asimilación de la información entregada sobre el proyecto. Se recopila además la información de los procesos elementales del negocio y las revisiones técnicas formales y de aceptación con el comité de expertos. En estas últimas actividades, determinadas en el proceso de desarrollo es donde se involucra al Responsable de la Calidad.

Otros involucrados en esta disciplina son el Líder de Software, los Jefes de Sistema, el Ingeniero de Software Principal, el Arquitecto Principal, el Diseñador Principal de la Base de Datos, el Especialista Funcional. Participan además el Ingeniero en Procesos, el Analista del Sistema, Especialista en Información y el Planificador.

Para la realización de este subproceso es necesaria la gestión de las entradas. Entre ellas se destacan el Plan de Requisitos, el Proyecto Técnico, y el Cronograma de Captura de Requerimientos. Por parte del cliente es necesaria la gestión de los Mapas de procesos, el Organigrama, la Ficha de procesos y el Mapa de relaciones internas y externas. Se precisan además las Bases del Sistema de Información y la Identificación de funciones por clase de cargo, esta última por parte del cliente.

Como artefactos de salida generados en esta disciplina y sobre los que se centra fundamentalmente la atención de la gestión de calidad en cuanto a revisiones, se encuentran el Modelo de Negocio, el Documento de evaluación de las áreas de la organización, los Procesos elementales del negocio, la Base documental del negocio, el Plan de Requisitos y el Cronograma de Captura de Requerimientos.

La otra disciplina que le sigue al *Modelo de Negocio* es la que se denominó *Requisitos*. La misma se enfoca en establecer y mantener un acuerdo con los clientes y otros interesados en lo que el sistema debe hacer. Se centra además en proveer al equipo de desarrollo de un mejor entendimiento de los requisitos del sistema; en definir el alcance del sistema y proveer una base para planear el contenido de las iteraciones; en proveer una base para la estimación del costo y tiempo de desarrollo del sistema así como en definir una interfaz de usuario para el sistema, centrada en las necesidades del usuario.

En el subproceso de *Requisitos,* para alcanzar los objetivos anteriormente mencionados, la organización del proceso de desarrollo contempla la realización de determinadas actividades como son la actualización del cronograma para la propuesta del modelamiento del sistema; la definición de los requisitos del sistema, escribir los casos de uso del sistema y realizar revisiones internas formales y de validación de los requisitos del sistema. Es en esta disciplina que se realiza además un ajuste del cronograma, la elaboración del plan de contactos con el cliente, se elabora el modelo de casos de uso, el prototipo no funcional y las respectivas revisiones.

El punto de partida para desarrollar dichas actividades lo constituyen el Cronograma de captura de requerimientos, el glosario de términos y los procesos elementales del negocio que de conjunto con los involucrados se complementan en función de obtener los artefactos de salida. El Analista Principal junto al resto de los analistas, especialistas en Base de Datos y especialistas funcionales, así como el arquitecto principal, son los implicados en los *Requisitos,* para obtener un Documento preliminar de Casos de Uso, el Glosario de Términos, el documento de requerimientos, el Modelo de Casos de Uso del Sistema y el Documento de prototipo no funcional.

El *Análisis y Diseño*, es otra de las disciplinas definidas por la organización del proceso para el desarrollo del software. El fin que se persigue es el de transformar los requisitos en el diseño del sistema, desarrollar una arquitectura robusta para el sistema y adaptar el diseño al entorno de implementación. Para alcanzar dichos objetivos se realiza el Modelo de Análisis, el Modelo de Diseño y se define la Arquitectura del Sistema y diseña la Base de Datos de la aplicación. En este momento participan el Líder de Software, los Jefes de Sistema, los Analistas y Diseñadores de Sistema, el Equipo de Arquitectura, el de Base de Datos y los Especialistas Funcionales. Los mismos, partiendo del Documento preliminar de Casos de Uso, el Glosario de Términos, el Documento de Requerimientos, el Modelo de Casos de Uso del Sistema y el Documento de Prototipo no funcional, trabajan en función de obtener el Modelo de Análisis y el de Diseño, la Descripción de la Arquitectura, y el Modelo Físico de Datos Documento.

Otro subproceso definido en la organización del proceso de desarrollo es el definido como *Implementación*. Es en esta disciplina que se persigue definir la organización del código en capas, en términos de subsistemas de implementación. Implementar los elementos de diseño en términos de elementos de implementación constituye otro fin de la *Implementación*. Otro objetivo importante a alcanzar

es el de probar los componentes como unidades e integrar los resultados producidos por los implementadores individuales o equipos en un sistema ejecutable.

Como actividades para alcanzar los objetivos de esta disciplina se realizan los modelos de implementación y de despliegue y se implementan las funcionalidades correspondientes. Involucrados en las mismas se determinó la influencia de el Líder de Software, los Jefes de Sistema, el equipo de Base de Datos, el de Arquitectura y el de Programación, el Gestor de Cambios y los Escritores Técnicos. Se parte desde el Modelo de Diseño, la Descripción de la Arquitectura y del Modelo Físico de Datos Documento para lograr obtener el Modelo de Implementación y el de Despliegue.

El último subproceso definido dentro de la disciplina del proceso es el denominado *Prueba* el cual se encamina a evaluar la calidad del producto; encontrar y documentar los defectos del software; validar y probar las suposiciones hechas en el diseño y especificación de requisitos a través de una demostración concreta; validar que el producto de software trabaja según lo diseñado y validar que los requisitos fueron implementados apropiadamente. Se especifica que esta disciplina comprende las pruebas internas que realiza el equipo de desarrollo y las pruebas con el cliente. Según lo definido por el proceso de desarrollo, es en esta disciplina que se desarrollan las actividades encaminadas a elaborar el plan de pruebas, realizar el diseño de los casos de prueba y ejecutar los diferentes tipos de pruebas. Este subproceso involucra al Asegurador o Responsable de la Calidad, el Equipo de Pruebas, el Planificador, el Equipo de Desarrollo, los Escritores Técnicos, Especialistas Funcionales y Funcionales del Cliente.

Los artefactos que fluyen como entradas de esta disciplina son los Documentos de Requisitos, el Documento de Casos de Uso del Sistema y las Reglas del Negocio. Como artefactos resultantes se prevén el Plan de Pruebas, los Datos de las Pruebas, los Casos de Prueba y el Documento de Resultados de las Pruebas.


**FIG. 6 DISCIPLINAS DEL PROCESO**

Existen tres disciplinas más, definidas dentro del proceso de desarrollo del software para la segunda fase de Registro y Notarías, que son consideradas disciplinas de soporte. Estas son: *Gestión de Proyectos*, *Gestión de Configuración y Cambios*, y *Entorno.*

En la disciplina de *Gestión de Proyectos*, se pretende proveer de un framework para la administración del proyecto; proveer guías prácticas para la planificación, organización, ejecución y monitoreo del proyecto así como manejar los riesgos. Es responsabilidad del Líder de Software, Jefes de Sistema, Equipo de Dirección Central y Equipo de Dirección de Sistemas, perseguir dichos objetivos. Como artefactos de salida de este subproceso de soporte se tiene el Plan de Desarrollo de Software, Plan de Aseguramiento de la Calidad, Plan de Gestión de Riesgos, Plan de Capacitación, Plan de Iteración, Plan de Gestión de Proyectos, Cronograma de Ejecución del Proyecto y las Actas y Minutas de Reuniones de Avance y Talleres Técnicos. Esta es una disciplina no vinculada directamente en el proceso productivo pero su realización influye en un mejor desarrollo de la producción, garantizando la planificación y gestión de la misma. Es por tal motivo que, al igual que en el resto de las disciplinas de soporte, la gestión de la calidad interna del proyecto no debe ser ajena a estos subprocesos, basándose en su conocimiento y desempeño.

La *Gestión de Configuración y Cambios,* como disciplina, tiene los objetivos de mantener la integridad del proyecto; asegurar la completitud y correctitud del producto configurado; proveer de un

entorno estable dentro del cual se desarrolle el producto; restringir los cambios a los artefactos de acuerdo a las políticas del proyecto; proveer de una traza auditable para los cambios sobre los artefactos; establecer una línea base al final de cada iteración (un registro del estado de cada artefacto, estableciendo una versión), la cual podrá ser modificada sólo por una Solicitud de Cambio aprobada.

Una serie de actividades están destinadas a conseguir estos objetivos dentro de las que se destacan la definición de los elementos de configuración y la elaboración de planes como el de gestión de configuración, el de gestión de cambios, el de gestión de versiones y uno de auditoría. Encargados de realizar dichos planes se definieron el Gestor de Configuración, el Gestor de Cambios, el Asegurador de la Calidad, el Ingeniero de Software Principal, los Jefes de Sistema y el Líder de Software, quienes a partir de el Proyecto Técnico, tienen la responsabilidad de generar un Plan de Gestión de Configuración, un Plan de Gestión de Versiones, un Plan de Gestión de Cambios y un Plan de Auditorías, como artefactos de salida del subproceso.

Otros de los subprocesos de soporte al proceso de producción de software definidos para la segunda fase de Registro y Notarías, lo constituye la disciplina de *Entorno*. En este subproceso se pretende desarrollar las actividades necesarias para configurar el proceso de desarrollo del proyecto y definir tanto la organización como el entorno en el que se desarrollará el software, incluyendo los procesos y las herramientas, para lo cual se desarrollará la definición de la configuración del ambiente de desarrollo y la implementación de la configuración del ambiente de desarrollo.

En esta disciplina intervienen de forma decisiva el Líder de Software, los Jefes de Sistemas y el Equipo de Dirección Central, quienes partiendo del Proyecto Técnico, generarán como artefactos de salida, el Proceso Específico de Proyecto, las Herramientas y las Plantillas Específicas de Proyecto.

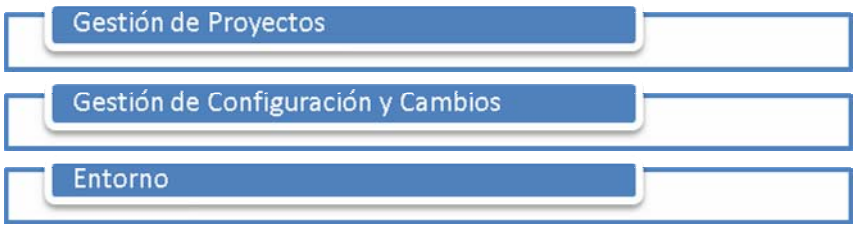



**FIG. 7 DISCIPLINAS DE SOPORTE**

En la organización del proceso de desarrollo se han definido otro conjunto de elementos que tributan a su organización (Anexo 1). Su conocimiento reporta una importancia vital para la gestión interna de calidad. Entre estos elementos se han definido unos principios para el desarrollo que se relacionan a continuación:

- Se realizarán iteraciones en la implementación de no más de 5 semanas.
- Se realizarán revisiones internas periódicas, todas las semanas en la las fases de inicio y elaboración.
- Cada etapa debe concluir con una planificación y cronograma detallado de la siguiente etapa.
- Capacitar a las personas en función de que todas las actividades se realicen de la mejor manera.
- Cada persona debe tener clara sus responsabilidades en cada una de las disciplinas, de los artefactos y las etapas.
- Rigurosa exigencia con el cumplimiento de las plantillas definidas y la manera que se ha establecido que se haga.

Otra de las definiciones presentes en la organización del proceso de desarrollo de Registro y Notarías, fase II lo constituye la relación Artefactos – Flujos – Fases. Por definir se encuentran algunos elementos tales como la relación Roles – Artefactos y el Diagrama de Flujo del proceso. Partiendo de la relación Artefactos – Flujos – Fases, se organizan las tareas y responsabilidades de la gestión de calidad para cada una de las fases del proceso de desarrollo, teniendo en cuenta los artefactos que se generan.

## *2.5 Lineamientos Propuestos por la Dirección de Calidad de la Infraestructura Productiva. Expediente de Proyecto.*

Según se define como punto de partida para la gestión de calidad de software a nivel de proyecto y como se reflejó en el capítulo anterior, deben ser adaptadas las distintas actividades y mantenimiento del software a las características concretas del proyecto y de su entorno, partiendo las guías que la Infraestructura Productiva prevé.

La Dirección de Calidad de la Infraestructura Productiva, encargada de dirigir y llevar a cabo la planificación y ejecución del control de calidad a los que deben ser sometidos cada proyecto de desarrollo de software en la universidad, propone un conjunto de modelos para la organización de un proyecto. Dichos modelos son recogidos en lo que se denomina Expediente de Proyecto.

Uno de los objetivos fundamentales enmarcados en el Expediente de Proyecto lo constituye el hecho de lograr homogeneidad en los proyectos productivos, en función de la certificación de calidad por el Laboratorio de Certificación. Precisamente constituye una entrada, como punto de partida para el proceso de certificación al que debe ser sometido cada producto.

Tres elementos fundamentales son los que interactúan en la gestión del expediente. Ellos son: Ingeniería, Gestión de Proyectos y Soporte.

*Ingeniería*

Dentro de la Ingeniería, el expediente relaciona varios modelos identificados a su vez o agrupados por categorías o subprocesos. Se define que en Requisitos debe contarse con el Plan de Gestión de Requisitos y el Modelo de Casos de Uso del Sistema. En Arquitectura y Diseño deben contemplarse los Patrones de arquitectura, los Patrones de diseño, Esbozo de la arquitectura, Modelo físico y Modelo lógico de los datos. Por parte de la Implementación y Pruebas se debe gestionar la Especificación de casos de prueba, el Plan de pruebas, Código fuente y Manual de usuario. Por su parte, el Modelo de despliegue y el Plan de instalación, constituyen otros modelos a gestionar dentro del Despliegue e Instalación como parte de la Ingeniería dentro del Expediente de Proyecto.

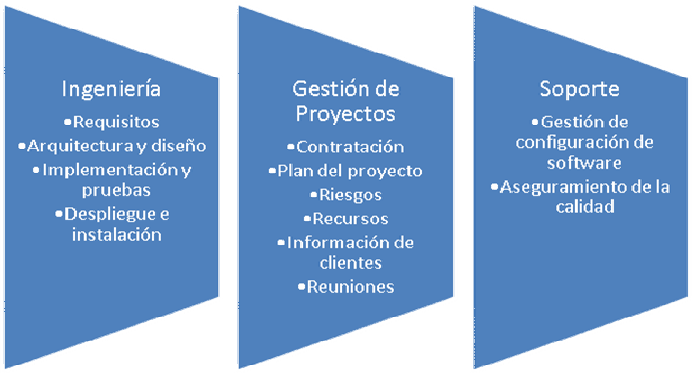



**FIG. 8 ELEMENTOS DEL EXPEDIENTE DE PROYECTO**

*Gestión de Proyectos*

En cuanto a Gestión de Proyectos dentro del expediente se relaciona que en el orden de *Contratación* debe contarse con el Documento Visión y el Plan de contratación de proveedores. Por parte del *Plan del proyecto* se deben gestionar los Cronogramas, el Presupuesto y el Plan de desarrollo y en cuanto a *Riesgos* debe existir una Lista de riesgos y un Plan de mitigación de riesgos. En el orden de los *Recursos* a ser gestionados, se deben detallar los Roles y responsabilidades, Plan de capacitación y Ambiente de desarrollo. En cuanto a Gestión de Proyectos en el expediente también se debe gestionar la *Información de clientes* y las *Reuniones*, como elementos que tributan también a la organización.

*Soporte*

Por último se relaciona el Soporte como otro punto en el que el Expediente de Proyecto hace énfasis. Dentro del Soporte, formando parte de la *Gestión de configuración de software*, se debe contar con una serie de elementos como es el caso de las Herramientas para la Gestión de configuración, el Plan de control de versiones, el Plan de control de cambio, Métricas, Plan de auditorías, Comité de control de cambios, Plan de gestión de configuración y un Listado de elementos de configuración. En lo que respecta a *Aseguramiento de la Calidad* se debe contar, según define la Dirección de Calidad de la Infraestructura Productiva, con Normas y estándares, Estilos de código, Plantillas, Plan de aseguramiento de la calidad, un Glosario, Registro y Respuesta a no conformidades.

Cada uno de los elementos relacionados anteriormente debe ser tenido en cuenta por la dirección del proyecto para su propia gestión. En cada uno de ellos se definen plantillas que organizan el proceso y favorecen a la organización, de modo que se logre homogeneizar la gestión de cada proyecto en función de la posterior certificación del producto.

# Capítulo 3: Propuesta de Solución

## *3.1 Introducción.*

En el presente capítulo se efectúa la propuesta de solución para la gestión de calidad interna del proyecto Registro y Notarías para su segunda fase. Se parte de un análisis previo, realizado en el capítulo anterior, de las características de la solución, basado en la identificación del proceso de desarrollo de software propuesto para esta fase. Las peculiaridades en el desarrollo del proyecto hasta la fecha, con las dificultades afrontadas y las experiencias, constituyen la fuente de esta propuesta de gestión. A partir de la definición de los objetivos de la calidad y de los intereses específicos de la dirección del proyecto, se plantean actividades y tareas enfocadas a lograr el cumplimiento de dichos objetivos, organizadas en un Plan de Aseguramiento de Calidad.

Un momento importante lo constituye la organización del equipo de gestión de calidad. Analizando las características del proyecto, se propone la estructura del equipo de gestión de calidad, definiendo los roles, las responsabilidades y base de conocimientos necesaria para su desempeño. A partir de ahí se propone además un Plan de Capacitación para la formación del equipo.

Se elabora esta propuesta analizando también los lineamientos que plantea la Dirección de Calidad de la Infraestructura Productiva.

## *3.2 Plan de Aseguramiento de Calidad para la Gestión de Calidad Interna.*

El Plan de Aseguramiento de la Calidad parte de la idea de lograr una planificación de las actividades que ejerzan influencia sobre el proceso de desarrollo del software de modo que gestionen su realización y la obtención de resultados acorde a los esperados. Se pretende una planificación lo más alcanzable posible y que se ajuste a las características del proyecto, teniendo en cuenta el mejor aprovechamiento de recursos y esfuerzos para su consecución.

En el plan, se consideran los siguientes elementos:

- Objetivos de Calidad.
- Organización. (Roles, Tareas y Actividades)
- Flujo de trabajo.
- Documentación.
- Métricas.
- Estándares y Guías.
- Cronograma.
- Herramientas, Técnicas y Metodologías.
- Resolución de problema y Acción Correctiva.
- Análisis de Riesgos.
- Plan de Revisiones.
- Pruebas y Evaluación.
- Gestión de Configuración.
- Registros de Calidad.
- Entrenamiento.

El ***Propósito*** que se persigue con este Plan de Aseguramiento de Calidad es el de describir cómo se asegurará la calidad del producto de forma general, lo artefactos, herramientas y procesos a través de los cuales se planifica lograr dicho propósito.

El ***Alcance*** del Plan de Aseguramiento de Calidad radica en que es elaborado a partir del propuesto por la metodología RUP, sólo que ha sido adaptado a las condiciones específicas del proyecto de forma general. Por tanto, su uso es muy particularmente del proyecto.

En el documento no existen ***Definiciones, Acrónimos ni Abreviaturas***. Aún no figuran referencias.

En ***Resumen***, el Plan de Aseguramiento de la Calidad es el documento rector donde se responsabilizan las partes y se organiza el trabajo del Grupo de Calidad del proyecto para lograr un mejor nivel de productividad. Se especifican los aspectos de cada etapa de desarrollo y en el Flujo de trabajo se

muestra un diagrama de los procesos que tendrán lugar para asegurar la calidad del producto durante el ciclo de vida del software en cuestión.

### *3.2.1 Objetivos de Calidad.*

Los objetivos de calidad que nos marcamos son los objetivos básicos del proyecto, asegurando su cumplimiento. Específicamente se propone lograr:

- Asegurar la calidad del trabajo, a través de la vigilancia, prevención, comprobación y valoración sistemática en el proyecto, a lo largo del ciclo de vida del mismo, velando porque el producto software cumpla con los requerimientos establecidos por el cliente.
- Mediante la gestión de riesgos identificar posibles errores antes de que se conviertan en puntos fatales o problemáticos.
- Mantener el trabajo sobre la base de los diferentes estándares y normas internacionales existentes, así como la metodología establecida por RUP.
- Velar y asegurar el desarrollo e implantación de un sistema informático que soporte las decisiones estratégicas del Cliente dentro del marco legal establecido.
- Corresponder y hacer cumplir con los lineamientos de calidad establecidos por Calisoft para los proyectos productivos de la UCI.
- Asegurar y contribuir al alcance de los objetivos trazados para la Solución Tecnológica del Documento Técnico del Proyecto.
- Lograr que el equipo de calidad cuente con el personal capacitado con el conocimiento y las habilidades necesarias para realizar las tareas y actividades encaminadas a lograr la calidad del proyecto.
- Colaborar con que la Gestión de Configuración y demás proceso de soporte sean desarrollados de tal manera que satisfaga las necesidades de la evolución del producto software y los procesos de producción del mismo.

### *3.2.2 Organización.*

#### *3.2.2.1 Roles.*

| Rol | Descripción | Conocimientos Mínimos | Nombre y Apellidos |
|---|---|---|---|
| **Responsable de Calidad** | - Es una persona orientada al detalle. Asegura que la aplicación producida se ajusta a las especificaciones y está razonablemente libre de errores.<br>- Proporciona una metodología para realizar las pruebas.<br>- Coordina las pruebas de calidad interna, las pruebas de aceptación del cliente y pilotos de conjunto con el Laboratorio de Certificación.<br>- Evalúa los resultados que se obtienen de las pruebas de calidad. | - Metodología RUP y UML.<br>- Calidad de Software.<br>- Ingeniería de Software.<br>- Conocimientos básicos sobre el negocio. | |
| **Diseñador de Pruebas** | - Diseña los casos de prueba.<br>- Evalúa y documenta el resultado de las pruebas realizadas al | - Metodología RUP y UML.<br>- Pruebas de Software. | |

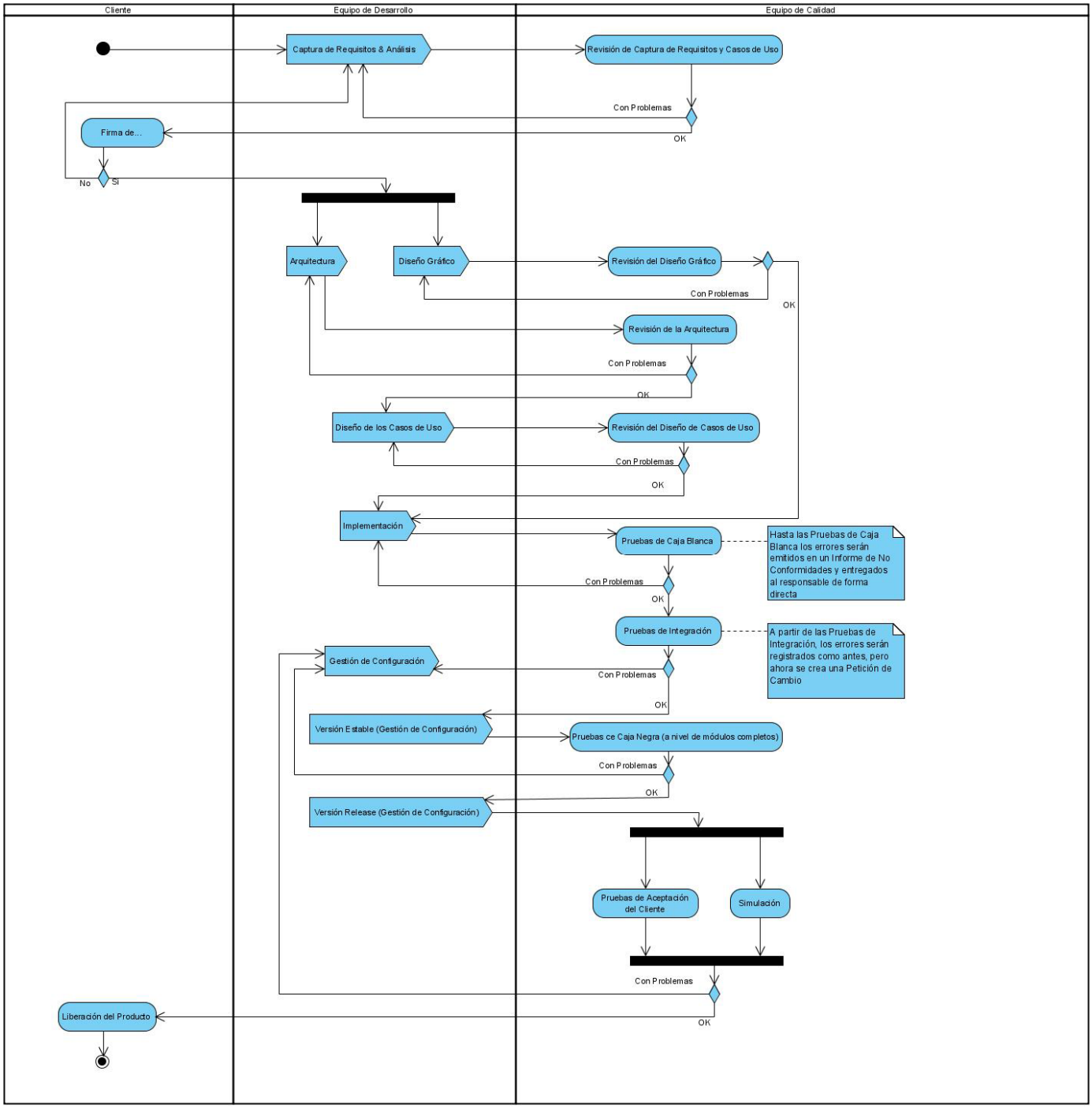
| | | | |
|---|---|---|---|
| | software.<br>- Define listas de chequeo. | | |
| **Revisor Técnico** | - Chuequea que los artefactos generados se ajusten a las pautas y lineamientos establecidos para su confección. | - Metodología RUP y UML. | |
| **Probador** | - Ejecuta las pruebas diseñadas.<br>- Anota los resultados obtenidos. | - Conocimientos del negocio.<br>- Habilidades mínimas de computación. | |

### *3.2.2.2 Tareas y Responsabilidades.*

| Tarea | Precondición | Poscondición | Responsable | Comentarios |
|---|---|---|---|---|
| **Revisión de la Captura de Requisitos** | Iniciado la Captura de Requisitos | | Equipo de Calidad | |
| **Revisión del Análisis** | | | Equipo de Calidad | |
| **Revisión de la Arquitectura** | | | Equipo de Calidad | |
| **Revisión de los Diseños Gráficos** | | | Equipo de Calidad | |
| **Revisión de Diseños de Casos** | | | Equipo de Calidad | |

| | | | | |
|---|---|---|---|---|
| **de Uso** | | | | |
| **Auditoría de la Base de Datos** | | | Equipo de Calidad | |
| **Pruebas de Caja Blanca** | | | Equipo de Calidad | |
| **Revisión de la Documentación** | | | Equipo de Calidad | |
| **Revisión de la Gestión de Configuración** | | | Equipo de Calidad | |
| **Pruebas de Caja Negra** | | | Equipo de Calidad | |
| **Pruebas de Integración** | | | Equipo de Calidad | |
| **Pruebas de Seguridad** | | | Equipo de Calidad | |
| **Pruebas de instaladores** | | | Equipo de Calidad | |
| **Pruebas de Aceptación del Cliente** | | | Equipo de Calidad | |
| **Simulación** | | | Equipo de Calidad | |

### *3.2.3 Flujo de Trabajo.*



### *3.2.4 Documentación.*

Para la confección del Plan de Aseguramiento de la Calidad se ha utilizado una serie de documentos rectores dentro del proyecto que a continuación se relacionan:

- Organización del Proceso de Desarrollo.

### *3.2.5 Métricas.*

Las métricas que se definen están encaminadas a analizar el comportamiento de los defectos en el proyecto. El objetivo fundamental es el de aportar argumentos para la toma de acciones preventivas que permitan mejorar el proceso de revisiones y el propio proceso de desarrollo de software, a partir del análisis de las métricas.

Se proponen un conjunto de métricas basadas en el artículo de la Revista Investigativa Operacional de la Dirección de Información Científico Técnica (DICT) de la Universidad de La Habana: “Una Propuesta de Introducción de las Revisiones en el Proceso de Desarrollo de Software” (7)

En este caso se realiza una propuesta de uso de métricas para el cumplimento de las metas de planificación; control y seguimiento; y mejoramiento. Para el cumplimiento de la meta de planificación se cuestiona cuánto cuesta y cuánto consume el proceso de revisión; en este sentido se utilizan las métricas Esfuerzo promedio por líneas de código fuente (ELC) y Eficiencia del inspector en la fase de preparación (EIP). Para la meta de control y seguimiento se cuestiona cuál es la calidad del software inspeccionado, en qué medida el personal técnico sigue los procedimientos establecidos para las revisiones y cuál es el estado del proceso de inspección; al efecto utilizamos las métricas de Promedio de defectos detectados por líneas de código fuente (DELC), Productividad de la revisión promedio (PR), Razón de preparación promedio de los inspectores (RPP), Densidad de defectos (DD), Promedio de líneas revisadas en cada revisión (PLCR), Porcentaje de reinspección, sólo para inspecciones (PRI), Eficiencia del inspector en la fase de preparación (EIP) y Cantidad total de líneas revisadas (TLCR). Por último para el cumplimiento de la meta de mejoramiento se cuestionará cuál es la productividad del proceso de revisión, y se propone la utilización de las métricas de Efectividad de eliminar los defectos en una revisión (EED) o la efectividad de Eliminar los defectos de la fase j en la revisión i (EEDi,j), el Promedio de defectos detectados por líneas de

código fuente (DELC), Productividad de la revisión promedio (PR), Razón de preparación promedio de los inspectores (RPP), Promedio de líneas revisadas en cada revisión (PLCR), Densidad de defectos (DD) y Esfuerzo promedio por defectos detectados (EDE).

| Meta | Pregunta | Métrica |
|---|---|---|
| **Planificación** | ¿Cuánto cuesta el proceso de revisión? | ELC |
| | ¿Cuánto tiempo consume el proceso de revisión? | ELC |
| | | EIP |
| **Control y Seguimiento** | ¿Cuál es la calidad del software inspeccionado? | DELC |
| | | PR |
| | | RPP |
| | | DD |
| | ¿En qué medida el personal técnico sigue los procedimientos establecidos para las revisiones? | PR |
| | | RPP |
| | | PLCR |
| | | PRI |
| | | EIP |
| | ¿Cuál es el estado del proceso de inspección? | TLCR |
| | | EIP |
| **Mejoramiento** | ¿Cuán efectivo es el proceso de revisión? | EED |
| | | $(EED_{i,j})$ |
| | | DELC |
| | | PR |
| | | RPP |
| | | PLCR |
| | | DD |
| | ¿Cuál es la productividad del proceso de revisión? | EDE |
| | | PR |
| | | RPP |
| | | PLCR |

### *3.2.6 Estándares y Guías.*

| Estándar | Etapa a aplicar | Comentarios |
|---|---|---|
| **IEEE 830** | Captura de Requisitos | |
| **ISO 12207** | | |
| **IEEE 1233** | | |
| **IEEE 1471** | Arquitectura | |
| **IEEE 1016** | Diseño Gráfico | |
| **ISO 9126** | Pruebas y Métricas | |
| **ISO 12119** | Documentación | |
| **IEEE 829** | | |
| **IEEE 1063** | | |
| **Lineamientos mínimos de calidad de la universidad** | Al proyecto de forma general | |
| **RUP** | Durante todo el ciclo de vida | |

### *3.2.7 Cronograma.*

El cronograma de las tareas y actividades de calidad se define partiendo de la organización, definición de tareas y responsabilidades y contando con los Planes de Iteración y Desarrollo del equipo de software.

### *3.2.8 Herramientas, Técnicas y Metodologías.*

Para el trabajo del grupo de calidad definimos las siguientes normas o estilos:

- Aplicación de los lineamientos mínimos de calidad (LMC) de la Universidad.
- Evaluación bajo las pautas antes establecidas por los responsables de cada área de trabajo.
- Aplicación de las normas internacionales. (IEEE, ISO, NC).
- Utilización de los conceptos de RUP.

- Aplicación de Listas de Chequeo confeccionadas por el responsable del rol determinado, a partir del Plan de Trabajo de dicho rol y las creadas por el Grupo de Calidad a partir de Normas Internacionales adaptadas.

### *3.2.9 Resolución de Problemas y Acción Correctiva.*

Al ser detectado cualquier error, problema o incongruencia en cualquiera de las etapas de revisión y pruebas, se levantará un documento titulado Informe de No Conformidades (Anexo 2, Anexo 3), donde será relacionada cada no conformidad encontrada. Este pasará al responsable del área en revisión con copia al responsable del grupo y al jefe del proyecto.

### *3.2.10 Análisis de Riesgos.*

El objetivo es realizar un análisis de los riesgos que se presentan en la gestión de calidad interna del proyecto centrándonos en las áreas del proceso de desarrollo de software que manejan los temas de aseguramiento de la calidad final.

- Riesgos relacionados con la entrega por parte del equipo de desarrollo.
- Riesgos relacionados con cronograma de actividades y tareas de calidad.
- Recursos humanos involucrados.
- Documentación generada por calidad interna.

**Estrategia para la gestión de los Riesgos**

**Evitar el riesgo:** Reorganizar el proyecto de manera que no sea afectado por riesgo alguno, muy difícil por no decir casi imposible.

**Transferir el riesgo:** Reorganizar el proyecto para que alguien más se encargue de gestionarlo, ese alguien podría ser el cliente, un vendedor, el banco, etc.

**Aceptar el riesgo:** Decidir vivir con el riesgo. Monitorear los síntomas del riesgo y elaborar un plan de contingencia, en caso de que se materialice éste.

Si se decide aceptar el riesgo, es necesario mitigarlo, es decir tomar alguna acción inmediata que reduzca su impacto.

La estrategia que se asume es: **Aceptar el riesgo**, por lo tanto, luego de identificar los riesgos que asumimos en nuestro proyecto, se propone realizar un análisis de los mismos y el impacto que cada uno de ellos acarrearía para la realización exitosa de las tareas y actividades, y a partir de aquí conformar un plan para mitigarlos, en un formato como el que aparece a continuación:

**Lista de riegos, análisis e impacto.**

| Magnitud del riesgo | Descripción e Impacto | Estrategia de mitigación \ Plan de Contingencia |
|---|---|---|
| | | |
| | | |
| | | |
| | | |

**Ponderación de los riesgos:** Luego se realizaría una ponderación de los mismos ordenándolos por orden descendente de impacto, es decir, de mayor a menor gravedad.

**Aplicación de Pareto:** Con la identificación de los riesgos involucrados en cada área de riesgo establecida y con el debido análisis del impacto y consecuencias que en el peor escenario que podría tener el riesgo una vez materializado se puede seleccionar el 20% de los riegos para tratarlos rigurosamente y así resolver el 80% de los problemas que estos pueden ocasionar.

**Continuidad del presente trabajo:** En lo adelante se debe trabajar en función de crear un plan de trabajo o cronograma que ayude a gestionar los riesgos determinados, colocando hitos para el análisis y replanteamiento del plan de mitigación de manera cíclica.

### *3.2.11 Plan de Revisiones.*

El Plan de Revisiones es un documento encargado de describir de forma específica cómo el Equipo de Calidad realizará las revisiones. Su planificación, flujos de trabajo, herramientas y funcionamiento de los roles, deben quedar muy bien definidos.

El plan es elaborado a partir de lo propuesto por RUP, sólo que ha sido adaptado a las condiciones específicas del proyecto de forma general. Por tanto, su uso es muy particular del proyecto.

Hace referencia al documento Plan de Listas de Chequeo. Se definen dos roles, el Responsable de Calidad y el Revisor Técnico, cada cual con su descripción y el conjunto de conocimientos mínimos para realizar la tarea.

El Responsable de Calidad del equipo es la persona que se encarga de planificar y organizar las revisiones, orientando los artefactos a revisar así como su entrega y las listas de chequeo que se utilizarán para ello.

El Revisor aplica las listas de chequeo orientadas, registrando los resultados de cada elemento.

El Responsable de Calidad conforma el Informe de No Conformidades con todos los problemas que se hayan detectado y lo hace llegar al responsable de sus soluciones.

A la par de esta actividad el Revisor aplica métricas a los resultados de las listas de chequeo.

Como última actividad el Responsable de Calidad realiza un resumen de los resultados a través de las métricas y se archivan los resultados para su posterior uso, quedando así concluido el flujo de trabajo de las revisiones.

### *3.2.12 Pruebas y Evaluación.*

Para las pruebas se realiza un Plan de Pruebas. En ese documento se describe de forma específica cómo el Equipo de Calidad realizará todas sus pruebas. Su planificación, flujos de trabajo, herramientas y funcionamiento de los roles, deben quedar muy bien definidos.

Es elaborado a partir del propuesto por RUP, sólo que ha sido adaptado a las condiciones específicas del proyecto de forma general. Por tanto, su uso es muy particular del proyecto.

Para las pruebas se definen tres roles fundamentales: el Responsable de Calidad, el Diseñador de Pruebas y el Probador. Se refleja su descripción y el conjunto de conocimientos mínimos para su buen desempeño. Además, se determina el flujo de trabajo que regirá la realización de dicha actividad.

### *3.2.13 Registros de Calidad.*

Los tipos de registros que se guardarán serán:

- Actas de reuniones efectuadas.
- Listas de Chequeo aplicadas.
- Casos de Prueba.
- Los Informes de No Conformidades.
- Resúmenes de Resultados.
- Correos de trabajo y comunicación de intercambio de trabajo.

Todos estos registros serán debidamente guardados en algo así como una bitácora.

### *3.2.14 Entrenamiento.*

El plan de entrenamiento del Grupo de Calidad, ha estado dado por el siguiente cronograma:

| Fecha | Objetivos | Resultados | Plan |
|---|---|---|---|
| | Curso de Capacitación Inicial sobre temas de Calidad. | Nivelación de los conocimientos del equipo. | Curso de Capacitación |

| | | | |
|---|---|---|---|
| | Profundización en los roles de RUP, estudio de Normas Internacionales, creación de Listas de Chequeo. | Elevación de los conocimientos del equipo, así como la creación de las herramientas necesarias para el trabajo. | |

# CONCLUSIONES

Con este trabajo se logró proponer un mecanismo de gestión de la calidad interna para la segunda fase del proyecto Registros y Notarías. Para lograrlo se desarrolló un Plan de Calidad para concebir las actividades a realizar para garantizar la gestión de calidad interna del proyecto Registro y Notarías, partiendo de la propuesta de desarrollo de software, en la segunda fase del mismo. Se organizó la estructura del equipo de calidad interna para el proyecto Registro y Notarías, con una definición de roles de acuerdo a lo que propone la metodología RUP y los modelos TSP y PSP, las responsabilidades de cada uno y el perfil de competencias sustentado por un plan de capacitación que se estableció.

Con la realización del trabajo, se pudo llegar a las siguientes conclusiones:

- La calidad de un producto de software o de un proceso de desarrollo de software no está en manos solamente de la gestión de calidad interna, sino además de la influencia de otras áreas de gestión, contempladas o no, en la existencia y aplicación de una metodología de desarrollo de software.
- Por otra parte, la presencia de una metodología de desarrollo de software no debe prescindir de un mecanismo de gestión de calidad interna que planifique, ejecute, controle y sea capaz de autogestionarse, en pro de tributar a la organización relativa al tema de la calidad.
- La gestión de calidad interna de un proceso de desarrollo de software no debe realizarse como fenómeno independiente, sino que debe surgir a partir de las necesidades que se plantee la administración del proyecto y teniendo en cuenta esas necesidades e intereses, pues trazarse el mecanismo siguiendo las concepciones en torno a la gestión de calidad de software.
- El mecanismo de gestión de calidad interna, en vista a lograr sus objetivos, debe presentarse, discutirse y aprobarse por la administración del proyecto y debe ser puesto en conocimiento de todo el equipo involucrado en el proceso de desarrollo de software de modo que se logre un mayor compromiso de las partes implicadas en los temas relativos a la calidad.
- La exactitud de la planificación que se realice por parte de la gestión de calidad interna en cuanto a cronogramas tiene un alto grado de dependencia de la fidelidad con que se planifique el proceso de desarrollo de software y de la medida en que sea más explícito el cronograma o plan de iteración del proyecto.

# RECOMENDACIONES

Establecer la propuesta de mecanismo de gestión de calidad interna en el proyecto Registro y Notarías y utilizarla como referencia para otros proyectos de desarrollo de software independientemente de su naturaleza.

Previo al establecimiento de la propuesta:

- Involucrar al equipo de desarrollo de software en cuanto a temas relativos a la calidad con la realización de talleres, de modo que conozcan la gestión de la calidad como concepción y sean capaces de asumirla y adaptarla como práctica cotidiana en su desempeño.
- Utilizar métodos y estrategias para la presentación, discusión y aprobación de la propuesta de calidad interna de un proyecto de software de modo que se logre el conocimiento de todos los implicados y más que eso, el compromiso con la calidad.
- Estudiar sobre técnicas y mecanismos para lograr asimilar de forma factible, los intereses de la administración de los proyectos de desarrollo de software en los temas relativos a la calidad y aplicar dichos mecanismos para lograr mejores resultados relativos a la gestión interna.

Durante la implementación:

- Realizar planificaciones en ciclos cortos o a corto plazo en cuanto a actividades de gestión de calidad, de modo que permita que las mismas sean más exactas y fieles para evitar desgaste en replanificaciones.

Para ampliar la investigación:

- Estudiar sobre otras áreas de gestión, así como metodologías de desarrollo de software que reporten un nivel alto de organización en función de lograr índices aceptables de calidad en los productos y en los procesos de desarrollo de software.
- Plantearse como objetos de estudio para futuras investigaciones, las metodologías de desarrollo de software y la gestión de procesos y de proyecto, para fortalecer otras áreas que tributan a la calidad de software.

BIBLIOGRAFÍA

1. **Scalone, Lic. Fernanda.** *Estudio comparativo de los modelos y estándares de calidad de software.* Buenos Aires : Universidad Tecnológica Nacional. Facultad Regional Buenos Aires, 2006.

2. **Antonio, Angélica de.** Instituto de Informática de la Universidad Austral de Chile. *Gestión, Control y Garantía de la Calidad del Software.* [En línea] 2007. [Citado el: 6 de febrero de 2007.] www.inf.uach.cl/rvega/asignaturas/info265/G_Calidad.pdf.

3. **Informática, Departamento de Control de Calidad y Auditoría.** Sistemas. [En línea] 31 de agosto de 2000. [Citado el: 13 de abril de 2007.] http://sistemas.dgsca.unam.mx/publica/pdf/Control%20de%20Calidad.PDF.

4. **Luzuriaga, Juan Manuel.** Monografías.com. [En línea] 1997. [Citado el: 16 de abril de 2007.] http://www.monografias.com/trabajos6/isof/isof.shtml.

5. **Pressman, Roger S.** *Ingeniería del Software. Un enfoque práctico.* La Habana : Félix Varela, 2005. Vol. 1.

6. **Sommerville, Ian.** *Software Engineering.* s.l. : Addison- Wesley, 2004.

7. *Una Propuesta de Introducción de las Revisiones en el Proceso de Desarrollo de Software.* **Delgado Dapena, Martha Dunia, Álvarez Cárdenas, Sofía y Rosete Suárez, Alejandro.** Ciudad de La Habana, Cuba : Revista Investigación Operacional, 2005.

8. **Domínguez, Guillermo y Lozano, Luz.** El concepto de calidad y su evolución. [aut. libro] Instituto Nacional de Empleo. *Calidad y Formación: Binomio Inseparable.* Madrid : Publicaciones Madrid, 2003.

9. **VELA, JOSE DE JESUS RODRIGUEZ.** Gestiopolis.com. *ADMINISTRACIÓN DE PROYECTOS DE DESARROLLO DE SISTEMAS DE INFORMACIÓN .* [En línea] Carlos López / Webprofit Ltda., 2007. [Citado el: 20 de abril de 2007.] http://www.gestiopolis.com/recursos2/documentos/fulldocs/ger/adproysisinf.htm#_ftnref49.

10. **Seen, James A.** Instituto tecnológico de Chihuahua II. *CONTENIDO DE RESIDENCIAS PROFESIONALES EN AL ÁREA DE SOFTWARE.* [En línea] [Citado el: 20 de abril de 2007.] http://www.itchihuahuaii.edu.mx/etec/data/Documentos/ResProfesYTitul/.

11. **Malevski, Yoram.** *Manual de gestión de la calidad total a la medida.* Guatemala : 1a. ed.- Guatemala, GT: Piedra Santa, 1995. OEA.

12. **Sánchez, Abigail.** Monografías.com. *Historia de la calidad total.* [En línea] [Citado el: 6 de febrero de 2007.] http://www.monografias.com/trabajos7/catol/catol2.shtml.

13. **Castañeda, Christian.** Monografías.com. *Ciclo de vida de un sistema de información.* [En línea] 1997. [Citado el: 20 de abril de 2007.] http://www.monografias.com/trabajos29/ciclo-sistema/ciclo-sistema.shtml.

14. **Galáz, Solange.** Monografías.com. *Ingeniería de Software.* [En línea] 1997. [Citado el: 20 de abril de 2007.] http://www.monografias.com/trabajos5/inso/inso.shtml.

15. **Palacio, Juan.** Navegapolis.net. *Gestión y procesos en empresas de software.* [En línea] Noviembre de 2005. [Citado el: 15 de enero de 2007.] http://www.navegapolis.net/content/view/156/59/.

# ANEXOS

*Anexo 1*

ORGANIZACIÓN PRELIMINAR DEL PROCESO DE DESARROLLO PARA REGISTRO Y NOTARÍAS FASE II

Principios para el desarrollo:

- Se realizarán iteraciones en la implementación de no más de 5 semanas.
- Se realizarán revisiones internas periódicas, todas las semanas en la las fases de inicio y elaboración.
- Cada etapa debe concluir con una planificación y cronograma detallado de la siguiente etapa.
- Capacitar a las personas en función de que de todas las actividades se realicen de la mejor manera.
- Cada persona debe tener clara sus responsabilidades en cada una de las disciplinas, de los artefactos y las etapas.
- Rigurosa exigencia con el cumplimiento de las plantillas definidas y la manera que se ha definido que se haga.

Definir:

- Relación Artefactos – Flujos – Fases
- Relación de Roles – Artefactos
- Diagrama de flujo del proceso

Relación Artefactos – Flujos – Fases

| **Disciplina** | **Artefacto** | **Fases** | | | |
|---|---|---|---|---|---|
| | | **Inicio**<br><br>**I1** | **Elab**<br><br>**E1…En** | **Const**<br><br>**C1…Cn** | **Trans**<br><br>**T1…Tn** |
| **Disciplinas del proceso** | | | | | |
| **Modelo de Negocio** | Documento de evaluación de las áreas de la organización | c | | | |
| | Base documental del negocio | c | | | |
| | Procesos Elementales de Negocio | c | r | | |
| **Requerimientos** | Plan de Requisitos | c | | | |
| | Glosario de Términos | c | r | | |
| | Documento de Requerimiento | c | r | r | |
| | Modelo de Casos de Uso del Sistema | c | r | r | |
| **Análisis y Diseño** | Documento de Prototipo no Funcional | | c | r | |
| | Modelo lógico de los datos | c | r | r | |
| | Modelo de Diseño | | c | r | |
| | Descripción de la Arquitectura | | c | r | |
| | Modelo Físico de Datos | | c | r | |
| **Implementación** | Modelo de Implementación | | c | r | r |
| | Modelo de Despliegue | | c | r | r |
| **Prueba** | Plan de Pruebas | | c | r | r |
| | Plan de Revisiones técnicas | c | r | r | r |
| | Datos de las pruebas | Se obtiene en cada iteración | | | |
| | Casos de Prueba | Se obtiene en cada iteración | | | |
| | Documento de Resultado de las pruebas | Se obtiene en cada iteración | | | |

| | | | | | |
|---|---|---|---|---|---|
| | Documento resultado de las revisiones técnicas | Se obtiene en cada iteración | | | |
| **Disciplinas de Soporte** | | | | | |
| **Gestión de Proyectos** | Plan de Desarrollo de Software | c | r | r | r |
| | Plan de Aseguramiento de la Calidad | c | r | r | r |
| | Plan de Gestión de Riesgos | c | r | r | r |
| | Plan de Capacitación | c | r | r | r |
| | Plan de Iteración | Se obtiene en cada iteración | | | |
| | Plan de gestión de proyectos | c | r | r | r |
| | Cronograma de ejecución del proyecto | c | r | r | r |
| | Actas y minutas de reuniones de avance y talleres técnicos | Se obtiene en cada iteración | | | |
| **Gestión de Configuración y Cambio** | Plan de Gestión de Configuración | c | r | r | |
| | Plan de Control de Versiones | c | r | r | |
| | Plan de Gestión de Cambios | c | r | r | |
| | Plan de Auditorías | c | | | |
| | Solicitud de Cambio | Se obtiene en cada iteración | | | |
| **Entorno** | Proceso Específico del Proyecto | c | r | r | |
| | Herramientas | c | | | |
| | Plantillas Específicas de Proyecto | c | r | r | |

Relación Artefactos – Roles

| **Artefacto** | **Rol Responsable** | **Roles que intervienen** |
|---|---|---|
| | | |
| | | |
| | | |

Planificación de las Fases

| **No.** | **Fases** | **Etapa 1** | |
|---|---|---|---|
| | | **Cantidad de Iteraciones** | **Duración (semanas)** |
| | | | |
| | | | |

Hitos definidos para la Etapa

| **No.** | **Hito** | **Duración en semanas** |
|---|---|---|
| | | |
| | | |

Diagrama de Flujo del Proceso

*Anexo 2*

*PLANTILLA DE INFORME DE NO CONFORMIDADES*

**RN**
**Informe de No Conformidades**
**Versión 1.0**

*[Documento oficial donde se registrarán todos los errores y/o sugerencias encontradas durante las pruebas o las revisiones efectuadas]*

*[FORMA DE USO:*

- *la casilla No. debe ser llenada forma consecutiva*
- *el Elemento evaluado es el objeto real. Ejemplo: documentación, aplicación, etc.*
- *el Aspecto correspondiente es la parte de ese Elemento evaluado. Ejemplo: formato, redacción, ortografía, contenido; interfaz X, etc.*
- *en la Descripción de la No Conformidad, se profundizará en el aspecto.*
- *en el Tipo se debe definir si es Importante o Sugerencia la no conformidad tratada.*
- *en Responsable de Solución debe hacerse referencia al mismo, así como su firma de conocimiento de la misma.*
- *en Versión se debe referenciar el número de versión que se revisa.*

*]*

<Número de registro>
Fecha:
Etapa de la Detección:

| No. | Elemento evaluado | Aspecto correspondiente | Descripción de la No Conformidad | Tipo | Responsable de Solución | Versión |
|---|---|---|---|---|---|---|
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

Anexos

*[El uso de los anexos es fundamental, ya que cuando se encuentre el error debe tirársele una foto (utilizar el Print Screen) para ayudar a la mejor comprensión del problema.*

*Estos se asociarán por el No. Correspondiente a la No Conformidad en el pie de la figura insertada]*

*Anexo 3*

*PLANTILLA DE RESULTADOS DE PRUEBAS*

*[Después de aplicadas las métricas a todas iteraciones del Caso de Uso Pruebas se deben resumir los resultados, para ello sugerimos el uso de la siguiente tabla]*

| **Característica de Calidad** | **Subcaracterística de Calidad** | **Cantidad de Iteraciones** | **% Cumplimiento** |
|---|---|---|---|
| | | | |
| | | | |
| | | | |
| | | | |

Anexo 4

*PLAN DE REVISIONES*

**RN**
**Plan de Revisiones**
**Versión 1.1.0**

Revisiones históricas

| Fecha | Versión | Descripción | Autor |
|---|---|---|---|
| | | | |

Tabla de Contenidos

Plan de Revisiones

1. Introducción.
    - 1.1. Propósito.
      Este documento es el encargado de describir de forma específica, cómo el Equipo de Calidad realizará las revisiones. Su planificación, flujos de trabajo, herramientas y funcionamiento de los roles, deben quedar muy bien definidos.
    - 1.2. Alcance.
      Este Plan es elaborado a partir del propuesto por RUP, sólo que ha sido adaptado a las condiciones específicas del proyecto de forma general. Por tanto, su uso es muy particular del proyecto.
    - 1.3. Definiciones, Acrónimos y Abreviaturas.
      No existen Definiciones, Acrónimos ni Abreviaturas en este documento.
    - 1.4. Referencias.
      Plan de Listas de Chequeo.
    - 1.5. Resumen.
      *[Resumen de los aspectos del plan]*

2. Organización.

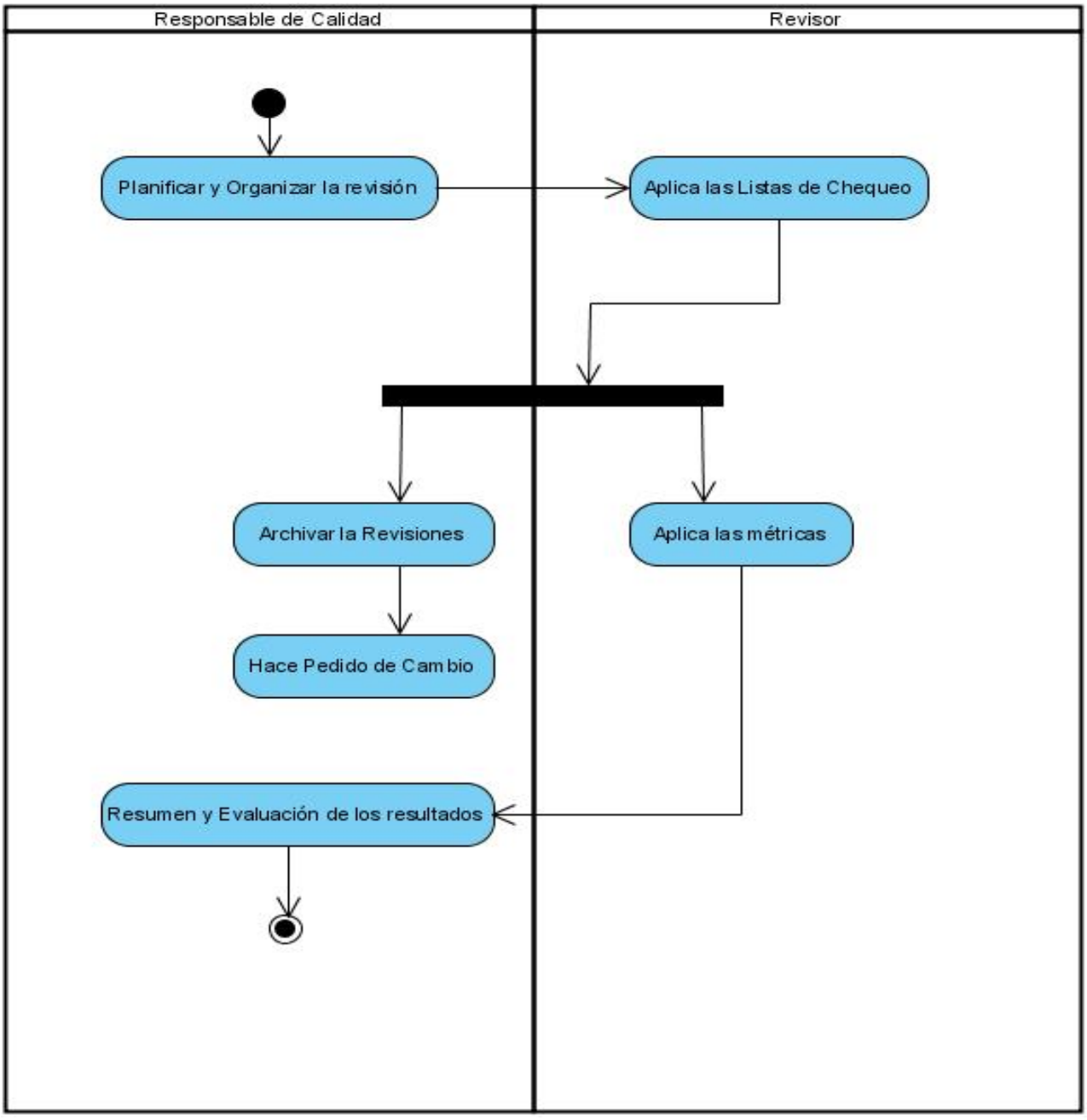
| Rol | Descripción | Conocimientos mínimos |
|---|---|---|
| Responsable de Calidad | • Es una persona orientada al detalle. Asegura que la aplicación producida se ajusta a las especificaciones y está razonablemente libre de errores. | • RUP y UML<br>• Calidad de Software.<br>• Ingeniería de Software.<br>• Posee al menos |

| | | |
|---|---|---|
| | • Proporciona una metodología para realizar las pruebas.<br>• Coordina las pruebas de calidad internas, las pruebas de aceptación del cliente y pilotos.<br>• Evalúa los resultados que se obtienen en las pruebas de calidad. | conocimientos básicos sobre el negocio. |
| Revisor Técnico | • Chequea que los artefactos generados se ajusten a las pautas y lineamientos establecidos para su confección. | • RUP y UML |

3. Plan de elaboración de Listas de Chequeo.

Esta planificación la hemos recogido en un documento al cual hacemos referencia. Plan de Listas de Chequeo.

4. Flujo de Trabajo.




5. Metodología de aplicación.

El responsable de calidad del equipo es la persona que se encarga de planificar y organizar las revisiones, orientando los artefactos a revisar así como su entrega y las lista de chequeo que se utilizarán para ello.

El revisor aplica las listas de chequeo orientadas, registrando los resultados de cada elemento.

El responsable de calidad conforma el Informe de No Conformidades con todos los problemas que se hayan detectado y las hace llegar al responsable de sus soluciones.

A la par de esta actividad el revisor aplica métricas a los resultados de las listas de chequeo.

Como última actividad el responsable de calidad realiza un resumen de los resultados a través de las métricas y se archivan los resultados para su uso posterior. Queda así concluido el flujo de trabajo de las revisiones.

6. Herramientas.

Inicialmente se usarán las Listas de Chequeo.

*Anexo 5*

PLAN DE LISTAS DE CHEQUEO

**RN**
**Plan de Listas de Chequeo**
**Versión 1.0**

1. Introducción

En el presente documento se pretende planificar de forma general todas las Listas de Chequeo que se deben confeccionar para ser aplicadas por parte del Equipo de Calidad durante el proceso de desarrollo de software.

2. Planificación

La creación de las Listas de Chequeo estaría dada por la etapa a la que se le aplicará, el Plan de Trabajo, las recomendadas por parte del especialista de esa área, las normas existentes que se pretendan aplicar, las que nos recomienda RUP y la cantidad de entregables existentes, además de otras que puedan surgir y nos sean útiles.

Para ello se muestra una tabla a continuación donde se pretende estimar dicha información:

| Disciplina | Cant LC | Artefacto | Responsable del artefacto | Listas de chequeo | Bases de la LC | Aplicada por | Responsable de revisiones. |
|---|---|---|---|---|---|---|---|
| **Captura de requisitos** | 5 | Plan de Captura de Requisitos | Analistas | LC Plan de Captura de Requisitos | -IEEE 830_1998<br>-IEEE 1233_1998 | | |
| | | Modelos de Casos de Uso | Analistas | LC Modelos de Casos de Uso | -RUP | | |
| | | Especificaciones de casos de uso | Analistas | LC Especificaciones de Casos de Uso | -RUP | | |
| | | Especificaciones Suplementarias | Analistas y Arquitecto Principal | LC Especificaciones Suplementarias | -RUP | | |
| | | Diccionario de Datos | Analistas | LC Diccionario de Datos | | | |
| **Diseño y Arquitectura** | 3 | Descripción de la Arquitectura | Arquitecto Principal | LC Descripción de la Arquitectura | -RUP | | |
| | | Modelo de Diseño | Arquitecto Principal | LC Modelo de Diseño | -RUP | | |
| | | Prototipo de Interfaz de usuario | Diseñador de Interfaz de Usuario | LC Prototipo de interfaz de usuario | -Pautas de diseño<br>-Prototipo | | |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | | | | elemental de Interfaz de Usuario. | | |
| **Base de Datos** | 2 | Modelo Lógico de Datos | Diseñador principal de Base de Datos | LC Modelo Lógico de Datos | -RUP (parte del modelo de datos) | | |
| | | Modelo Físico de Datos | Diseñador Principal de Interfaz de Usuario | LC Modelo Físico de Datos | -RUP (parte del modelo de datos) | | |
| **Implementación** | 1 | Modelo de Implementación | Diseñador | LC Modelo de Implementación | -RUP | | |
| | | Elementos de Implementación | Implementador | LC Elementos de Implementación | - Políticas de Implementación del proyecto | | |
| **Documentación** | 2 | Glosario de términos | Documentador | | | | |
| | | Material de soporte al usuario final | Documentador | LC Material de soporte al usuario final | | | |
| **Calidad** | 2 | Casos de prueba | Responsable de Calidad | LC Casos de prueba | -RUP | | |
| | | Plan de | Responsable de | LC Plan de pruebas | -RUP | | |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | Pruebas | Calidad | | | | |
| **Gestión de Configuración** | 1 | Plan de Gestión de Configuración | Responsable de Gestión de Configuración | LC Plan de Gestión de Configuración | -Una que prepare calidad | | |
| **Administración del proyecto** | 1 | Plan de Iteración | Jefe de Proyecto | LC Plan de Iteración | | | |
| | 1 | Artefactos de Instalación | Arquitecto Principal | LC Artefactos de Instalación | | | |
| **Total de LC** | **18** | | | | | | |

Anexo 6

*PLANTILLA DE LISTAS DE CHEQUEO*

**RN**
**Listas de Chequeo de** <*poner el nombre correspondiente*>
**Versión 1.0**

*[Documento para la confección de Listas de Chequeo]*
*[**Forma de Uso:***

- *en el **Nivel** solo se define cuando existen aspecto de mayor importancia que otros, para ello se usarán signos de exclamación (!), entre más tenga más importante es*
- *en el **Criterio de evaluación** se ubicará el aspecto a evaluar, siempre con una redacción nítida*
- ***Evaluación** es para el caso de la aplicación de la Lista de Chequeo. Dicha evaluación se encontrará en el rango de 0-5*
- ***N.P.** significa No Procede y es para el caso de la aplicación que ese aspecto no sea factible su valoración*
- ***Observaciones** es para cualquier cosa que quiera incluir la persona que aplica dicha lista*
- *La parte de **Métricas, Submétricas y % cumplimiento** es para la aplicación de métricas a las iteraciones realizadas*

*Estos comentarios en azul deben borrarse a la hora de entregar finalmente este documento de forma oficial.]*

Fecha:
Responsable que la aplicó:

| **Nivel** | **Criterio de evaluación** | **Evaluación** | **N. P.** | **Observaciones** | **Métrica** | **Submétrica** | **%cumplimiento** |
|---|---|---|---|---|---|---|---|
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |

*Anexo 7*

PLAN DE PRUEBAS

**RN**
**Plan de Pruebas**
**Versión 1.1.0**

Revisiones históricas

| Fecha | Versión | Descripción | Autor |
|---|---|---|---|
| | | | |

Tabla de Contenidos

Plan de Pruebas

1. Introducción

    1.6. Propósito.

    Este documento es el encargado de describir de forma específica, cómo el Equipo de Calidad realizará todas sus pruebas internas. Su planificación, flujos de trabajos, herramientas y funcionamiento de los roles, deben quedar muy bien definidos.

    1.7. Alcance.

    Este Plan es elaborado a partir del propuesto por RUP, sólo que ha sido adaptado a las condiciones específicas del proyecto de forma general. Por tanto, su uso es muy particularmente del proyecto.

    1.8. Definiciones, Acrónimos y Abreviaturas.

    No existen Definiciones, Acrónimos y Abreviaturas en este documento.

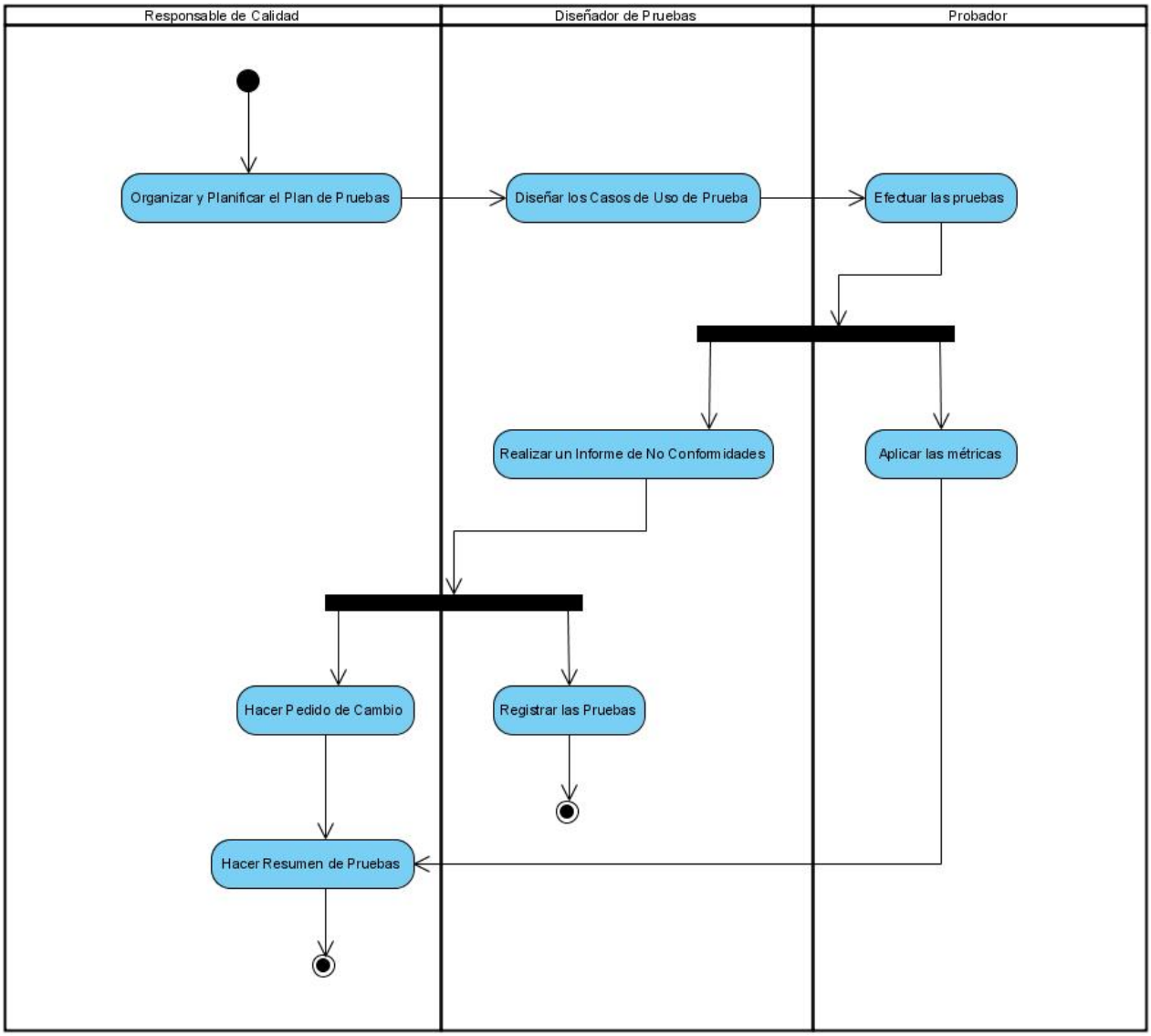
    1.9. Referencias.
    Plan de Desarrollo del Proyecto.

    1.10. Resumen

2. Organización.

| Rol | Descripción | Conocimientos mínimos |
|---|---|---|
| Responsable de calidad | • Es una persona orientada al detalle. Asegura que la aplicación producida se ajusta a las especificaciones y está razonablemente libre de errores.<br>• Proporciona una metodología para realizar las pruebas.<br>• Coordina las pruebas de calidad internas, las pruebas de aceptación del cliente y pilotos.<br>• Evalúa los resultados que se obtienen en las pruebas de calidad. | • RUP y UML<br>• Calidad de Software.<br>• Ingeniería de Software.<br>• Posee al menos conocimientos básicos sobre el negocio. |
| Diseñador de pruebas | • Diseña los casos de prueba.<br>• Evalúa y documenta el resultado de las pruebas realizadas al software.<br>• Define listas de chequeo. | • RUP y UML<br>• Pruebas de software. |
| Probador | • Ejecuta las pruebas diseñadas.<br>• Anota los resultados obtenidos. | • Conocimientos del negocio.<br>• Habilidades mínimas de computación. |

3. Flujo de Trabajo




4. Metodología de aplicación.
5. Pruebas de Unitarias.
6. Pruebas de Caja Negra.
7. Pruebas de Integración.

8. Pruebas de Instaladores.

Anexos

A. Plantillas de Casos de Uso de Pruebas de Caja Negra.

B. Plantillas de Casos de Uso de Pruebas de Integración.

C. Plantillas de Casos de Uso de Pruebas de Instaladores.

Anexo 8

*CURSO DE CAPACITACIÓN INICIAL SOBRE TEMAS DE CALIDAD*

| **Local** | **Activi dad** | **Tipo de Actividad** | **Tema** | **Contenido** | **Tipo Eval.** |
|---|---|---|---|---|---|
| Aula | 1/2 | Conferencia | Evaluación de Software | - Rol que desempeña la evaluación en el ciclo de vida de un software. Su papel en el proyecto.<br>- Composición de paquete software<br>- Características de calidad de paquete software (dado por normas internacionales)<br>- Instrucciones para la realización de pruebas/ensayos | Preg. Escrita |
| Aula | 3/4 | Seminario | Pruebas | - Tipos de pruebas. Su importancia | Seminario |
| Aula | 5/6 | Conferencia teórico práctica | | - Introducción a las características generales del flujo de trabajo del grupo responsable de calidad.<br>- Organización de un equipo de trabajo de prueba.<br>- Definición de roles.<br>- Requerimientos, asociación con un CU.<br>- Evaluación de los CU.<br>- Utilización de listas de chequeo y su conformación | Preg. Escrita |

| | | | | | |
|---|---|---|---|---|---|
| Laboratorio | 7/8 | Clase práctica | | - Diseño de Casos de Uso Pruebas<br>- Reporte de no conformidad.<br>- Cuaderno de trabajo.<br>- Aplicación de listas de chequeo. | Preg. Escrita |
| Aula | 9/10 | Conferencia | Métricas | - Características de calidad externas de un producto final y su importancia en la evaluación<br>- Métricas y su aplicación. | |

*Anexo 9*

DEFINICIÓN DE LAS MÉTRICAS PARA MEDICIONES

1. Efectividad de eliminar los defectos en una Revisión:

$$EED = \frac{DE_i}{DL} * 100$$

$$DL = DE_i + DEP$$

$$DEP = \sum_{k=i+1}^{n} DE_k$$

DE: cantidad de defectos detectados durante la revisión.

DEP: cantidad de defectos encontrados posterior a la revisión, es decir la cantidad de defecto encontrados en las n-i restantes revisiones que se indican en el plan de revisiones y auditorías del proyecto. También puede calcularse este valor considerando los defectos detectados en las revisiones efectuadas hasta el momento (k = m, con i < m < n) en que se desea analizar la métrica, pero serán resultados parciales que pueden cambiar al finalizar el producto.

DL: cantidad total de defectos presentes en el producto, cuando éste ha sido terminado y se entrega al cliente para su operación.

Si se representan en un gráfico los valores resultantes de la métrica de efectividad se puede analizar qué revisiones de las realizadas al Proyecto de Software resultan poco efectivas y así valorar la posibilidad de incluirla o no en la misma fase del Desarrollo del Proyecto o en otros cuyas características sean similares al analizado o tomar cualquier otra decisión que contribuya al mejoramiento del Proceso de Revisiones.

Ahora bien, esta medida da una vista global de la efectividad de la revisión, pero en ocasiones no basta con esta información y es necesario profundizar para conocer cuáles tipos de errores no han sido detectados y que por tanto conspiran contra la efectividad de dicha Revisión, para ello se propone la métrica siguiente.

2. Efectividad de eliminar los defectos de la fase j en la revisión i.

Permite a los directivos conocer la efectividad de las revisiones en cuanto a la cantidad de defectos que pertenecen a una fase y que son encontrados oportunamente. Por tanto, se puede analizar la cantidad de defectos, por ejemplo de la fase de requisitos, que se han propagado hasta la implantación del sistema o hasta cualquier otra fase de desarrollo del proyecto.

$$EED_{i,j} = \frac{DE_{i,j}}{DL_j} * 100$$

DEi,j: cantidad de defectos detectados durante la revisión i, correspondientes a la fase j.

DLj: cantidad total de defectos presentes en el producto correspondientes a la fase j.

El comportamiento de estas dos primeras métricas puede ser representado en un único modelo que de una idea global de la eficiencia de las Revisiones y de cada una de las fases consideradas en ellas. Por ejemplo, se pudiera plotear los valores de la efectividad de diferentes revisiones realizadas a un proyecto y compararlas entre sí.

3. Densidad de defectos.

$$DD = \frac{DT}{TP}$$

DT: cantidad total de defectos encontrados en el producto.

TP: tamaño del producto, puede ser estimado en líneas de código (en miles) o en puntos de función.
Da una medida de la proporción de defectos del producto con respecto a su tamaño. De la forma en que está definida debe ser evaluada al finalizar el producto y puede ser aprovechada como experiencia para proyectos futuros, no obstante en las etapas tempranas de desarrollo del proyecto esta métrica puede ser utilizada considerando la estimación preliminar de líneas de código o puntos de función que haya sido considerada en la planificación del proyecto.

4. Porcentaje de reinspección (sólo para inspecciones).

$$PRI = \frac{CR}{CI} * 100$$

CR: cantidad de disposiciones de reinspecciones, que son indicadas por el equipo de inspección si entiende que los defectos son demasiados o muy complejos como para considerar satisfactoria la inspección al producto que se realiza. En estos casos se evalúa de mal el producto inspeccionado y se indica realizar nuevamente una revisión cuando hayan sido corregidos los defectos.

CI: cantidad total de inspecciones.

5. Cantidad total de líneas revisadas.

$$TLCR = \sum_{k=1}^{n} LCR_k$$

LCRk: cantidad de líneas de código revisadas en la revisión k.

n: número total de revisiones.

6. Promedio de líneas revisadas en cada revisión.

$$PLCR = \frac{TLCR}{n}$$

7. Productividad de la revisión promedio.

$$PR = \frac{TLCR}{\sum_{k=1}^{n} TDurac_k}$$

TDurack: tiempo de duración de la revisión k.

8. Razón de preparación promedio de los inspectores.

$$RPP = \frac{TLCR}{\sum_{k=1}^{n} \frac{TPrep_k}{TInsp_k}}$$

TPrepk: tiempo de preparación de la revisión k.

TInspk: total de inspectores que participaron en la revisión k.

La Razón de preparación promedio de los inspectores en una revisión es un indicador de la cantidad de líneas que inspeccionó un inspector del total de líneas inspeccionadas durante la fase de preparación, de las revisiones que se le realizaron al proyecto, y puede emplearse para la planificación de futuras revisiones a proyectos con características similares al analizado. La fase de preparación es la fase de detección de defectos.

9. Eficiencia del inspector en la fase de preparación (EIP).

Esta medida permite comparar lo planificado con lo real en cuanto al tiempo de preparación invertido por los inspectores en la fase de preparación. Además se puede hacer análisis sobre el desempeño de los inspectores y saber qué inspectores son los más eficientes.

Esta métrica también puede ser analizada para los proyectos, de forma que se tenga en cuenta para la planificación de proyectos que se desarrollen con posterioridad, en cuyo caso el modelo reflejaría la relación de horas planificadas y reales para cada proyecto.

10. Esfuerzo promedio por líneas de código fuente.

$$ELC = \frac{\sum_{k=1}^{n} Esf_k}{TLCR}$$

$$Esf_k = TPrep_k + TPart_k * TDurac_k + TRetrab_k$$

Esfk: esfuerzo de la revisión k.

TPartk: total de participantes de la revisión k.

TRetrabk: tiempo de retrabajo o de corrección de defectos de la revisión k.

11. Esfuerzo promedio por defectos detectados.

$$EDE = \frac{\sum_{k=1}^{n} Esf_k}{\sum_{k=1}^{n} DE_k}$$

12. Promedio de defectos detectados por líneas de código fuente.

$$DELC = \frac{\sum_{k=1}^{n} DE_k}{TLCR}$$

Da una idea de la cantidad de defectos encontrados en el proyecto, por lo tanto sirve para analizar la efectividad del proceso de revisión y además permite analizar el mejoramiento continuo de la calidad del proceso de desarrollo, pues los proyectos deben tender a tener cero defectos y a esto deben contribuir las Revisiones. Se puede utilizar un modelo similar al utilizado para mostrar la métrica de densidad de defectos.

# GLOSARIO

**Artefactos:** Una parte de la información que es producida, modificada, o usada por un proceso, define un área de responsabilidad, y está sujeta al control de versión. Un artefacto puede ser un modelo, un elemento del modelo, o un documento. Un documento puede adjuntar otros documentos.

**Cliente:** Una persona u organización, interna o externa a la organización productora, que toma responsabilidad financiera por el sistema. El cliente es el último destinatario del producto desarrollado y sus artefactos. Beneficiado con el producto o servicio.

**Defecto:** Cualquier requerimiento, elemento de diseño o de implementación que si no es cambiado, causará un diseño, implementación, prueba, uso, o mantenimiento inapropiado del producto.

**Error:** Una acción humana que puede producir resultados incorrectos, como la introducción de uno o varios defectos.

**Funcionalidad:** (Functionality, ISO 9126): grado en que las necesidades asumidas o descritas se satisfacen. Se divide en las subcaracterísticas idoneidad, precisión, interoperabilidad, seguridad.

**Equipo:** Equipo de software. Dos personas o más que trabajan para lograr una meta, objetivo o misión común, donde cada individuo tiene asignado un rol específico y donde el completamiento de la misión depende de los miembros del equipo.

**ISO:** (International Organization for Standardization) Es la Organización Internacional para la Normalización; responsable para la normalización a escala mundial. ISO está formado por distintos comités técnicos, cada uno de los cuales es responsable de la normalización para cada área de especialidad. El propósito de ISO es promover el desarrollo de la normalización para fomentar a nivel internacional el intercambio de bienes y servicios y para el desarrollo de la cooperación en actividades económicas, intelectuales, científicas y tecnológicas. El resultado del trabajo técnico dentro de ISO se publica en forma final como normas internacionales.

**IEEE:** (Institute of Electrical and Electronics Engineers) Asociación técnico-profesional mundial dedicada a la estandarización, entre otras cosas. Es la mayor asociación internacional sin fines de lucro formada por profesionales de las nuevas tecnologías, como ingenieros eléctricos, ingenieros en electrónica, científicos de la computación e ingenieros en telecomunicación. Se creó en 1884.

**Madurez:** (*Maturity,* ISO 9126) Subcaracterística de fiabilidad, que indica la frecuencia con que ocurren los fallos.

**Mantenimiento:** (*Maintainability,* ISO 9126) Esfuerzo requerido para implementar cambios. Se divide en las subcaracterísticas capacidad de ser analizado, cambiabilidad, estabilidad, facilidad de prueba.

**Métrica:** Medida o medición. La continúa aplicación de técnicas basadas en la medición al proceso de desarrollo de software y a sus productos para proveer información administrativa significativa y oportuna, junto con el uso de esas técnicas para mejorar el proceso y sus productos.

**Operabilidad:** (*Operability*, ISO 9126) Subcaracterística de facilidad de uso, que indica las características del software que influyen en el esfuerzo del usuario para operar y control operacional.

**Proceso:** (ISO-15504) Proceso de Software, es el proceso (o procesos), usado por una organización (o proyecto), para planificar, administrar, ejecutar, monitorear, controlar y mejorar sus actividades, relacionadas con el software.

**Producto de Software:** (IEEE-12207) Es el conjunto de programas de computadora, procedimientos, documentación y datos, asociados.

**Revisión:** Reuniones de un grupo definido de personas cuyo objetivo es encontrar errores en un artefacto de software. Con revisiones para testear requisitos, diseño, planes, manuales y software Participantes de los revisiones son: los autores que han escrito el artefacto; los revisores que tienen que detectar errores;

el secretario que documenta los errores encontrados; el presentador que expone/explica el artefacto bajo testeo; el líder que dirige la reunión, elige la fecha para la reunión e invita a los participantes.

**Rol**: Papel, cometido o función que tiene o desempeña un actor.

**RUP:** (Racional Unified Process) El Proceso Unificado de Desarrollo Software o simplemente Proceso Unificado es un marco de desarrollo software iterativo e incremental. El refinamiento más conocido y documentado del Proceso Unificado es el Proceso Unificado de Rational o simplemente RUP.

**Seguridad:** (Security, ISO 9126) Subcaracterística de funcionalidad, que indica el grado en que un acceso no autorizado (accidental o deliberado) se prevenga y se permita un acceso autorizado.

**SEI:** (Software Engineering Institute) Es un instituto federal estadounidense de investigación y desarrollo, fundado por el Congreso Estadounidense en 1984 para desarrollar modelos de evaluación y mejora en el desarrollo de software, Es un referente en Ingeniería de Software por realizar el desarrollo del modelo SW-CMM (1991) que ha sido el punto de arranque de todos los que han ido formando parte del modelo que ha desarrollado sobre el concepto de capacidad y madurez, hasta el actual CMMI.

**UML:** (Unified Modeling Language) Es un lenguaje gráfico para visualizar, especificar y documentar cada una de las partes que comprende el desarrollo de software. Entrega una forma de modelar cosas conceptuales como lo son procesos de negocio y funciones de sistema, además de cosas concretas como lo son escribir clases en un lenguaje determinado, esquemas de base de datos y componentes de software reusables.