**UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS**
**Facultad #7**

**TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE INGENIERO EN INFORMÁTICA**

# **TÍTULO:** Propuesta de un procedimiento de pruebas de software en el área temática Sistema de Apoyo a la Salud

**AUTORAS:** Elsy Expósito González
Karina Mileisis Torres Quiñones

**TUTORA:** Ing. Katia Hurtado Duvergel

**ASESORA**: Susana Mas Fernandez

**Ciudad de La Habana, Junio 2008**
**Año 50 de la Revolución**

**DECLARACIÓN DE AUTORÍA**

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmamos la presente a los 25 días del mes de Junio del año 2008.

Elsy Expósito González

_____

Firma del autor

Karina Mileisis Torres Quiñones

_____

Firma del autor

Katia Hurtado Duvergel

_____

Firma del Tutor

## Pensamiento

***“La calidad no cuesta. No es un regalo pero es gratuita. Lo que cuesta dinero son las cosas que no tienen calidad, todas las acciones que resultan de no hacer bien las cosas a la primera vez”.*** ***Philip Crosby***

**Datos de Contacto**
Ing. Katia Hurtado Duvergel khurtado@uci.cu

Profesora Graduada de Ingeniería Informática en el año 2006. Con categoría docente Instructor Recién Graduado. Ha impartido las asignaturas de Seminario de Tesis y Metodología de la Investigación Científica. Actualmente se desempeña como Analista principal del proyecto Balance Material del área temática Sistemas de Apoyo a la Salud.

# Agradecimientos

*De Elsy*

- ✓ *Agradezco a mis padres y a mi hermanita por el apoyo insustituible en cada momento y mantenerme motivada para terminar el camino, por quererme mucho.*
- ✓ *A mi tutora Katia Hurtado Duvergel por su paciencia y ayuda en la realización del trabajo.*
- ✓ *A Rosa Bernaza por haber sido tan incondicional y por todo el tiempo dedicado.*
- ✓ *A Yaikiel por su ayuda, en momentos difíciles cuando pensaba que todo estaba perdido.*
- ✓ *A mi amigo Geyser por haberme dado la luz y la dirección correcta para el trabajo.*
- ✓ *A mis amigos que fueron un factor importante en la realización del trabajo,por todo su cariño.*
- ✓ *A Karina mi compañera de tesis, por darme tanto apoyo y ayuda en el desarrollo de la misma.*
- ✓ *Agradezco a todas las personas que de una forma u otra colaboraron con el desarrollo de la tesis.*
- ✓ *Muchas gracias por la ayuda y cooperación de todos.*

*De Karina Mileisis*

- ✓ *Agradezco a mis padres, por su amor, preocupación y apoyo en todo momento.*
- ✓ *A mis hermanos, por sus cariños y consejos en todo momento.*
- ✓ *A mi amor Pastor, por toda la preocupación y dedicación que mantuvo en la realización de este trabajo.*
- ✓ *A mi familia, por su apoyo en todos estos años.*
- ✓ *A mis amigas por su apoyo en todo momento.*
- ✓ *A mi compañera de tesis por su preocupación y dedicación en todo momento en la realización de este trabajo.*
- ✓ *A mi tutora Katia Hurtado por su ayuda ofrecida.*
- ✓ *A Rosa por todo el apoyo y guía brindada.*
- ✓ *A Jacquelyn por sus buenas sugerencias.*
- ✓ *A todos y cada uno.*

*Muchas Gracias*.

# Dedicatoria

*Existen personas muy especiales para mí que posibilitaron de una forma u otra la realización de este trabajo, es por esto que les dedico este trabajo a:*

- ✓ *Quiero dedicar esta tesis a mí por tantas horas de trabajo.*
- ✓ *A mis padres porque sin ellos no seria lo que soy, por guiarme siempre, por llenarme de su inmenso cariño y sensibilidad, por enseñarme a crecerme ante las dificultades y por quererme.*
- ✓ *A mi hermanita linda que siempre me dio su apoyo, eres muy especial, te quiero mucho Eve.*
- ✓ *A toda mi familia, por su cariño y dedicación que siempre me han dado, por sus consejos, mis abuelitas que siempre las recuerdo y las quiero mucho, mis tíos y tías, mis primas.*
- ✓ *A mis amigas Nana, Evelyn, Daylis, Amalita por siempre estar pendiente de mi y apoyarme, a mi amigo del alma Osmany por ser mi un incondicional en todo momento.*
- ✓ *A mis amigas del barrio por compartir conmigo todo tipo de momentos, Lisbeth, Tica y Gelenys.*
- ✓ *A Yaikiel por haber estado a mi lado en momentos tan difíciles y guiarme y darme fuerzas para terminar la tesis, por soportar todas mis pesadeces y ser paciente conmigo.*
- ✓ *A todos mis compañeras de aula, las muchachitas del apto que me han ayudado mucho y mis amigos, que me hicieron pasar momentos muy agradables, las extrañaré a todas…..*
- ✓ *A mi primo Freddy al hijo y al padre, a Maria, a Tony, a Martica por sus buenos consejos, todas estas personas que contribuyeron a mi armonía en estos 5 años de estudio.*
- ✓ *A mi tía Marcia por dedicarme tiempo y a la memoria de mi tío Jorge que siempre me quiso mucho y hubiese estado muy orgulloso, tío estarás siempre en mi corazón.*
- ✓ *A todos los que se interesaron por mis estudios alguna vez.*

*Elsy*

*Desde niña creía que llegar a este momento era un sueño imposible pero gracias a disimiles personas he podido llegar hasta aquí, a las cuales quiero dedicar este trabajo.*

- ✓ *Ante todo a Dios porque me ha ayudado siempre que se lo he pedido y me ha dado las fuerzas necesarias para seguir adelante en los momentos difíciles.*
- ✓ *A mis padres por su amor, sacrificio, entrega y no desmayar ante las dificultades para contribuir en mi educación y poder llegar a convertirme en una profesional.*
- ✓ *A mis hermanos por sus infinitos consejos, su preocupación en todo momento y ser ejemplos para mí a seguir.*
- ✓ *A mis abuelos por su apoyo incondicional, su confianza en mí y su amor infinito.*
- ✓ *A Claudita, por hacerme sonreír siempre.*
- ✓ *A mis tías y tios Marlenys,Margarita,Melba,Adis, Ninito, Juan y Francisco,asi comoa mis primas Marlenita y Yisel por haberme dado ánimos para seguir adelante.*
- ✓ *En especial a mi novio Pastor por todo su apoyo en este período tan difícil para mí, por su dedicación, preocupación, comprensión y amor en todo momento.*
- ✓ *A mis amigas Daima, Ilsa, Licy, Yuya, Yei, Ana, Sule ,Yuri y Reinier por haber alimentado y fortalecida valores en mi, además por haber reído y llorado junto a mí.*
- ✓ *A mis padres de la Habana Ilsa y Elmo por todo el cariño que me han brindado y acogerme como una hija, nunca los olvidare.*
- ✓ *Agradezco a la Revolución y en especial a Fidel por darme la oportunidad de convertirme en una profesional.*
- ✓ *A todos los profesores que dieron su apoyo en mi formación profesional.*
- ✓ *A todos y cada uno. Muchas Gracias.*

*Karina Mileisis*

# Resumen

El presente trabajo tiene como objetivo elaborar un procedimiento de pruebas de software que contribuya a la calidad de los productos que se desarrollan en el área temática Sistema de Apoyo a la Salud.

Este procedimiento está elaborado sobre la base de la metodología del Proceso Unificado de Desarrollo de Software (RUP), el flujo de trabajo de pruebas que propone la misma y el de la Ayuda del Rational 2003. En combinación con las buenas prácticas del Proceso Software Personal y el Proceso Software en Equipo. Además del modelo de pruebas W, la estrategia de pruebas planteada por Roger Pressman en su libro Un Enfoque Práctico, los niveles de pruebas y los tipos de pruebas de software recomendadas en la bibliografía especializada**.**

Este trabajo tiene como beneficio contribuir a la calidad de los productos que se desarrollan en el área temática SAS ya que se definen los pasos necesarios a ejecutar a lo largo del ciclo de vida de los mismos para controlar si están siendo implementados con la calidad requerida.



**Palabras claves:** procedimiento de pruebas de software, proceso de pruebas, modelos de pruebas, flujo de trabajo de pruebas de software, pruebas de software, plan de pruebas.

# Tabla de Contenidos

# Introducción

En el mundo actual la informatización es la base del desarrollo de las tecnologías en todos los sectores. Con el avance de la informática y el aumento de su impacto social, cada vez son más las instituciones u organizaciones que eligen incorporar aplicaciones de gestión de la información, logrando así una mayor dinámica en sus procesos de negocio. En ese sentido, el desarrollo de sistemas informáticos constituye un sector de gran importancia mundial, debido a que se encuentra en el centro de todas las grandes transformaciones de la sociedad actual. Con el progreso de la industria de software se han creado diferentes metodologías de desarrollo, modelos y procesos de software, que propician una mejor organización y control de la calidad de los productos.

Cuba ha identificado desde muy temprano la conveniencia y necesidad de dominar e introducir en la práctica social las Tecnologías de la Información y las Comunicaciones; y lograr una cultura digital como una de las características imprescindibles del hombre nuevo, lo que facilita a la sociedad a acercarse más hacia el objetivo de un desarrollo sostenible. [1]

Una de las estrategias trazadas para lograr tal objetivo es el desarrollo de una industria nacional de software, la cual es una tarea de gran prioridad para el Estado Cubano debido a la alta perspectiva económica que posee, así como para el aseguramiento de un grupo de actividades del país.

La Industria Cubana del Software está llamada a convertirse en una significativa fuente de ingresos para el país, como resultado del correcto aprovechamiento de las ventajas del alto capital humano disponible. La promoción de la misma en el ámbito internacional ha tenido como línea estratégica aprovechar la enorme credibilidad que tiene Cuba en sectores tales como la salud, la educación y el deporte. [2]

Para lograr una repercusión positiva en el incremento de la exportación y una producción sostenida de software de alta calidad en estos sectores, es muy importante propiciar la calidad, la cual es una actividad que ha surgido como consecuencia de la fuerte demanda de sistemas software, en todos los procesos que se desarrollan en la actualidad.

En Cuba para alcanzar el éxito en la industria del software, se ha insertado el tema de la calidad de software en el plan de estudio de algunas universidades y se han creado empresas dedicadas a la revisión de la eficiencia de los productos software que se desarrollan.

Las pruebas de software son una parte fundamental del proceso de aseguramiento de la calidad; aunque realizárselas a un sistema de informático no significa necesariamente que el proceso de desarrollo esté asegurado y tampoco que de manera directa esté mejorando. Pero implementar un proceso de pruebas de software, y más aún, sostenerlo en el tiempo, es un buen inicio para más adelante aumentar el alcance del sistema informático realizado.

La utilización de metodologías o procedimientos estándares para el análisis, diseño, programación y prueba del software, permiten uniformar la filosofía de trabajo, en aras de lograr una mayor confiabilidad, mantenibilidad y facilidad de prueba; elevando la productividad, tanto para la labor de desarrollo como para el control de la calidad del software, la cual se puede medir después de elaborado el producto; pero esto puede resultar muy costoso si se detectan problemas derivados de errores en el diseño. Por ello es preciso tener en cuenta tanto la obtención de la calidad como su control, durante todas las etapas del ciclo de vida del software.

Se logra el éxito en la producción de software si se hace con calidad y se demuestra, esto sólo es posible con un control de pruebas adecuado, para lo que es necesario contar con un procedimiento y/o software especializado que garantice el seguimiento de pruebas y depuraciones.

La Universidad de las Ciencias Informáticas surge como uno de los programas para contribuir en gran medida a la informatización de la sociedad cubana, participando así en el desarrollo de diversos proyectos. Para ello cuenta con 10 facultades donde cada una posee un determinado perfil.

La Facultad 7 tiene asignada diversas tareas encaminadas a informatizar el sector de la salud debido a que la digitalización de los procesos en este sector es primordial pues permite no sólo contar con métodos novedosos, sencillos y eficaces de gestión de información en consultas, hospitales y centros de investigación biomédica, sino también disponer de complejos software que reducen la posibilidad de error en el diagnóstico de las enfermedades, facilitando y agilizando de esta manera el envío y recepción de los procesos médicos. De esta manera el almacenamiento de la información médica se hace más confiable, es decir no corre el riesgo de sufrir cambios ni de extraviarse; como puede suceder cuando este proceso se realiza de forma manual.

Recientemente en la Facultad 7 se han creado varias áreas temáticas. Un área temática es una estructura que agrupa un conjunto de proyectos en los cuales todos coinciden en el tema de desarrollo

e investigación; el objetivo de la misma es lograr una mejor organización. El área temática Sistema de Apoyo a la Salud (SAS), posee como objetivos fundamentales, desarrollar productos software vinculados esencialmente con la salud y dar soporte técnico a los productos software que se implementan en la misma. SAS tiene actualmente 6 proyectos, los cuales son:

1. Sistema para la Planificación y el Balance de Materiales Gastables en el Ministerio de Salud Pública.
2. Sistema de Gestión de Información en el Proceso de Formación de Recursos Humanos en la Salud.
3. Colaboración Médica.
4. Atención Remota a Servidores.
5. Informatización de la Facultad7.
6. Sistemas de Información Estadística Complementario de Salud (SIEC-Salud).

Los proyectos de informatización del área temática SAS vistos anteriormente, no cuentan con un procedimiento de pruebas adecuado que sea capaz de contribuir a mejorar la calidad de los productos software que se desarrollan en el área temática, para que los mismos al ser terminados tengan la calidad mínima requerida. Esto trae consigo que los productos terminados se envíen al laboratorio de Calidad de la facultad, donde se hacen pruebas de Caja Negra, teniendo en cuenta los requisitos y casos de uso propuestos.

En Calidad se detectan errores de entrada, de interfaz y de validación, esto se debe a que las pruebas que se hacen internamente en el área temática son revisiones muy sencillas, empíricas, que no están basadas en el conocimiento. Normalmente el Equipo de Desarrollo de cada proyecto se encuentra presionado por la necesidad de cumplir con las fechas establecidas en el cronograma y el proceso de pruebas no se cumple o se ejecuta de una manera desorganizada, sin métodos y sin considerar los tiempos de pruebas establecidos por RUP en cada fase, ya que esta es la metodología utilizada por todos los proyectos del área temática.

El resultado es un software sin las pruebas mínimas requeridas y sin el nivel de calidad esperado. También existe gran inexperiencia del funcionamiento del flujo de pruebas por parte del personal que elabora en los proyectos, lo que significa que no se realizan pruebas en ninguna de las etapas de desarrollo del producto. No existe un grupo de ingenieros capaces de realizar las pruebas necesarias para que cuando el proyecto llegue al laboratorio de Calidad de la facultad el proceso de pruebas sea más rápido y eficiente.

Como consecuencia de esto se incurre en faltas que implican pérdidas de tiempo y de recursos, tanto materiales como humanos y malas estimaciones de costo, que según Roger Pressman en su libro Un Enfoque Práctico, en algunos casos podría ser hasta 200 veces más costoso; propiciando de esta forma que el software sea regresado hacia sus desarrolladores con el fin de arreglar todos los errores detectados.

Teniendo en cuenta esta sucesión de dificultades que se vienen presentando en los diferentes proyectos del área temática de Sistema de apoyo a la salud se detecta como **problema científico** que el mecanismo que se sigue para realizar las pruebas software en el área temática de Sistemas de Apoyo a la Salud está afectando la calidad de los productos que se implementan en la misma.

A partir del problema científico planteado se enmarca como **objeto de estudio** los procesos de pruebas de software, definiéndose como **campo de acción** el proceso de pruebas de software en el área temática Sistema de Apoyo a la Salud.

Para darle cumplimiento al problema científico planteado se ha definido como **objetivo general**, proponer un procedimiento de pruebas de software en el área temática Sistemas de Apoyo a la Salud.

La **idea a defender** es la siguiente: la elaboración de un procedimiento adecuado para realizar las pruebas software en el área temática Sistemas de Apoyo a la Salud, contribuirá a la calidad de los productos que se implementan en la misma.

Para dar cumplimiento al objetivo general de este trabajo investigativo se trazaron las siguientes **tareas de investigación:**

1- Analizar el estado del arte de las pruebas y su relación con las diferentes metodologías de desarrollo de software.
2- Analizar diversos modelos de pruebas.
3- Seleccionar las metodologías de desarrollo de software, modelos, procesos y pruebas para la propuesta de solución.
4- Analizar el estado de las pruebas en el ámbito Internacional, Nacional y en la Universidad de las Ciencias Informáticas.
5- Analizar los mecanismos relacionados con las pruebas de software que se siguen en los proyectos productivos del área temática Sistemas de Apoyo a la Salud.

6- Presentar un procedimiento de pruebas de software para el área temática Sistemas de Apoyo a la Salud.

El trabajo fue dividido en tres capítulos. A continuación se muestra el nombre de cada uno con una breve explicación de su contenido:

**Capítulo 1: Fundamentación Teórica**

Se realiza una descripción de algunas metodologías de desarrollo de software y de sus flujos de trabajo de pruebas, se exponen características de diversos procesos de software y algunos conceptos definidos por diferentes especialistas en el tema de la calidad de los productos software. Se abordan temas referentes a modelos de pruebas, casos de pruebas, estrategias de pruebas, niveles, métodos y tipos de pruebas, así como las técnicas del Proceso de Software Personal y el Proceso de software en Equipos.

**Capítulo 2: Actualidad de las pruebas**

Se realiza una caracterización sobre el estudio y utilización de las pruebas de software en el ámbito internacional y nacional, en proyectos productivos de la Universidad de las Ciencias Informáticas y el área temática Sistema de Apoyo a la Salud de la Facultad 7. Para esto se emplearon diferentes técnicas como son la entrevista y encuesta.

**Capítulo 3: Descripción de la solución propuesta**

Se propone un procedimiento de pruebas para los productos que se desarrollan en el área temática SAS, en éste se definen las personas involucradas para realizar las pruebas de software, se describen períodos para la organización del proceso de pruebas a lo largo del ciclo de vida de los productos software, así como 4 actividades fundamentales a cumplir para cada prueba y un grupo de artefactos que se involucran en el proceso.

# Capítulo 1: Fundamentación Teórica

## 1.1 Introducción

El presente capítulo brinda una descripción general del estado del arte de las pruebas de software. Se realiza una descripción sobre algunas metodologías de desarrollo de software y específicamente de los flujos de trabajo de pruebas propuestos por algunas de ellas. Se abordan diferentes tipos de modelos de pruebas y procesos de software. También se hace énfasis en qué consiste el diseño de los casos de pruebas, las estrategias de pruebas, los niveles de pruebas, los métodos de pruebas y los tipos de pruebas, teniendo en cuenta las necesidades y las características del entorno donde se aplicará la solución propuesta.

## 1.2 Reseña de algunas metodologías de desarrollo de software

En el mundo de la computación tan cambiante de hoy en día, y sobre todo de gran evolución tecnológica se ha hecho necesario desarrollar metodologías para contribuir a la calidad de los productos software y obtener un mejoramiento continuo de todos los procesos relacionados con el avance de los mismos.

Los mecanismos de evaluación y mejora en el desarrollo de software, han permitido que las empresas implementen la metodología que más se ajuste a sus necesidades y forma de trabajar. Teniendo en cuenta este enfoque, se presenta a continuación como se aborda el tema de las pruebas en las siguientes metodologías.

### 1.2.1 Metodología SCRUM

**Definición**

Scrum es una metodología ágil de gestión de proyectos, básicamente se puede decir que es un modelo de gestión de proyecto**.** [3]

**Objetivos**

- ✓ Elevar al máximo la productividad de un software en equipo.
- ✓ Maximizar la realimentación sobre el desarrollo del software pudiendo corregir problemas y mitigar riesgos de forma temprana. [4]

**Principales características**

- ✓ El desarrollo de software se realiza mediante iteraciones, denominadas sprints, con una duración de 30 días. El resultado de cada sprint es un incremento ejecutable que se muestra al cliente.

- ✓ Las reuniones a lo largo del desarrollo del proyecto son muy importantes, entre ellas se destaca la reunión diaria de 15 minutos del equipo de desarrollo para coordinación e integración.
- ✓ La metodología resulta sencilla, solo necesita de algunos roles y artefactos para su aplicación.
- ✓ El proyecto puede empezar con cualquier actividad y se puede pasar de una actividad a otra en cualquier momento maximizando la flexibilidad y la productividad del equipo.
- ✓ Puede ser aplicado a distintos modelos de calidad puesto que estos te dicen qué tienes que hacer, es decir, te dicen qué tienes que gestionar en el proyecto, pero no te dicen cómo. Ahí es donde entra Scrum como modelo de gestión del proyecto.[5]

**Fases**

**Pre-Juego o Planeamiento:** El propósito es establecer la visión, definir expectativas y asegurar la financiación. Las actividades de esta fase son la escritura de la visión, el presupuesto, el registro de acumulación o retraso (Backlog) del producto inicial y los ítems estimados, así como la arquitectura de alto nivel, el diseño exploratorio y los prototipos.

**Pre-Juego o Montaje:** El propósito es identificar más requerimientos y priorizar las tareas para la primera iteración. Las actividades son planificación, diseño exploratorio y prototipos. Se diseñan cómo los requisitos de la reserva serán puestos en ejecución. Esta fase incluye la modificación de la arquitectura del sistema y el diseño de alto nivel.

**Juego o Desarrollo:** El propósito es implementar un sistema listo para entregar en una serie de iteraciones de treinta días llamadas “corridas” (sprints). Las actividades son un encuentro de planeamiento de corridas en cada iteración, la definición del registro de acumulación de corridas y los estimados, y encuentros diarios de Scrum.

**Pos-Juego o Liberación:** El propósito es el despliegue operacional. Las actividades son: documentación, entrenamiento, mercadeo y venta. [6]

**Trabajadores y Responsabilidades**

**Propietario del producto (PP):** Representa a todos los interesados en el producto final. Es quien determina las prioridades.

**Gestor o Manager del Scrum (SM):** Gestiona y facilita la ejecución del proceso. Entre sus responsabilidades esta la formación y entrenamiento del proceso, así como la incorporación de Scrum en la cultura de la empresa y garantía del cumplimiento de los roles y las responsabilidades.

**Equipo (E):** Es el responsable de transformar el Backlog de la iteración en un incremento de la funcionalidad del software. Construye el producto.

**Interesados (I):** Asesoran y observan. [7]

**Artefactos**

**Pila del producto (Product Backlog):** Es la relación de los requisitos del producto, a los cuales no es necesario realizarle un excesivo detalle pero si deben presentarse en orden de prioridades en una lista en evolución y abierta a todos los roles que incorporan constantemente las necesidades del sistema. El propietario del producto es su responsable y es quien decide sobre su contenido, priorización y disponibilidad. La misma se mantiene durante todo el ciclo de vida (hasta la retirada del sistema) y nunca llega a ser una lista completa y definitiva.

**Pila del período (Sprint Backlog):** Son los requisitos comprometidos por el equipo para realizar un sprint con un nivel de detalle suficiente para su ejecución, a fin de lograr al final del mismo un incremento de la funcionalidad.

**Incremento:** Es parte del producto desarrollado en un sprint, en condiciones de ser usada (pruebas, codificación limpia y documentada). [8]

**Actividades**

**Período (Sprint):** Es el período de tiempo durante el que se desarrolla un incremento de funcionalidad. Constituye el núcleo de Scrum, que divide de esta forma el desarrollo de un proyecto en un conjunto de pequeñas “carreras”.

**Planificación del sprint:** Es una jornada de trabajo donde el propietario del producto explica las prioridades y dudas al equipo. El equipo estima el esfuerzo de los requisitos prioritarios y se elabora la pila del sprint, luego el Gestor o Manager del Scrum define en una frase el objetivo del sprint.

**Revisión del sprint:** Es una reunión Informativa de aproximadamente 4 horas dirigida por el Gestor. En esta reunión se presenta el incremento de las funcionalidades implementadas. El propietario del producto trata con los asistentes y con el equipo las posibles modificaciones en el Backlog y se anuncia el próximo sprint.

**Revisión diaria:** La revisión diaria tiene una duración de 15 minutos y es dirigida por el Gestor de Scrum, el cual pregunta a todos los asistentes ¿Qué hiciste ayer? ¿Cuál es el trabajo para hoy? y ¿Qué necesitas? Al terminar la revisión se actualiza la pila del sprint. [9]

### 1.2.2 Metodología Programación Extrema

**Definición**

La Programación Extrema es una metodología ágil de desarrollo de software que se enfoca en la simplicidad, la comunicación y la realimentación o reutilización del código desarrollado. Se basa en poner junto a todo el equipo y emplear prácticas sencillas, con suficiente realimentación para facilitar que el equipo vea donde está y ajuste las prácticas a su situación concreta. [10]

**Objetivos**

- ✓ Satisfacer al cliente proporcionándole al mismo el software que él necesita y cuando lo necesite, por tanto responder muy rápido a las necesidades del cliente es una de las principales características de esta metodología, incluso cuando los cambios sean al final de ciclo de la programación.
- ✓ Potenciar al máximo el trabajo en grupo, por tanto los jefes de proyecto, los clientes y desarrolladores son parte del equipo y están involucrados en el desarrollo del software. [11]

**Principales caracterisiticas**

La metodología se basa en:

- ✓ **Pruebas:** Se basa en las pruebas realizadas a los principales procesos, de tal manera que adelantándose en algo hacia el futuro, se pueda hacer pruebas de las fallas que pudieran ocurrir. Es como si se adelantaran a obtener los posibles errores.
- ✓ **Recodificación**: Se basa en la reutilización de código, para lo cual se crean patrones o modelos estándares, siendo más flexible al cambio.
- ✓ **Programación por parejas**: Todo el código de producción lo escriben dos personas frente al ordenador, con un sólo ratón y un sólo teclado. Cada miembro de la pareja juega su papel: uno codifica en el ordenador y piensa la mejor manera de hacerlo, el otro piensa más estratégicamente.

El emparejamiento es dinámico, cada miembro lleva a cabo la acción que el otro no está haciendo en ese momento. [12]

**Fases**

**Exploración**: En esta fase, los clientes plantean a grandes rasgos las historias de usuario que son de interés para la primera entrega del producto. Se prueba la tecnología y se exploran las posibilidades de la arquitectura del sistema construyendo un prototipo.

**Planificación de la entrega:** En esta fase el cliente establece la prioridad de cada historia de usuario, y correspondientemente, los programadores realizan una estimación del esfuerzo necesario de cada una de ellas. Se toman acuerdos sobre el contenido de la primera entrega y se determina un cronograma en conjunto con el cliente.

**Iteraciones:** Esta fase incluye varias iteraciones sobre el sistema antes de ser entregado. En la primera iteración se puede intentar establecer una arquitectura del sistema que pueda ser utilizada durante el resto del proyecto. Al final de la última iteración el sistema estará listo para entrar en producción.

Los elementos que deben tomarse en cuenta durante la elaboración del Plan de la Iteración son: historias de usuario no abordadas, velocidad del proyecto, pruebas de aceptación no superadas y tareas no terminadas en la iteración anterior. Todo el trabajo de la iteración es expresado en tareas de programación, cada una de ellas es asignada a un programador como responsable, pero llevadas a cabo por parejas de programadores.

**Producción:** La fase de producción requiere de pruebas adicionales y revisiones de rendimiento antes de que el sistema sea trasladado al entorno del cliente. Al mismo tiempo, se deben tomar decisiones sobre la inclusión de nuevas características a la versión actual, debido a cambios durante esta fase.

**Mantenimiento:** Mientras la primera versión se encuentra en producción, el proyecto XP debe mantener el sistema en funcionamiento al mismo tiempo que desarrolla nuevas iteraciones. Para realizar esto se requiere de tareas de soporte para el cliente. De esta forma, la velocidad de desarrollo puede bajar después de la puesta del sistema en producción. La fase de mantenimiento puede requerir nuevo personal dentro del equipo y cambios en su estructura.

**Muerte del Proyecto**: Es cuando el cliente no tiene más historias para ser incluidas en el sistema. Esto requiere que se satisfagan las necesidades del cliente en otros aspectos como rendimiento y confiabilidad del sistema. Se genera la documentación final del sistema y no se realizan más cambios en la arquitectura. La muerte del proyecto también ocurre cuando el sistema no genera los beneficios esperados por el cliente o cuando no hay presupuesto para mantenerlo. [13]

**Trabajadores y Responsabilidades**

Aunque en otras fuentes de información aparecen algunas variaciones y extensiones de roles XP, a continuación se describe los roles de acuerdo con la propuesta original de Kent Beck, el padre de XP.

**Programador:** El programador escribe las pruebas unitarias y produce el código del sistema. Debe existir una comunicación y coordinación adecuada entre los programadores y otros miembros del equipo.

**Cliente:** El cliente escribe las historias de usuario asignándole la prioridad a las mismas y decide cuáles se implementan en cada iteración centrándose en aportar mayor valor al negocio. El cliente es sólo uno más dentro del proyecto.

**Encargado de pruebas:** El encargado de las pruebas redacta junto al cliente las pruebas funcionales. Ejecuta las pruebas regularmente, difunde los resultados en el equipo y es responsable de las herramientas de soporte para las pruebas.

**Encargado de seguimiento:** El encargado de seguimiento proporciona realimentación al equipo en el proceso XP. Su responsabilidad es verificar el grado de acierto entre las estimaciones realizadas y el tiempo real dedicado, comunicando los resultados para mejorar futuras estimaciones. También realiza el seguimiento del progreso de cada iteración y evalúa si los objetivos son alcanzables con las restricciones de tiempo y recursos presentes. Determina cuándo es necesario realizar algún cambio para lograr los objetivos de cada iteración. [14]

### 1.2.3 Metodología RUP

**Definición**

RUP es una metodología de desarrollo de software robusta. La misma por sus características, tiene una forma disciplinada de asignar tareas y responsabilidades en una empresa de desarrollo de software. [15]

**Objetivo**
Asegurar la producción de software de calidad dentro de plazos y presupuestos predecibles. [16]

**Principales caracteristicas**
- ✓ Dirigido por casos de uso.
- ✓ Centrado en la arquitectura.
- ✓ Iterativo (mini-proyectos) e incremental (versiones). [17]

**Fases**

**Inicio:** Esta fase tiene como propósito establecer el alcance del proyecto y proponer una visión general de la arquitectura de software. Se identifican todas las entidades externas con las que se trata (actores) y se define la interacción a un alto nivel de abstracción. Se identifican los casos de uso que orientarán la funcionalidad. Se diseñan las arquitecturas candidatas.

**Elaboración:** Tiene como objetivos realizar el análisis del dominio del problema y definir el plan del proyecto donde se planifiquen las actividades necesarias y recursos requeridos. RUP presupone que la fase de elaboración brinda una arquitectura suficientemente sólida junto con requerimientos y planes bastante estables. Se describen en detalle la infraestructura y el ambiente de desarrollo, así como el soporte de herramientas de automatización. Al final de esta fase, debe estar identificada la mayoría de los casos de uso y los actores, debe quedar descripta la arquitectura de software y se debe crear un prototipo de ella.

**Construcción:** El propósito de esta fase es completar la funcionalidad del sistema para ello se deben refinar los requerimientos pendientes, administrar el cambio de los artefactos construidos, ejecutar el plan de administración de recursos y tener en cuenta las mejoras en el proceso de desarrollo para el proyecto. En esta fase todos los componentes restantes se desarrollan y se incorporan al producto. Se debe compilar también una versión de entrega. Es la fase más prolongada de todas.

**Transición:** Comienza cuando el producto está suficientemente maduro para ser entregado. Se corrigen los últimos errores y se agregan los rasgos pospuestos. La fase consiste en prueba beta, piloto, entrenamiento a usuarios y despacho del producto a mercadeo, distribución y ventas. Se produce también la documentación. Se llama transición porque se transfiere a las manos del usuario, pasando del entorno de desarrollo al de producción. [18]

**Comportamiento de las pruebas en cada flujo de trabajo**

En RUP el proceso de prueba tiene una particular organización y relación con las restantes disciplinas.


**Figura 1.1. La prueba de software en la metodología RUP.**

- ✓ La disciplina **Requerimientos**, captura los requisitos para el producto de software, el cual es una de las entradas principales para identificar qué pruebas se van a ejecutar y es la base de una ejecución de pruebas debido a la captura clara y eficiente de estos.

- ✓ La disciplina **Análisis y Diseño**, determina el diseño apropiado para el software, la cual es otra de las entradas principales para identificar qué pruebas se van a ejecutar, con el objetivo de establecer el diseño apropiado del software.

- ✓ La disciplina de **Implementación** produce versiones operacionales del sistema (builds) que son validados por la prueba. Dentro de una iteración, múltiples builds serán probados.

- ✓ La disciplina **Despliegue** entrega el producto de software completo al usuario final. Antes el software es validado en el proceso de prueba, aunque las pruebas de aceptación y las pruebas betas son ejecutadas como parte del despliegue.

- ✓ La **Gestión de proyecto** planifica el proyecto y las iteraciones, el de Plan de Iteración es un artefacto que es una importante entrada usada cuando se va a definir la misión de evaluación para la prueba.

- ✓ La disciplina **Gestión de la Configuración y Cambios** controla los cambios dentro del proyecto. La prueba verifica que cada cambio ha sido completado apropiadamente. [19]

**Trabajadores, Artefactos y actividades**
La metodología de Rup define por cada flujo de trabajo trabajadores, artefactos y actividades.

- ✓ **Trabajadores:** Roles que definen el comportamiento y responsabilidades de los individuos.
- ✓ **Artefactos:** Son los elementos de entrada y salida de las actividades. Son productos tangibles del proyecto. Las cosas que el proyecto produce o usa para componer el producto final (modelos, documentos, código, ejecutables).
- ✓ **Actividades**: Son tareas que tiene un propósito claro y se asignan a un rol. Es realizada por un trabajador y manipula elementos. [20]

### 1.2.4 Análisis valorativo de las metodologías de desarrollo de software

Después de estudiar las diferentes metodologías, se expone que las técnicas de Programación Extrema proporcionan un camino para obtener productos de calidad medible en el desarrollo de proyectos de software, intenta reducir la complejidad del software, además realiza un seguimiento de las pruebas y el hecho de que sea una metodología ligera, la hace especialmente idónea para un entorno heterogéneo con un grupo grande de desarrolladores. A pesar de todas estas ventajas que proporciona esta metodología no es la más recomendable a usar porque las pruebas son realizadas después que el código está escrito, lo cual provoca una disminución del alcance de las pruebas.

La metodología Scrum es una forma de gestionar proyectos de software y además no es una metodología de análisis, ni de diseño, como podría ser RUP, es una metodología de gestión del trabajo. Esta metodología no es la más recomendable a utilizar en el procedimiento de pruebas a definir, porque como método Scrum enfatiza valores y prácticas de gestión, sin pronunciarse sobre requerimientos, implementación y demás cuestiones técnicas, esto ayuda a la organización del proyecto que se desarrolla; pero se queda insuficiente para guiar las pruebas, además no se hace énfasis en los requerimientos, esto provoca que no se puedan medir las pruebas desde un inicio y solo se proyecta sobre el control de la administración.

Por lo tanto, se plantea que el procedimiento de pruebas que se propone esté guiado por la metodología RUP, porque es la metodología que se sigue en el desarrollo de los proyectos del área temática SAS, también es preciso recalcar que RUP es una de las más completa y organizada,

presenta además entre uno de sus flujos de trabajo uno específicamente dedicado a la realización de las pruebas de software, entre las prácticas comunes que utiliza está una relacionada con la prueba de calidad del software.

## 1.3 Flujos de trabajo de pruebas de software

### 1.3.1 Flujo de trabajo de pruebas de RUP

En el flujo de trabajo de pruebas propuesto por RUP se verifica el resultado de la implementación probando cada construcción, incluyendo tanto construcciones internas como intermedias, así como las versiones finales del sistema a ser entregadas a terceros.

**Roles y responsabilidades**

**Diseñador de pruebas:** Los diseñadores de pruebas no llevan a cabo las pruebas, sino que se dedican a la preparación y evaluación de las mismas. Son responsables de:

- ✓ La integridad del modelo de pruebas, asegurando que el modelo cumpla con su propósito.
- ✓ Debe planear, seleccionar y describir las pruebas, los casos de prueba y los procedimientos de prueba correspondientes que se necesitan.
- ✓ Evaluar las pruebas de integración y de sistema cuando estas se ejecutan.

**Ingeniero de componentes**

- ✓ Son responsables de los componentes de prueba que automatizan algunos de los procedimientos de prueba (no todos los sistemas de pruebas pueden ser automatizados).

**Ingeniero de pruebas de integración**

- ✓ Realizan las pruebas de integración que se necesitan para cada construcción producida en el flujo de trabajo de la implementación.
- ✓ Prueba el resultado, es decir una construcción, creado por el integrador de sistemas de flujo de trabajo de la implementación y realiza la documentación de los defectos como resultado de las pruebas de integración.

**Ingeniero de pruebas de sistema**

- ✓ Realiza las pruebas de sistema necesarias sobre una construcción que muestra el resultado (ejecutable) de una iteración completa.
- ✓ Documenta los defectos en los resultados de las pruebas del sistema.
- ✓ Debe tener familiaridad con el comportamiento observable externo e interno del sistema.

**Artefactos**

**Modelo de pruebas**

- ✓ El modelo de pruebas es una colección de casos de pruebas, procedimientos de prueba y componentes de prueba.
- ✓ Describe principalmente como se prueban los componentes ejecutables en el modelo de la implementación con pruebas de integración y de sistema.
- ✓ Describe como pueden ser probados aspectos específicos del sistema. Ejemplo: si la interfaz del usuario es utilizable y consistente o si el manual de usuario del sistema cumple con su cometido.

**Caso de prueba**

Un caso de prueba especifica una forma de probar el sistema, incluyendo la entrada o resultado con la que se ha de probar y las condiciones bajo las que ha de probarse.

**Procedimiento de prueba**

- ✓ Un procedimiento de prueba especifica cómo realizar uno o varios casos de prueba o partes de estos.

**Componente de prueba**

- ✓ Un componente prueba automatiza uno o varios procedimientos de prueba o partes de ellos.
- ✓ Pueden ser desarrollados utilizando un lenguaje de guiones o un lenguaje de programación, o pueden ser grabados con una herramienta de automatización de pruebas.
- ✓ Se utilizan para probar los componentes en el modelo de implementación proporcionando entradas de prueba, controlando y monitorizando la ejecución de los componentes a probar y, posiblemente, informando de los resultados de pruebas.
- ✓ Estos pueden ser implementados utilizando tecnología de objetos.

**Plan de prueba**

- ✓ El plan de prueba describe las estrategias, recursos y planificación de la prueba.
- ✓ La estrategia de prueba incluye la definición del tipo de pruebas a realizar para cada iteración y sus objetivos, el nivel de cobertura de prueba y de código necesario y el porcentaje de pruebas que deberían ejecutarse con un resultado específico.

**Defecto**

- ✓ Un defecto es una anomalía del sistema, como por ejemplo un síntoma de un fallo software o un problema descubierto en una revisión.

- ✓ Puede ser utilizado para localizar cualquier cosa que los desarrolladores necesitan registrar como síntoma de un problema en el sistema que ellos necesiten controlar.

**Evaluación de prueba**

- ✓ Una evaluación de prueba es una evaluación de los resultados de los esfuerzos de prueba, tales como la cobertura del caso de prueba, la cobertura de código y el estado de los defectos.

**Actividades**

**Planificar prueba:** El propósito de la prueba es planificar los esfuerzos de prueba en una iteración llevando a cabo las siguientes tareas:

- ✓ describir una estrategia de prueba.
- ✓ estimar los requisitos para el esfuerzo de la prueba .Ejemplos: los recursos humanos y sistemas necesarios.
- ✓ planificar el esfuerzo de la prueba.

**Diseñar pruebas:** El propósito de diseñar las pruebas es:

- ✓ Identificar y describir los casos de prueba para cada construcción.
- ✓ Identificar y estructurar los procedimientos de prueba especificando cómo realizar los casos de prueba.
- ✓ Los diseñadores de pruebas desarrollan una estrategia de prueba para la iteración, es decir, deciden qué tipo de pruebas ejecutar, cómo ejecutar dichas pruebas, cuándo ejecutarlas y cómo determinar si el esfuerzo de prueba tiene éxito. [21]

### 1.3.2 Flujo de trabajo de pruebas de la Ayuda del Rational 2003

La Ayuda del Rational 2003 es una de las versiones más actuales de la metodología de RUP, la cual cuenta con un flujo de trabajo de pruebas que incorpora nuevos elementos en comparación con el flujo de trabajo de pruebas expuesto por RUP**.**

**Roles y Responsabilidades**

**Administrador de pruebas**

- ✓ Responsable del éxito de las pruebas y de la calidad de las mismas.
- ✓ Planifica y administra los recursos.
- ✓ Resuelve los problemas que impidan realizar las pruebas.
- ✓ Contribuye en el plan de pruebas

- ✓ Elaborar o valorar el resumen de las pruebas cuando se concluya este flujo de trabajo.

**Analista de prueba**

- ✓ Identifica y define las pruebas requeridas.
- ✓ Monitorea el progreso de la prueba y el resultado en cada ciclo de prueba, evaluando la calidad total experimentada como un resultado de las actividades de prueba.
- ✓ Representa apropiadamente las necesidades de los stakeholder que no tienen representación regular y directa en el proyecto.

**Diseñador de pruebas**

- ✓ Define el método de prueba y asegura su implementación exitosa.
- ✓ Identifica las técnicas apropiadas, herramientas e instrucciones para implementar estrategias de pruebas necesarias y encauza los recursos correspondientes para las pruebas.

**Probador**

- ✓ Es el responsable durante las actividades principales de las pruebas, el cual incluye la conducción de las pruebas necesarias y el registro del resultado de la prueba.

**Principales artefactos**

**Plan de Pruebas**

- ✓ El principal objetivo de este artefacto es planificar el proceso de la pruebas, determinar el objetivo fundamental de la misma e identificar los recursos requeridos.
- ✓ Definir los parámetros necesarios para realizar las pruebas propuestas.
- ✓ Explicar los pasos para la utilización de la estrategia de prueba definida.

**Estrategia de prueba**

- ✓ Tiene como propósito convencer a las personas involucradas de la importancia de realizar las pruebas.
- ✓ Define el plan estratégico para la realización del esfuerzo de la prueba.
- ✓ Define los métodos, las técnicas y los tipos de pruebas necesarios para ejecutar el proceso.

**Arquitectura de automatización de la prueba**

- ✓ Proporciona una descripción arquitectónica comprensible de las pruebas que se realizan al sistema, se incluyen además vistas arquitectónicas para representar los aspectos del sistema.
- ✓ Define las características principales del sistema.
- ✓ Permite una mejor comprensión del sistema, para que las pruebas tengan un resultado satisfactorio.
- ✓ Describe aspectos claves como: capacidad de mantenimiento, extensibilidad, fiabilidad, coincidencia, distribución, seguridad (valor) y recuperación.

**Configuración del entorno de la prueba**

- ✓ Especifica la configuración de hardware, software, y los ajustes de ambiente requeridos, esto influye en el cumplimiento del objetivo de la prueba.
- ✓ Especifica las condiciones necesarias para realizar las actividades de pruebas requeridas.
- ✓ Permite controlar y tener organizada todas las actividades de configuración, y ayuda a asegurar que los resultados de la prueba sean exactos y válidos.
- ✓ Proporciona una descripción de todas las computadoras necesarias, software y hardware de periférico de Entrada/Salida.

**Resumen de evaluación de las pruebas**

- ✓ El Resumen de evaluación de las pruebas organiza y presenta un análisis de los resultados de la prueba y las medidas claves para la revisión y evaluación de las mismas.
- ✓ Contiene una declaración y evaluación general de las pruebas realizadas, que proporciona recomendaciones para futuras prueba.
- ✓ Recoge, organiza, y presenta los resultados de las pruebas efectuadas y los puntos claves para evaluar la calidad de las mismas.

**Resultado de la prueba**

- ✓ Tiene como propósito hacer un sumario con el análisis de los resultados de las pruebas efectuadas y una evaluación detallada de los objetivos de la misma.
- ✓ Registra las conclusiones de las pruebas y los detalles más importantes.
- ✓ Este documento puede variar según la tecnología e instrumentos usados durante la ejecución de las pruebas.

**Actividades**

- ✓ **Definir la misión de la evaluación:** Su función fundamental es identificar el método apropiado de la prueba para la iteración y estar de acuerdo con los stakeholder de las metas correspondientes que dirigirán las pruebas.

  A partir del Plan de iteración y el Plan de Desarrollo de Software se define el Plan de Prueba, se identifica los objetivos y la estrategia de las pruebas. Se define además como monitorear y evaluar el progreso de las pruebas.

- ✓ **Verificar el enfoque de la prueba:** Su función fundamental es demostrar que diferentes técnicas facilitarían la prueba planificada, es verificar demostrando que el enfoque trabajara, producirá resultados precisos y es apropiado para los recursos disponibles.

- ✓ **Verificar la estabilidad del build:** Su función fundamental es validar que el build es suficientemente estable para la prueba detallada y para comenzar la evaluación. Es importante definir los detalles de la prueba, implementarla, ejecutarla, analizar los fallos y determinar los resultados de la prueba.

- ✓ **Probar y evaluar:** Su función fundamental es realizar pruebas adecuadas a lo ancho y en profundidad para permitir una evaluación suficiente de los elementos que son objetivos de las pruebas. Esta evaluación suficiente es dirigida por la misión de evaluación actual y los motivadores de prueba.

- ✓ **Lograr la misión aceptable:** Su función fundamental es entregar un resultado útil de la evaluación de las pruebas para los stakeholder, este resultado está determinado en términos de la misión de la evaluación. En la mayoría de los casos significarían enfocar el esfuerzo en ayudar al equipo de proyecto en lograr los objetivos del Plan de Iteración que se aplica al ciclo de prueba actual.

- ✓ **Mejorar la calidad de las pruebas**: Su función fundamental es mantener y mejorar la calidad de las pruebas. Esto es especialmente importante si la intención es reutilizar las ventajas desarrolladas en el actual ciclo de prueba en ciclos de pruebas posteriores. [22]

### 1.3.3 Análisis valorativo de los flujos de trabajo de pruebas definidos por RUP y la Ayuda del Rational 2003

Luego de haber realizado un estudio de los diferentes trabajadores, artefactos y actividades propuestas en cada uno de los flujos de trabajos de pruebas analizados, se plantea que para el procedimiento de pruebas que se propone se tomen de forma híbrida los roles, artefactos y actividades a utilizar para obtener una óptima propuesta de solución.

## 1.4 Modelos de prueba

A diferencia de los modelos clásicos que describen como se aplican las pruebas de software en un momento concreto (ver figura 1), existen modelos de pruebas que proponen como realizar las mismas a lo largo de todo el ciclo de vida de un software (ver figura 2), tales como el Modelo de pruebas V y Modelo de pruebas W.

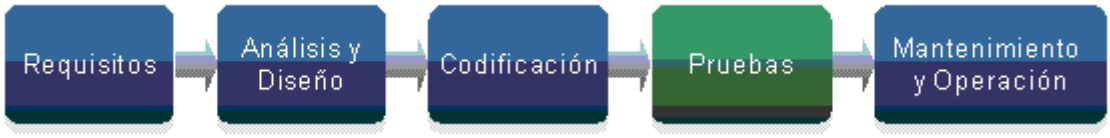



**Figura 1.2. Modelo de pruebas clásico**

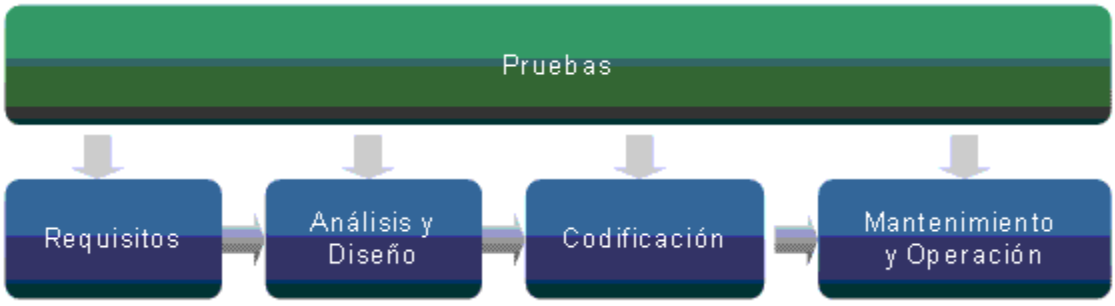



**Figura 1.3. Modelo de pruebas a lo largo del ciclo de vida**

### 1.4.1 Proceso de pruebas basado en el Modelo de Pruebas V

El Modelo de pruebas en V define desde etapas tempranas que se vayan diseñando a través de los planes de pruebas las pruebas que más tarde en el ciclo de vida serán ejecutadas.

Este Modelo de pruebas es extensible a modelos de desarrollo iterativos si se considera la V como una Iteración del proyecto completo.

A pesar de las ventajas que nos proporciona este modelo, en su procedimiento no contempla la posibilidad de retornar a etapas inmediatamente anteriores, cosa que en la realidad puede ocurrir y se toma toda la complejidad del problema de una vez y no en iteraciones o ciclos de desarrollo, lo que disminuye el riesgo.


**Figura 1.4. Modelo de Pruebas en V**

En el gráfico del modelo en V, las labores de pruebas aparecen en los bloques verdes y en los bloques azules se representan las actividades del ciclo de vida del software antes de su puesta en producción.

### 1.4.2 Proceso de pruebas basado en el Modelo de pruebas W

El modelo en W surge como refinamiento del modelo en V. Refleja mejor la interdependencia que existe entre el Equipo de Desarrollo y el Equipo de Pruebas a lo largo de todo el proceso de desarrollo del sistema destacando los siguientes puntos:

- ✓ En las primeras etapas se consideran labores de pruebas que no aparecen reflejadas en el modelo original como son: la revisión de los requisitos, la revisión de la especificación del sistema, la revisión de la arquitectura, la revisión del diseño detallado y las revisiones de código.

- ✓ En las etapas finales también se distingue del modelo original en V porque se desglosan las tareas de pruebas propiamente dichas y las labores de depuración y corrección de los errores detectados, que serán llevadas a cabo por desarrolladores.

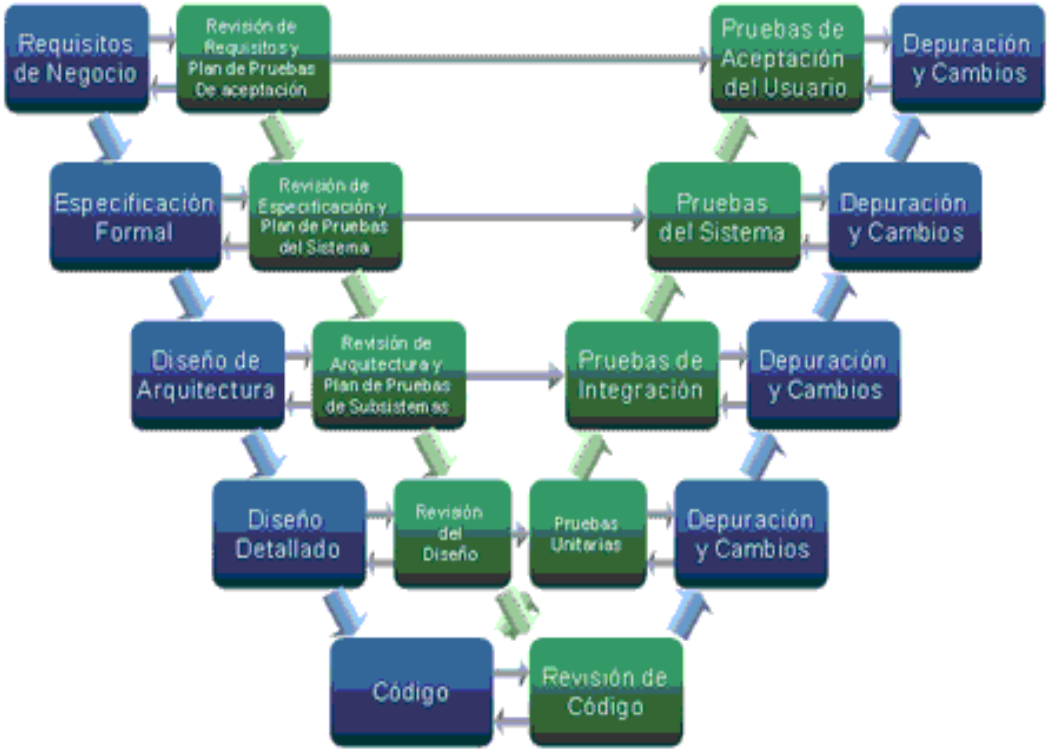



**Figura 1.5. Modelo de Pruebas en W**

En el gráfico del modelo en W las labores de pruebas aparecen en los bloques verdes y en los bloques azules se representan las actividades del ciclo de vida del software antes de su puesta en producción.

Desde el punto de vista del equipo que realiza las pruebas, se pueden distinguir dos momentos de pruebas:

- ✓ **Pruebas sobre productos no software** (revisión de la documentación del proyecto)
  - Revisión de requisitos.
  - Revisión de la especificación de casos de uso.
  - Revisión del diseño de la arquitectura.
  - Revisión del diseño detallado.

- ✓ **Pruebas sobre el software**
  - Revisiones del código y elaboración de las pruebas unitarias.
  - Elaboración de las pruebas de integración de subsistemas.

- Elaboración de las pruebas de sistema.
- Elaboración de las pruebas de aceptación.

Luego de elaborar las pruebas para cada etapa por la que transita el software, se realiza la ejecución de las pruebas y se efectúa la depuración e introducción de cambios si se encuentran errores. [23]

### 1.4.3 Análisis comparativo entre el Modelo V y el Modelo W

Luego de haber realizado un estudio de ambos modelos de pruebas. Se plantea que a través del modelo de pruebas w se guíe el procedimiento de pruebas a definir, debido a que el modelo de pruebas v no contempla la posibilidad de retornar a etapas inmediatamente anteriores, cosa que en la realidad puede ocurrir y se toma toda la complejidad del problema de una vez y no en iteraciones o ciclos de desarrollo. Sin embargo el modelo de pruebas W surge como refinamiento del modelo de pruebas en V, refleja mejor la interdependencia que existe entre el Equipo de Desarrollo y el Equipo de Pruebas a lo largo de todo el proceso de desarrollo del sistema. Además en las primeras etapas se concentran labores de verificación de productos no software y de preparación para la verificación y validación del software; y en las etapas finales se desarrolla todo el esfuerzo de verificación y validación del software producido  cosa que el modelo de pruebas v no se tiene en cuenta.

## 1.5 Procesos de Software

Los procesos de software son una secuencia de pasos requeridos para desarrollar o mantener el software, los cuales sirven de guía para los ingenieros en su trabajo. Estos proporcionan marcos de referencia para aplicar métodos y herramientas.

Por tanto, una práctica disciplinada de los mismos proporciona beneficios si es bien definida, se puede monitorear y mejorar su rendimiento, dando de esta manera la posibilidad de tener un mejor control. Una buena práctica del Proceso Software Personal (PSP) contribuye en gran medida a un buen desarrollo del Proceso Software en Equipos (TSP), es decir, en la manera en que las personas sean capaces de individualmente disciplinarse se contribuirá a que se pueda trabajar en equipo y obtener los productos más rápidos y con mayor eficiencia. A continuación se describen estos dos procesos que contribuyen en gran medida a un buen desarrollo de software.

### 1.5.1 Proceso Software Personal

**Definición**

El Proceso Software Personal (PSP, por sus siglas en inglés) es un proceso de auto mejoramiento diseñado para ayudar a cada persona a controlar y mejorar su manera de realizar el trabajo. Es una estructura que proporciona marcos, guías y procedimientos para desarrollar software. [24]

**Propósito**

El propósito fundamental de PSP es ayudar a los ingenieros de software a ser mejores, puede ayudar a planificar mejor, calcular precisamente el desempeño y medir la calidad de los productos.

**Características**

- ✓ Se concentra en las prácticas de trabajo de los ingenieros de forma individual.
- ✓ Se centra en la administración del tiempo y en la administración de la calidad a través de la eliminación temprana de defectos.
- ✓ Sirve para producir software de calidad.
- ✓ Posee 7 fases las cuales contribuyen a planificar las tareas de cada individuo e ir mejorando la planificación de las mismas, en la medida en que las personas sean realistas y objetivas con los resultados de sus tareas. Esto contribuye a que los individuos puedan manejar mejor los errores que comenten y facilita la mejora de la calidad de sus resultados, contribuyendo a detectar de forma tempranamente los mismos. [25]

**Relación del Proceso Software Personal con las pruebas**

La causa de que los productos muchas veces no tengan la calidad requerida es en ocasiones porque las personas cometen muchos errores en la realización de un software y como resultado de esto aparecen los defectos, los cuales son a menudo la causa principal de los problemas de costes y programaciones.

Un defecto se refiere a algo que esta equivocado en un programa tal como un error sintáctico, una falta de tipografía, un error de puntuación o una sentencia incorrecta del programa. Estos pueden estar en los programas, en los diseños o incluso en los requisitos, las especificaciones o en otra documentación.

Muchos de estos se corrigen cuando se compila y se prueban estos programas, pero generalmente los defectos permanecen en el producto acabado. El Proceso Software Personal define como entender y prevenir los mismos a través del Cuaderno de Registro de Defectos.

**Cuaderno de Registro de defectos**

Esta diseñado para reunir datos de defectos. Este cuaderno permite reunir datos de defectos de cada programa que se codifique, describe cada defecto con bastante detalle para poder entenderlo posteriormente. La utilización del Cuaderno de Registro de Defectos ayuda a la contabilización y organización de los mismos. [26]

### 1.5.2 Proceso Software en Equipo

Debido a los problemas que surgen internamente en los equipos de trabajo como pueden ser: liderazgo ineficiente, demoras, fallas en los compromisos entre otros surge la necesidad de crear un Proceso Software en Equipo a través del cual se pueda medir de alguna forma la calidad del trabajo en equipo durante el desarrollo de un sistema informático.

**Definición**

El Proceso Software en Equipo es complementario al Proceso Software Personal y permite a los equipos desarrollar software de calidad. Es un conjunto de procesos estructurados que indican qué hacer en cada fase del desarrollo del proyecto y muestra cómo conectar cada fase para construir un producto completo. [27]

**Propósito**

El propósito de TSP es tener a todos sus miembros trabajando de forma colaboradora para producir un buen resultado. [28]

**Características**

- ✓ El proceso puede ser adaptado para los diferentes grupos de trabajo.
- ✓ Se basa en el trabajo colectivo e individual pudiéndose medir el desarrollo de cada miembro de forma colectiva e individual**.**
- ✓ Mantiene a todo el equipo luchando junto por un resultado común para todos.

- ✓ Brinda la posibilidad de establecer los objetivos a conveniencia de las personas que integran el equipo a través de los ciclos de desarrollo teniendo en cuenta sus responsabilidades y las siguientes reglas a utilizar:
- ✓ Si no alcanzaste los objetivos del primer ciclo y aún te parecen razonables, úsalos nuevamente.
- ✓ Si las metas te parecen inalcanzables, ponte nuevas metas que sean menos complejas y posibles de alcanzar.
- ✓ Si logras con facilidad alcanzar las metas del primer ciclo, ponte metas más ambiciosas en el próximo ciclo. [29]

**Trabajadores y responsabilidades**

**Líder del Equipo**: Su principal meta es lograr que el equipo funcione de forma efectiva, asumiendo cada cual la tarea que le corresponde. Sus principales responsabilidades son:

- ✓ Construir y mantener un equipo con un trabajo eficiente.
- ✓ Mantener motivados a todos los miembros del equipo para alcanzar las metas del proyecto.
- ✓ Resolver todas las inquietudes que los miembros del equipo le planteen.
- ✓ Mantener al instructor informado de los progresos del equipo.

**Líder de Planificación:** El líder de Planificación debe guiar a los miembros del equipo en elaborar un plan bien detallado, precisando los momentos en los que se evaluaran los progresos del plan. Sus principales responsabilidades son:

- ✓ Producir un plan preciso, completo y adecuado para el equipo y para cada uno de sus miembros.
- ✓ Realizar un adecuado reporte del trabajo realizado cada semana**.**

**Líder de Desarrollo**: El líder de desarrollo debe preocuparse por el funcionamiento del proyecto, garantizar que el producto se desarrolle con alta calidad y efectividad. Sus principales responsabilidades son:

- ✓ Producir un producto superior en cada iteración.
- ✓ Hacer un buen uso de todas las habilidades de los miembros del equipo.

**Líder de Apoyo**: El líder de Apoyo debe asegurar que el soporte que se le de al producto sea adecuado y controlado. Sus principales responsabilidades son:

- ✓ Garantizar que el equipo utilice las herramientas idóneas y los métodos adecuados para realizar su trabajo.
- ✓ Velar porque no se efectúen cambios no autorizados al producto.
- ✓ Registrar todos los riesgos del equipo así como sus soluciones y posteriormente reportarlos cada semana.
- ✓ Discutir y decidir con el equipo de trabajo cuales serán los objetivos que se re-usaran en el siguiente ciclo.

**Líder de Calidad:** El Líder de Calidad debe asegurar que se utilice apropiadamente el TSP para producir un producto libre de defectos. Sus principales responsabilidades son:

- ✓ Todos los miembros del equipo deben reportar adecuadamente el correcto uso de los datos del TSP.
- ✓ El equipo debe seguir las orientaciones del TSP de forma fiel para garantizar la calidad del producto.
- ✓ Garantizar que todas las inspecciones son correctamente reportadas y moderadas.
- ✓ Garantizar que todos los encuentros del equipo del proyecto son reportados adecuadamente y puestos en la libreta de trabajo del proyecto. [30]

### 1.5.3 Análisis valorativo del Proceso Software Personal y Proceso Software en Equipo

La estrategia de PSP enseña a los ingenieros cómo planear y darle un seguimiento a su trabajo, a utilizar un proceso bien definido y medido, a establecer metas mesurables, y finalmente a la utilización del rastreo constante para alcanzar dichas metas. Demuestra además cómo manejar la calidad desde el principio del trabajo, cómo analizar los resultados de cada trabajo, y cómo utilizar los resultados para mejorar el proceso del proyecto siguiente.

Por otro lado TSP se basa en el trabajo colectivo e individual, definiendo para cada uno de los integrantes del equipo objetivos específicos y medibles regidos por los objetivos generales establecidos para el equipo, esto es muy importante pues posibilita medir el desarrollo de cada miembro de forma colectiva e individual. La idea esencial de este proceso es tener a todo el equipo luchando junto por un resultado común para todos, debido a que para tener éxito, los equipos necesitan tener una mezcla de talento y habilidades.

Teniendo en cuenta las buenas prácticas que definen el Proceso Software Personal y el Proceso Software en Equipo, sé plantea utilizar ambos para la propuesta del Equipo de Pruebas a definir en el procedimiento que se propondrá.

## 1.6 Introducción a las pruebas de software

El ciclo de vida de un sistema software empieza en el momento en que nace la idea de desarrollar el sistema y acaba cuando el software por alguna u otra razón deja de ser usado. Entre estos dos momentos el producto software pasa por varias fases: la definición, el desarrollo y el mantenimiento. Las dos últimas fases implican costes muy altos como pérdida de tiempo y recursos, si en el desarrollo del mismo no se tienen en cuenta diferentes aspectos esenciales para obtener en su primera versión el software esperado.

Una de las características típicas durante desarrollo de software basado en el ciclo de vida es la realización de controles periódicos, normalmente coincidiendo con los hitos del proyecto o la terminación de documentos. Estos controles pretenden una evaluación de la calidad de los productos generados para poder detectar posibles defectos cuanto antes. Sin embargo, todo sistema o aplicación, independientemente de estas revisiones, debe ser probado mediante su ejecución controlada antes de ser entregado al cliente. Estas ejecuciones o ensayos de funcionamiento, posteriores a la terminación del código del software, se denominan habitualmente pruebas. [31]

### 1.6.1 Definiciones de las pruebas de software

Una prueba de software es una verificación dinámica del comportamiento del software a partir de un conjunto finito de casos de prueba. Involucra las operaciones del sistema bajo condiciones controladas, evaluando los resultados. [32]

Según Roger Pressman en su libro Un Enfoque Práctico, las pruebas del software son un elemento crítico para la garantía de la calidad del software y representa una revisión final de las especificaciones, del diseño y de la codificación.

Es la ejecución de un programa con la intención de descubrir un error, consistiendo en una técnica experimental para la búsqueda de errores en los programas. [33]

De las definiciones expuestas anteriormente, se ajusta más al trabajo que se presenta, la propuesta que hace Roger Pressman sobre las pruebas, ya que se refiere a las pruebas desde el inicio de las especificaciones hasta el código; eso es precisamente lo que se quiere lograr con este trabajo, plantear las pruebas desde las especificaciones de los requisitos hasta la implementación del código.

### 1.6.2 Objetivos de las pruebas de software

Glen Myers establece como objetivos de las pruebas:

- ✓ La prueba es el proceso de ejecución de un programa con la intención de descubrir un error.
- ✓ Un buen caso de prueba es aquel que tiene una alta probabilidad de mostrar un error no descubierto hasta entonces.
- ✓ Una prueba tiene éxito si descubre un error no detectado hasta entonces.

El objetivo principal es diseñar pruebas que sistemáticamente saquen a la luz diferentes clases de errores, haciéndolo con la menor cantidad de tiempo y de esfuerzo. [34]

### 1.6.3 Características de las pruebas de software

- ✓ Ha de tener una alta probabilidad de encontrar un fallo.
- ✓ No debe ser redundante. Si ya funciona, no se prueba más.
- ✓ Debe ser la “mejor de la cosecha”. Si hay donde elegir, se escoge la mejor.
- ✓ No debería ser ni demasiado sencilla ni demasiado compleja. Si es muy sencilla no aporta nada, si es muy compleja a lo mejor no sabemos lo que ha fallado. [35]

### 1.6.4 Diseño de los casos de pruebas

- ✓ La idea fundamental para el diseño de los casos de prueba consiste en elegir algunas posibilidades de funcionamiento que, por sus características, se consideren representativas del resto.
- ✓ En el caso de que no se detecten defectos en el software al ejecutar dichos casos, se puede tener un cierto nivel de confianza en que el programa no tiene defectos. La dificultad está en saber elegir los casos de prueba que se deben ejecutar. [36]

**Características fundamentales de los casos de pruebas**

Un caso de pruebas es una especificación, usualmente formal, de un conjunto de entradas de prueba, condiciones de ejecución y resultados esperados, identificados con el propósito de hacer una evaluación de aspectos particulares de un elemento objeto de prueba:

- ✓ Los casos de pruebas reflejan trazabilidad con los casos de uso, ya que estos muestran una secuencia ordenada de eventos, al describir flujos básicos, flujos alternos, precondiciones y postcondiciones.
- ✓ Las especificaciones de diseño del Sistema, ya que se debe verificar que el software fue implementado según el diseño y que los elementos arquitectónicos garantizan la calidad del software**.** [37]

**Aspectos fundamentales de los casos de pruebas**

- ✓ Son la base para diseñar y ejecutar los procedimientos de pruebas.
- ✓ La profundidad de las pruebas es proporcional al número de casos de pruebas.
- ✓ El diseño, desarrollo y los recursos necesarios son gobernados por los casos de pruebas requeridos.
- ✓ Han de tener alta probabilidad de detectar un nuevo tipo de error.
- ✓ No han de ser redundante, ni muy simple ni muy complejo.
- ✓ Han de ser representativo para el tipo de error a detectar.
- ✓ Especifican componente a probar, datos de entrada, resultado esperado. [38]

Los casos de uso son la fuente principal de los casos de pruebas, estos se encuentran como entregables del documento Plan de Pruebas de RUP.

Los casos de pruebas están relacionados con el nivel de la prueba y con el tipo de prueba, la cual a su vez contiene la técnica que permite ejecutar el tipo de prueba. Los casos de pruebas proveen las instrucciones para el procedimiento de las pruebas. [39]

### 1.6.5 Estrategia de pruebas de software

La estrategia de prueba describe el enfoque y los objetivos generales de las actividades de pruebas. Incluye los niveles de pruebas (unidad, integración, sistema, aceptación) y el tipo de prueba a ser ejecutadas. La estrategia define:

- ✓ Técnicas de pruebas (manual o automática) y herramientas a ser usadas.
- ✓ Que criterios de éxitos y culminación de la prueba serán usados.
- ✓ Consideraciones especiales afectadas por requerimientos de recursos o que tengan implicaciones en la planificación.[40]

**Objetivos de la estrategia**

- ✓ Planificar las pruebas necesarias en cada iteración, incluyendo las pruebas de unidad, integración y las pruebas de sistema.
- ✓ Diseñar e implementar las pruebas creando los casos de prueba que especifican qué probar, cómo realizar las pruebas y creando, si es posible, componentes de prueba ejecutables para automatizar las pruebas.
- ✓ Realizar diferentes pruebas y manejar los resultados de cada prueba sistemáticamente. Los productos de desarrollo de software en los que se detectan defectos son probadas de nuevo y posiblemente devueltas a otra etapa, como diseño o implementación, de forma que los defectos puedan ser arreglados. [41]

### 1.6.6 Niveles de las pruebas de software

La estrategia de pruebas define diferentes tipos de niveles por los que pasa una prueba. Estos están organizados de forma espiral comenzando con el nivel de unidad, posteriormente le sigue el nivel de integración, luego el nivel de sistema y por último el nivel de aceptación.

**Nivel de Unidad**

En este nivel de prueba fundamentalmente se ejecutan casos de pruebas de caja blanca diseñados para:

- ✓ Probar las estructuras de datos locales para asegurar que los datos que se mantienen temporalmente, conservan su integridad durante todos los pasos de ejecución del algoritmo.
- ✓ Ejercitar todos los caminos independientes (caminos básicos) de la estructura de control con el fin de asegurar que todas las sentencias del módulo se ejecutan por lo menos una vez. Y, finalmente, se prueban todos los caminos de manejo de errores. [42]

**Pruebas de Unidad**

Las pruebas de unidad son un proceso para probar los subprogramas, las subrutinas, los procedimientos individuales o las clases en un programa. Algunas de sus caracterisiticas son:

- ✓ Las pruebas de unidad son una manera de manejar los elementos de prueba combinados, puesto que se centra la atención inicialmente en unidades más pequeñas del programa.
- ✓ La prueba de una unidad facilita la tarea de eliminar errores, puesto que, cuando se encuentra un error, se sabe que existe en un módulo particular.

- ✓ Las pruebas de unidad introducen paralelismo en el proceso de pruebas del software presentándose la oportunidad de probar los múltiples módulos simultáneamente.[43]

Las pruebas que se realizan como parte de la prueba de unidad son:

- ✓ Prueba a la interfaz del módulo para asegurar que la información fluye de forma adecuada hacia y desde la unidad de programa que está siendo probada.
- ✓ Se examinan las estructuras de datos locales para asegurar que los datos que se mantienen temporalmente conservan su integridad durante todos los pasos de ejecución del algoritmo.
- ✓ Se prueban las condiciones limites para asegurar que el módulo funciona correctamente en los limites establecidos como restricciones de procesamiento.
- ✓ Se ejercitan todos los caminos independientes (caminos básicos) de la estructura de control con el fin de asegurar que todas las sentencias del módulo se ejecutan por lo menos una vez.
- ✓ Se prueban todos los caminos de manejo de errores. [44]

**Nivel de Integración**

En el nivel de integración se realiza el proceso de unión de los módulos, con el objetivo de tomar los módulos probados mediante la prueba de unidad y construir una estructura de programa que esté de acuerdo con lo que dicta el diseño. [45]

**Pruebas de Integración**

Las pruebas de integración son una técnica sistemática para construir la estructura del programa mientras que, al mismo tiempo, se llevan a cabo pruebas para detectar errores asociados con la interacción.

Se realizan durante la construcción del sistema. Con el objetivo de verificar la correcta unión de los componentes una vez que han sido probados unitariamente con el fin de comprobar que interactúan correctamente a través de sus interfaces, tanto internas como externas, cubren la funcionalidad establecida y se ajustan a los requisitos no funcionales especificados en las verificaciones correspondientes. [46]

**Nivel de Sistema**

En el nivel de sistema las pruebas las realizan personas que no están vinculadas directamente con el software construido, estas pueden ser especialistas en el tema, para de esta manera encontrar las deficiencias, que el mismo pueda tener y que sus desarrolladores no lo hayan tenido en cuenta.

**Pruebas de Sistema**

Se realizan durante la construcción del sistema (partes completas). Con el objetivo de probar fondo el mismo, verificando que cada elemento encaja de forma adecuada y comprobando su funcionalidad e integridad globalmente, en un entorno lo más parecido posible al entorno final de producción.

**Nivel de Aceptación**

Luego de terminado el software y de la implantación en el entorno de producción, es necesario saber la aceptación del mismo de acuerdo con los requisitos de funcionalidad y rendimiento especificados por los usuarios, por tanto se han creado pruebas para determinar la aceptación del producto.

**Pruebas de Aceptación**

Las pruebas de aceptación son definidas por el usuario del sistema y preparadas por el equipo de desarrollo, aunque la ejecución y aprobación final corresponden al usuario.

La mayoría de los desarrolladores de productos de software dedicado a las pruebas de aceptación, llevan a cabo un proceso denominado pruebas alfa y beta para descubrir errores que parezca que sólo el usuario final puede descubrir.

- ✓ **Prueba alfa***:* Se lleva a cabo, por un cliente, en el lugar de desarrollo. Se usa el software de forma natural con el desarrollador como observador del usuario y registrando los errores y problemas de uso. Las pruebas alfa se llevan a cabo en un entorno controlado.

- ✓ **Prueba beta***:* Se llevan a cabo por los usuarios finales del software en los lugares de trabajo de los clientes. A diferencia de la prueba alfa, el desarrollador no está presente normalmente. Así, la prueba beta es una aplicación en vivo del software en un entorno que no puede ser controlado por el desarrollador. El cliente registra todos los problemas que encuentra durante la prueba beta e informa a intervalos regulares al desarrollador. [47]

### 1.6.7 Métodos de prueba de software

Los métodos de prueba del software tienen el objetivo de diseñar pruebas que descubran diferentes tipos de errores con menor tiempo y esfuerzo. Estos métodos son:

- ✓ Método de Caja Blanca (estructural).
- ✓ Método de Caja Negra (funcional).[48]

**Pruebas de Caja Blanca o estructurales**

Las pruebas de Caja Blanca requieren del conocimiento de la estructura interna del programa y son derivadas a partir de las especificaciones internas de diseño o el código. Se basa en el minucioso examen de los detalles procedimentales. Se comprueban los caminos lógicos del software, proponiendo casos de prueba que examinen que están correctas todas las condiciones y/o bucles, para determinar si el estado real coincide con el esperado o afirmado. Esto genera gran cantidad de caminos posibles por lo que hay que dedicar esfuerzos a la determinación de las condiciones de prueba que se van a verificar. [49]

**Prueba del Camino Básico**

La prueba del Camino Básico es una técnica de prueba de Caja Blanca propuesta por Tom McCabe. Esta técnica permite obtener una medida de la complejidad lógica de un diseño y usar esta medida como guía para la definición de un conjunto básico.

Los pasos que se siguen para aplicar esta técnica son:

1. A partir del diseño o del código fuente, se dibuja el grafo de flujo asociado.
2. Se calcula la complejidad ciclomática del grafo.
3. Se determina un conjunto básico de caminos independientes.
4. Se preparan los casos de prueba que obliguen a la ejecución de cada camino del conjunto básico.

Los casos de prueba derivados del conjunto básico garantizan que durante la prueba se ejecuta por lo menos una vez cada sentencia del programa.

**Pasos para dibujar un grafo de flujo**

1. Señalar sobre el código cada condición de cada decisión tanto en sentencias If-then y Case-of como en los bucles While o Until.

2. Agrupar el resto de las sentencias en secuencias situadas entre cada dos condiciones.
3. Numerar tanto las condiciones como los grupos de sentencias, de manera que se les asigne un identificador único. *Anexo 1* [50]

**Complejidad Ciclomática**

La complejidad ciclomática es una métrica de software extremadamente útil pues proporciona una medición cuantitativa de la complejidad lógica de un programa. El valor calculado como complejidad ciclomática define el número de caminos independientes del conjunto básico de un programa y nos da un límite superior para el número de pruebas que se deben realizar, para asegurar que se ejecute cada sentencia al menos una vez. [51]

**Formas fundamentales de calcular la complejidad**

1) El número de regiones del grafo de flujo coincide con la complejidad ciclomática.
2) La complejidad ciclomática, *V(G)*, se define como:
   *V(G) = A - N + 2* donde: *A* es el número de aristas del grafo y *N* es el número de nodos.
3) La complejidad ciclomática, *V(G)*, también se define como:
   *V(G) = P +* 1, donde: *P* es el número de nodos predicado contenidos en el grafo *G*. [52]

Se recomienda hacer el cálculo por más de una vía para evitar equivocaciones

**Obtención de los Caminos Básico**

Para determinar los caminos básicos se siguen los siguientes pasos:

1) Se usa el valor calculado como complejidad Ciclomática, ya que define el número de caminos independientes y da un límite superior para el número de pruebas que se deben realizar para asegurar que se ejecute cada sentencia al menos una vez.
2) Se toma como un camino independiente a todos los camino del programa que introducen al menos un nuevo conjunto de sentencias de procesamiento o una nueva condición.
3) A partir del grafo de flujo se obtienen los caminos independientes siguiendo las aristas del grafo desde el nodo inicial hasta el final.

**Prueba de Condición**

Es un método de diseño de casos de prueba que ejercita las condiciones lógicas contenidas en el módulo de un programa.

Tipos de errores que pueden aparecer en una condición**:**

- ✓ Existe un error en un operador lógico.
- ✓ Existe un error en un paréntesis lógico.
- ✓ Existe un error en un operador relacional.
- ✓ Existe un error en una expresión aritmética. [53]

**Prueba de Flujo de Datos**

Se seleccionan caminos de prueba de un programa de acuerdo con la ubicación de las definiciones y los usos de las variables del programa.[54]

**Prueba de Bucles**

Es una técnica de prueba de Caja Blanca que se centra exclusivamente en la validez de las construcciones de bucles.

N es el número máximo de iteraciones permitidos por el bucle. *Anexo 20*

- ✓ Pasar por alto totalmente el bucle
- ✓ Pasar una sola vez por el bucle
- ✓ Pasar dos veces por el bucle
- ✓ Hacer m pasos por el bucle con m < n
- ✓ Hacer n-1, n y n + 1 pasos por el bucle [55]

**Pruebas de Caja Negra o funcionales**

Las pruebas de Caja Negra son pruebas funcionales. Se parte de los requisitos funcionales, a muy alto nivel, para diseñar pruebas que se aplican sobre el sistema sin necesidad de conocer como está construido por dentro. Las pruebas se aplican sobre el sistema empleando un determinado conjunto de datos de entrada y observando las salidas que se producen para determinar si la función se está desempeñando correctamente por el sistema bajo prueba. [56]

**Técnica de Particiones o Clases de Equivalencia**

La Partición Equivalente es una técnica del método de Caja Negra que divide el campo de entrada de un programa en clases de datos.

- ✓ Una condición de entrada es un valor numérico específico, un rango de valores, un miembro de un conjunto de valores o lógica.
- ✓ Una clase de equivalencia representa un conjunto de estados válidos y no válidos para una condición de entrada.

- ✓ La prueba de Partición Equivalente, se basa en evaluar las clases de equivalencia para una condición de entrada. [57]

Una de las cualidades que definen un buen caso de prueba, es que cada caso debe cubrir el máximo número de entradas. Además debe tratarse el dominio de valores de entrada dividido en un número finito de clases de equivalencia que cumplan la siguiente propiedad: la prueba de un valor representativo de una clase permite suponer «razonablemente» que el resultado obtenido (existan defectos o no) será el mismo que el obtenido probando cualquier otro valor de la clase. [58]

**Pasos para identificar clases de equivalencias**

1) Se examina cada condición de entrada y se divide en dos o más grupos.
2) Se identifican dos tipos de clases:
   - ✓ Clases de equivalencia válidas.
   - ✓ Clases de equivalencia no válidas.

| Condición de entrada | Clases de Equivalencia Válidas | Clases de Equivalencia No Válidas |
|---|---|---|
| | | |

Si la condición de entrada es un:

- ✓ Rango, se define una clase de equivalencia válida y dos no válidas.
- ✓ Valor específico, se define una clase de equivalencia válida y dos no válidas.
- ✓ Miembro de conjunto, se define una clase de equivalencia válida y otra no válida.
- ✓ Lógica, se define una clase válida y otra no válida.

3) Asignar un número único a cada clase de equivalencia.
4) Escribir casos de prueba que cubran tantas clases válidas no incorporadas como sea posible hasta que se cubran todas las clases de equivalencia válidas
5) Escribir casos de prueba que cubran una sola clase no válida no incorporada hasta que se cubran todas las clases de equivalencia no válidas. [59]

**Análisis de Valores Límites (AVL)**

La técnica de Análisis de Valores Límites selecciona casos de prueba que ejerciten los valores límites. La misma complementa la prueba de Partición Equivalente, pero en lugar de realizar la prueba con cualquier elemento de la partición equivalente, se escogen los valores en los bordes de la clase. Más que concentrarse únicamente en el dominio de entrada (condiciones de entrada), los casos de prueba se generan considerando también el espacio de salida. [60]

**Reglas:**

1) Si una condición de entrada especifica un rango de valores, se deben generar casos para los extremos del rango y casos no válidos para situaciones justo más allá de los extremos.
2) Si la condición de entrada especifica un número de valores, hay que escribir casos para los números máximo, mínimo, uno más del máximo y uno menos del mínimo de valores.

3) Usar la regla 1 para la condición de salida

4) Usar la regla 2 para cada condición de salida

En esta regla 3 y 4, debe recordarse que:

- ✓ Los valores límite de entrada no generan necesariamente los valores límite de salida
- ✓ No siempre se pueden generar resultados fuera del rango de salida (pero es interesante considerarlo).

5) Si la entrada o la salida de un programa es un conjunto ordenado (por ejemplo, una tabla, un archivo secuencial,...), los casos deben concentrarse en el primero y en el último elemento. [61]

**Prueba de Comparación**

Para aplicaciones críticas es sugerente desarrollar versiones de software independientes, incluso aunque solo se vaya a distribuir una de las versiones en el sistema final basado en computadora. Esas versiones independientes son la base de una técnica de prueba de Caja Negra denominada prueba de Comparación o prueba mano a mano.

Cuando se han producido múltiples implementaciones de la misma especificación, a cada versión del software se le proporciona como entrada los casos de prueba diseñados mediante alguna otra técnica de Caja Negra. Si las salidas producidas por las distintas versiones son idénticas, se asume que todas las implementaciones son correctas. Si la salida es diferente, se investigan todas las aplicaciones para determinar el defecto responsable de la diferencia en una o más versiones. [62]

### 1.6.8 Tipos de Pruebas de software

**Pruebas de Funcionalidad**

- ✓ **Función**: Pruebas fijando su atención en la validación de las funciones, métodos, servicios y caso de uso.
- ✓ **Seguridad**: Asegurar que los datos o el sistema solamente es accedido por los actores deseados.
- ✓ **Volumen**: Enfocada en verificar las habilidades de los programas para manejar grandes cantidades de datos, tanto como entrada, salida o residente en la base de datos.

**Pruebas de Usabilidad**

- ✓ **Usabilidad**: Prueba enfocada a factores humanos, estéticos, consistencia en la interfaz de usuario, ayuda sensitiva al contexto y en línea, asistencia a documentación de usuarios y materiales de entrenamiento.

**Pruebas de Fiabilidad**

- ✓ **Integridad**: Enfocada a la valoración de la robustez (resistencia a fallos).
- ✓ **Estructura**: Enfocada a la valoración unida a su diseño y formación. Este tipo de prueba se realiza a las aplicaciones web asegurando que todos los enlaces están conectados, que el contenido deseado es mostrado y no hay contenido huérfano.
- ✓ **Stress**: Enfocada a evaluar como el sistema responde bajo condiciones anormales. Ejemplos: Extrema sobrecarga, insuficiente memoria, servicios y hardware no disponible y recursos compartidos no disponibles.

**Pruebas de Performance**

- ✓ **Benchmark** (Rendimiento): Es un tipo de prueba que compara el rendimiento de un elemento nuevo o desconocido a uno de carga de trabajo de referencia conocido.
- ✓ **Contención**: Enfocada a la validación de las habilidades del elemento a probar para manejar aceptablemente la demanda de múltiples actores sobre un mismo recurso (registro de recursos, memoria).
- ✓ **Carga**: Usada para validar y valorar la aceptabilidad de los limites operacionales de un sistema bajo carga de trabajo variable, mientras el sistema bajo pruebas permanece constante. La variación en carga es similar a la carga de trabajo promedio y con picos que ocurre dentro de tolerancias operacionales normales.[63]

## 1.7 Conclusiones

Una vez analizados los temas abordados durante el capítulo, se han llegado a las siguientes conclusiones:

- ✓ La metodología de desarrollo de software a elegir para guiar el proceso de pruebas a definir es la metodología de RUP.
- ✓ Se escogerán de forma híbrida algunos roles, artefactos y actividades propuestos por los flujos de trabajo de pruebas analizados.
- ✓ El Modelo de pruebas por el cual se guía el procedimiento de pruebas a definir es el Modelo de Pruebas W.
- ✓ Se seleccionaron algunas de las buenas prácticas que propone el Proceso Software Personal.
- ✓ Se eligieron algunos de los roles propuestos por el Proceso Software en Equipo.
- ✓ Las pruebas de software son un aspecto determinante en la calidad de los productos software.

# Capítulo 2: Actualidad de las pruebas de software

## 2.1 Introducción

En este capítulo se realiza una caracterización sobre el estudio y utilización de las pruebas de software en los diversos proyectos productivos que se desarrollan en la Universidad de las Ciencias Informáticas y en especial en el área temática Sistema de Apoyo a la Salud de la Facultad 7. Para dar cumplimiento a este objetivo se realizó un diagnóstico con vista a detectar los conocimientos y puesta en práctica de las pruebas en las aplicaciones que se desarrollan en la misma. Para esto se emplearon diferentes técnicas como son las entrevistas y encuestas a los estudiantes y los profesores líderes que pertenecen a los proyectos del área.

## 2.2 Aplicación de técnicas para el diagnóstico

Las principales técnicas empleadas que permitieron fundamentar la investigación de este trabajo fueron la entrevista y la encuesta.

### 2.2.1 Entrevista

La entrevista es una conversación planificada entre el investigador y el entrevistado para obtener información. Su uso constituye un medio para el conocimiento cualitativo de los fenómenos o sobre características personales del entrevistado y puede influir en determinados aspectos de la conducta humana por lo que es importante una buena comunicación. [64]

### 2.2.2 Encuesta

Es una técnica que tiene como objetivo obtener cierta información deseada de un sujeto preseleccionado de antemano dentro de una muestra representativa, por medio de una conversación directa cuyas pautas vienen indicadas en un guión o cuestionario que ha sido previamente diseñado y probado en una muestra piloto o muestra inicial. Se utiliza cuando la información que se realiza puede ser obtenida a partir de la respuesta de una o varias personas a través del uso de un cuestionario pre elaborado.

La encuesta es semejante a la entrevista pero escrita, donde a través de un conjunto de preguntas se pretende obtener una información sobre el mundo interior del encuestado o su percepción. [65]

### 2.2.3 Muestreo Aleatorio Estratificado

Para la determinar la cantidad de población que será encuestada, y la cantidad de encuestas que se realizarán en el área temática Sistema de Apoyo a la Salud se utilizó el método estadístico Muestreo Aleatorio Estratificado.

El Muestreo Aleatorio Estratificado consiste en subdividir la población en subpoblaciones de tal forma que la unión de ellos será la población, y la intersección de cualquiera de los dos dará como resultado el conjunto vacío, es decir, no tendrán elementos comunes.

A las subpoblaciones se les llaman estratos, se trata de conformar estos de modo que los elementos dentro de ellas sean homogéneos. El tamaño de la muestra se distribuirá entre los estratos, en función de distintos criterios, pero lo que caracteriza al Muestreo Aleatorio Estratificado es que la selección de la muestra de cada estrato se hace bajo el procedimiento de Muestreo Irrestricto Aleatorio y se realiza independientemente de los diferentes estratos.

La población para la muestra es el total de estudiantes que hay en los proyectos del área temática.

**Notación**

Se supone que la población consta de n unidades y están distribuidas en L estratos, constituyen una partición de la población; se representará por *Nh* en el ni de u en el estado h-ésimo, de aquí:.

$$N = N_1 + N_2 + \ldots + N_h + \ldots + N_l$$

Total de población:

$$x_h = \sum_{1}^{N_h} x_{hi} \quad \text{media} \quad \bar{x}_h = \frac{1}{N_h} \sum_{i}^{N_h} x_{hi}$$

Fracción de muestreo del estrato:

$$f_h = \frac{n_h}{N_h}$$

Si el tamaño de la muestra de los estratos se distribuye proporcionalmente al número de unidades en el estrato, es decir si se cumple que:

$$n_h = n\frac{N_h}{N}$$ Para todo h. En este caso se dice que la distribución de la muestra se ha hecho con asignación proporcional.

Para la determinación del tamaño de muestra de una población con $S^2$ desconocida se puede determinar mediante la expresión:

$$n = \frac{\left(\frac{Z_{1-\frac{\alpha}{2}}}{d}\right)^2 P(1-P)}{1+\frac{1}{N}\left(\frac{Z_{1-\frac{\alpha}{2}}}{d}\right)^2 P(1-P) - \frac{1}{N}}$$

Donde:

1-α: Nivel de confianza

*d: E*rror absoluto

*N*: Tamaño de la muestra [66]

*z*1: Percentil de la distribución normal

*P*: Proporción de la población

Realizando el cálculo del tamaño de muestra para un nivel de confianza 1-α=90%, con un error absoluto d=0.10, se obtiene un percentil de la distribución normal *z*1=1.64 y se asume como proporción de la población P=0.5, *N=217* que es la cantidad total de los integrantes de los proyectos del área temática Sistema de Apoyo a la Salud; se obtiene que:

$$n = \frac{\left(\frac{1.64}{0.10}\right)^2 0.5(1-0.5)}{1+\frac{1}{217}\left(\frac{1.64}{0.10}\right)^2 0.5(1-0.5) - \frac{1}{217}} = \frac{67.24}{1,26496} = 53,15 \approx 53$$

Se deben aplicar 53 encuestas en el área temática ¨Sistema de Apoyo a la Salud¨.

Conociendo que n=53, *Nh* es la cantidad de integrantes de cada proyecto, N=217 que es el total de integrantes de los diferentes proyectos del área temática.

Calculando el tamaño de la muestra por estrato (se considera estrato a cada proyecto del área temática) aplicando $n_h = n\frac{N_h}{N}$ [67]; resultan los tamaños de muestra por estrato que se muestran en la siguiente tabla

**Tabla 1. Tamaño de muestra por estratos de población**

| Estratos | Cantidad de estudiantes | Tamaño de la muestra |
|---|---|---|
| Balance Material | 17 | 4 estudiantes |
| Docencia Médica | 36 | 9 estudiantes |
| Colaboración Médica | 16 | 4 estudiantes |
| Atención Remota a Servidores | 10 | 2 estudiantes |
| Portal de la Facultad 7 | 4 | 1 estudiante |
| Estadística | 24 | 6 estudiantes |
| Estudiantes atendidos por el área temática | 110 | 27 estudiantes |

## 2.3 Actualidad de las pruebas de software en el ámbito Internacional y Nacional

Una de las grandes problemáticas que aún continúa vigente en el desarrollo de sistemas software es la calidad con que los mismos se desarrollan, donde a pesar de que es una realidad para las personas involucradas en estas actividades que los fallos de software ocasionan graves pérdidas económicas, el nivel de conciencia en cuanto a la necesidad de incorporar mecanismos de calidad es muy pequeño.

Desde los años 70, las actividades de pruebas han duplicado el porcentaje que suponían dentro del coste total de los proyectos hasta representar en la actualidad entre el 40 y el 80%. Esta situación se ha originado debido al aumento del tamaño y la progresiva complejidad de los Sistemas de Información (múltiples plataformas y tecnologías, sistemas de empresa interconectados y extendidos con tecnologías Web), el fuerte incremento de los riesgos y la integración de nuevos servicios, que permite la gestión de las pruebas en forma de proceso.

Por tanto hoy en día la evaluación de la calidad a través del proceso de prueba se ha convertido en una tarea vital en el desarrollo de cualquier sistema informático, puesto que con el paso de los años se ha determinado la importancia que tienen para garantizar el tiempo, el costo y la calidad del producto, de tal forma que actualmente son un proceso cuyo propósito principal es evaluar la funcionalidad del software respecto a los requerimientos establecidos al inicio.

En Cuba con el objetivo de informatizar los procesos en diferentes sectores se han trazado algunas estrategias, entre las cuales se pueden mencionar: la creación de la Universidad de las Ciencias Informáticas, empresas de desarrollo de software y carreras Informáticas, en las cuales la idea fundamental es el desarrollo de software. Con el objetivo de desarrollar sistemas informáticos con la calidad requerida se han creado empresas como Sigta y Citmatel, las cuales ofrecen servicios de personal especializado para la planificación y ejecución de pruebas de software siguiendo metodologías internacionales y se han insertados cursos relacionados con la calidad del software en las universidades.

## 2.4 Actualidad de las pruebas de software en la UCI

La UCI constituye un nuevo modelo de formación–investigación–producción que ofrece amplias posibilidades al desarrollo de la Industria Cubana del Software. Actualmente se están desarrollando disímiles software tanto de producción nacional como de exportación. Teniendo en cuenta todo este proceso se han incrementando los estudios relacionados con las pruebas de software, ya que es muy importante que los productos al ser terminados tengan la calidad requerida para ser utilizados.

Con el objetivo de obtener una valoración general del conocimiento de las pruebas en la UCI, se realizó una encuesta a 80 estudiantes vinculados a proyectos productivos de tercero a quinto año, pertenecientes a 5 facultades (1, 2, 4, 5, 7) de la Universidad.

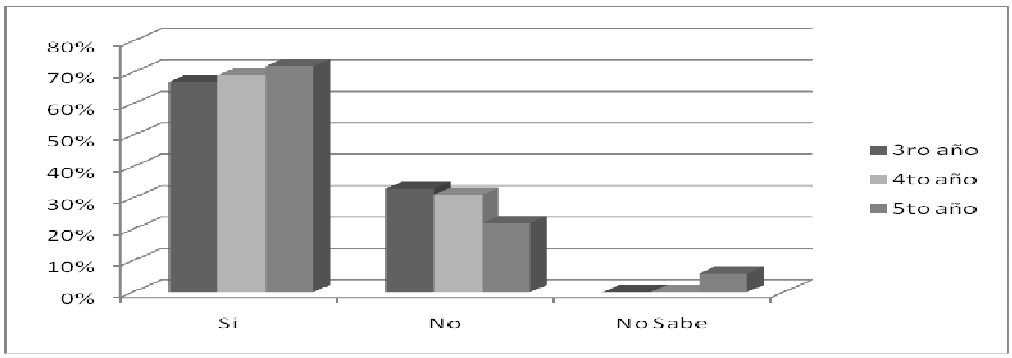



**Figura 2.1. Conocimientos de pruebas de software**

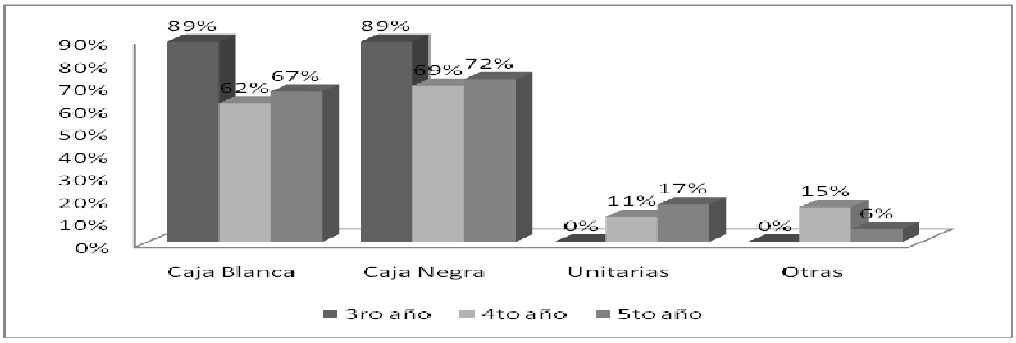



**Figura 2.2. Conocimientos sobre diversos tipos de pruebas**

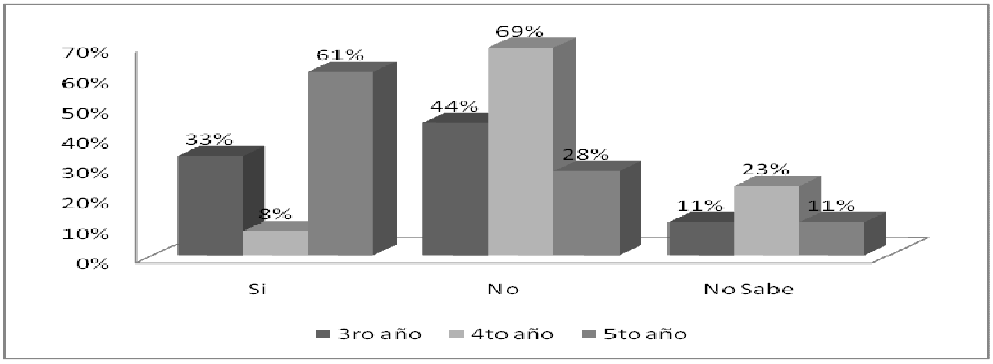



**Figura 2.3. Aplicación de pruebas a los productos software que se desarrollan en los proyectos productivos**

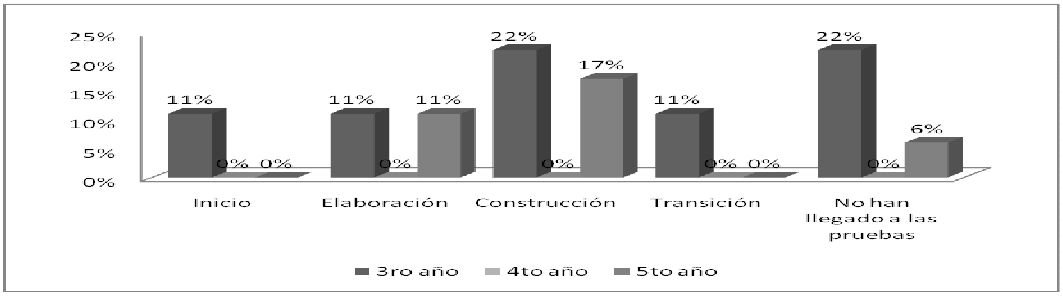


**Figura 2.4. Fases en que se aplican las pruebas a los productos software que se desarrollan en los proyectos productivos**

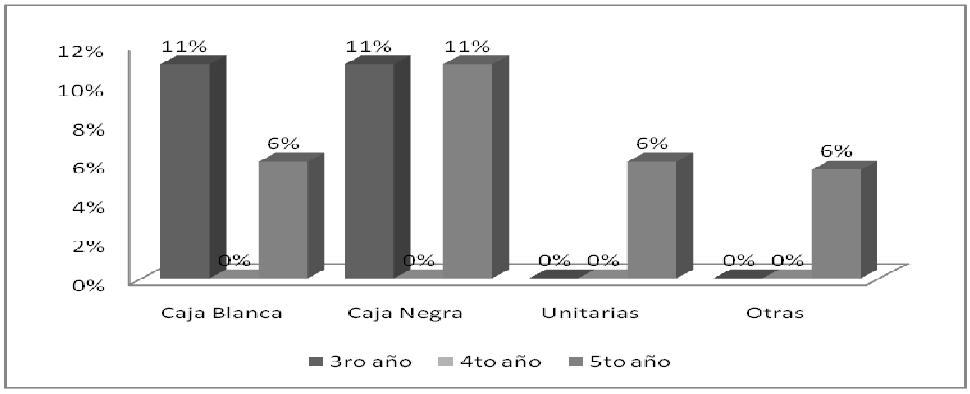



**Figura 2.5. Tipos de pruebas que se aplican a los productos software que se desarrollan en los proyectos productivos**

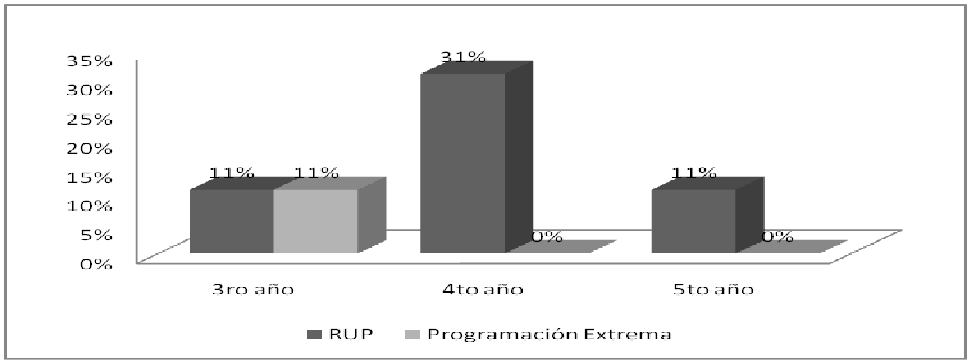



**Figura 2.6. Metodologías de desarrollo de software que se aplican en los proyectos productivos**

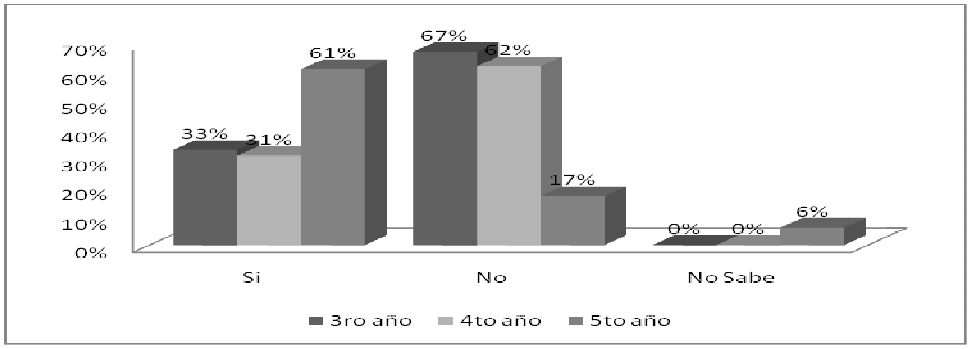



**Figura 2.7. Roles definidos en los proyectos para aplicar pruebas de software**

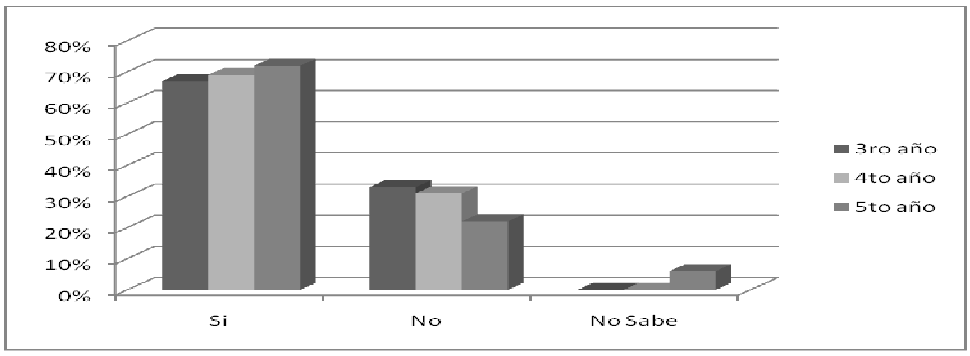



**Figura 2.8. Conocimiento sobre flujo de trabajo de pruebas**

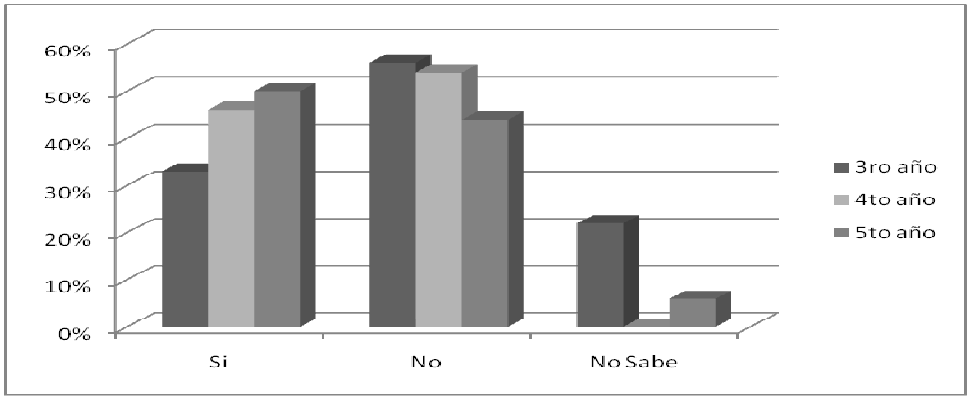



**Figura 2.9. Conocimiento sobre procedimientos de pruebas**

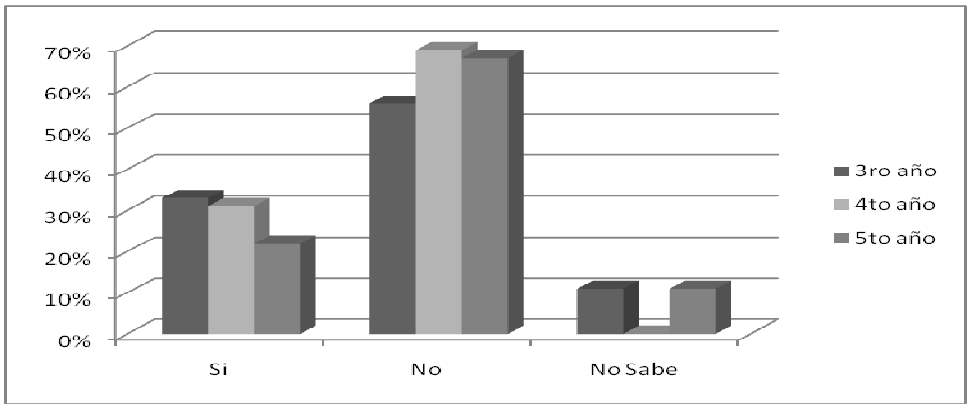



**Figura 2.10. Aplicación en los proyectos productivos de un de flujo de trabajo de pruebas**

### 2.4.1 Análisis de los resultados obtenidos

A partir del análisis de las gráficas expuestas se evidencia que los estudiantes de tercero a quinto año vinculados a los proyectos productivos tienen conocimientos sobre las pruebas de software,

esencialmente pruebas de Caja Blanca y Caja Negra, reflejando que poseen muy pocos conocimientos sobre las pruebas Unitarias, Rendimiento e Integración.

Se demostró que para la aplicación de la pruebas se definen algunos roles propuestos por el flujo de trabajo de pruebas que propone la metodología de RUP, la cual se evidencia que es la metodología de desarrollo de software que más se utiliza en los proyectos productivos de la UCI.

A pesar de que los estudiantes poseen conocimientos de las pruebas, se refleja un concepto erróneo de la etapa en que deben aplicarse y solo en algunos proyectos se aplican las mismas a los productos que se desarrollan. Por lo general se efectúan en la fase de transición aplicando algunas pruebas de Caja Blanca y Caja Negra.

En la encuesta realizada se evidencia que los estudiantes no poseen conocimientos sobre lo que es un flujo de trabajo de pruebas y se conoce poco sobre lo que es un procedimiento de pruebas, por lo cual en muy pocos proyectos se aplican. De aquí se deriva la importancia de este trabajo.

## 2.5 Caracterización del área temática Sistema de Apoyo a la Salud

En el área temática Sistema de Apoyo a la Salud (SAS) se agrupan soluciones informáticas que se le dan a un grupo de procesos médicos dentro del Sistema Nacional de Salud.

Sus objetivos fundamentales son:

- ✓ Desarrollar productos software vinculados esencialmente con la salud.
- ✓ Dar soporte técnico a los productos implementados en la misma.

Esta área temática cuenta con 6 proyectos los cuales no están relacionados directamente con servicios a los pacientes. Estos proyectos son:

1. Sistema para la Planificación y el Balance de Materiales Gastables en el Ministerio de Salud Pública.
2. Sistema de Gestión de Información en el Proceso de Formación de Recursos Humanos en la Salud.
3. Atención Remota a Servidores.
4. Colaboración Médica.
5. Informatización de la Facultad7.

6. Sistemas de Información Estadística Complementario de Salud (SIEC-Salud).

### 2.5.1 Caracterización de los proyectos del área temática SAS

**Sistema para la Planificación y el Balance de Materiales Gastables en el Ministerio de Salud Pública (Balance Material)**.

Objetivo del proyecto

La idea principal de este proyecto es realizar un software informático que informatice todo el proceso de gestión y manipulación de los materiales de uso médico del Ministerio de Salud, desde la planificación hasta la compra y distribución del mismo.

Características generales del proyecto

- ✓ El proyecto esta constituido por dos profesores y varios estudiantes que desempeñan los roles de documentadores, que son los encargados de generar toda la documentación definida por la metodología de desarrollo RUP, además de otros documentos como el Manual de Usuario; y el rol de programador que es responsables de diseñar la interfaces gráficas, diseñar la base de datos y generar el código fuente.
- ✓ Actualmente se está realizando una nueva versión del módulo de Planificación y se está en la fase inicial del módulo de Balance y Distribución.
- ✓ Se realizan pruebas exploratorias a los sistemas que están en desarrollo sin la documentación requerida.

**Atención Remota a Servidores**

Objetivo del proyecto

Este proyecto no posee un proceso a informatizar, sino que el mismo se encarga de prestarle servicios a clientes (proyectos) administrando sus servidores, así como realizar investigaciones sobre diferentes temas que sirvan de soporte al desarrollo de los proyectos de la Facultad 7.

Características generales del proyecto

- ✓ Es un proyecto investigativo.
- ✓ No posee un producto en especifico al cual realizarle pruebas, por tanto, no posee clasificación de roles que están definidos en la metodología de RUP para los proyectos de gestión.

- ✓ Parte de los alumnos que integran este proyecto poseen conocimientos de varios tipos de pruebas de rendimiento, fundamentalmente pruebas de Stress.
- ✓ Prestan servicios de pruebas a otros proyectos que se lo solicitan utilizando la herramienta Jmater.

**Sistema de Gestión de Información en el Proceso de Formación de Recursos Humanos en la Salud (Docencia médica)**

Objetivo del proyecto

Desarrollar un sistema para la informatización de los procesos de gestión docente durante la formación de recursos humanos en las universidades médicas cubanas.

Características generales del proyecto

- ✓ Existen 9 módulos que se encuentran en la fase de desarrollo.
- ✓ Están definidos roles de programador, diseñador de interfaz, analista, diseñador de base de datos y un equipo de consultas que es responsable de comprobar el funcionamiento de la base de datos.
- ✓ Se han realizado pruebas de Caja Negra por parte del equipo del proyecto donde no se ha tenido en cuenta toda la estructura de las pruebas, es decir el rigor que llevan realizar las mismas.

**Colaboración Médica**

Objetivo del proyecto

Realizar una aplicación web que permita gestionar la información del colaborador de la salud y el control del pago al mismo.

Características generales del proyecto

- ✓ Este proyecto es una aplicación web, con filosofía cliente servidor y basada en una arquitectura orientada a servicios.
- ✓ Está dividido en 3 módulos: Administración, Gestión de la Información y Pago al Colaborador.
- ✓ En este proyecto no se realizan ningún tipo de pruebas a la fase de desarrollo en que se encuentra.
- ✓ Los roles definidos son Analista, Diseñador de BD y Programador

**Portal de la Facultad 7**

Objetivo del proyecto

Realizar un portal que digitalice todos los eventos, procesos y actividades que se realizan en la facultad.

Características generales del proyecto

- ✓ Los roles que tienen definidos son: programador y diseñador.
- ✓ Consta de 2 módulos en fase de implantación y un módulo listo para revisar en el laboratorio de calidad de la facultad.
- ✓ No se realizan pruebas necesarias para verificar la calidad de los módulos que están implementando.

**Sistemas de Información Estadística Complementario de Salud (Estadística)**

Objetivo del proyecto

Informatizar el sistema estadístico de salud en Cuba.

Características generales del proyecto

- ✓ Los roles que tienen definidos son: jefe de proyecto, jefe de módulo, analista, programador y revisores técnicos.
- ✓ Consta de 5 módulos, todos en fase de implementación.
- ✓ Se realizan pruebas de Caja Negra.

Es importante destacar que la información de las características de los proyectos del área temática se obtuvo a partir de las entrevistas realizadas a los líderes de los proyectos.

### 2.5.2 Diagnóstico de las pruebas de software en el área temática SAS

Para hacer un diagnóstico del estado actual de los proyectos que se desarrollan en el área temática Sistema de Apoyo a la Salud fue necesario aplicar una encuesta para captar información cualitativa y cuantitativa sobre los conocimientos que se poseen de las pruebas de software. En el *Anexo 2* se puede ver el modelo de la encuesta aplicada.

Las preguntas de la encuesta fueron diseñadas para conocer el grado de conocimiento del personal involucrado en los proyectos, en cuanto a las pruebas de software y el flujo de trabajo de pruebas de software. Además saber si las mismas son aplicadas en los proyectos productivos del área temática, así como tener conocimiento de la metodología que es usada y si existen roles definidos para aplicar las mismas.

En la elaboración de la encuesta se combinaron los dos tipos de preguntas fundamentales, abiertas y cerradas. Las preguntas cerradas se hicieron con el interés de conocer la información cuantitativa con respecto al tema y las preguntas abiertas con el objetivo de saber la opinión del encuestado.

Las encuestas se realizaron en los 6 proyectos productivos de los cuales hay 4 que se dedican al desarrollo de software de gestión de salud y 1 especializado en la atención remota a servidores. Se realizaron un total de 53 encuestas; 6 a estudiantes que pertenecen al proyecto de Estadísticas, 9 al de Docencia Médica, 4 a Balance Material, 4 a Colaboración Médica, 2 a Atención Remota a Servidores ,1 a la Informatización del Portal de la Facultad 7 y 27 estudiantes que no pertenece a ningún proyecto pero trabajan en tareas relacionadas directamente con el área temática, dentro de este grupo de estudiantes se encuentra el equipo de calidad del área temática.

**Análisis de cada pregunta de la encuesta**

**Pregunta 1**

La pregunta arrojó, que de las 54 personas encuestadas, el 3.7% son estudiantes de segundo año, el 14.8% son de tercer año, el 9.3% son de cuarto año y un 72.2% son de quinto año. Estos resultados indican, que en el Área Temática de SAS, el mayor por ciento de los estudiantes que trabajan en los proyectos, pertenecen a quinto año, es por eso, que la mayor parte de los encuestados son del último año de la carrera.

**Pregunta 3**

El 90.7% conoce sobre qué son las pruebas de software y solo un 9.3% tienen desconocimiento sobre el tema. Del 90.7% que especifican que conocen sobre pruebas de software un 77.7% ejemplifica con los métodos de Caja Negra y Caja Blanca y un 12.9% conocen pruebas de Seguridad, Usabilidad, Configuración, Rendimiento, Integración, Stress. El 50% de los 54 estudiantes encuestados, respondieron que en sus proyectos productivos si se aplican pruebas de software y el otro 50% respondió, que no conocían si en sus proyectos se hacían pruebas.

Del 50% de los estudiantes que manifestaron que en sus proyectos se hacen pruebas, solo un 16% conocen las pruebas que se realizan y el resto de los estudiantes saben que se aplican pruebas, pero no conocen cuales son. Por lo que se puede concluir, que la mayoría de los estudiantes conocen sobre las pruebas de software, pero solo un 50% respondió que utilizan las pruebas.

Por otra parte, se demuestra, que a pesar de que muchos estudiantes de los encuestados se refieren a que en sus proyectos se hacen pruebas, no mencionan los tipos, métodos o técnicas de pruebas que se utilizan. La evaluación de esta pregunta, arroja que en los proyectos se conoce sobre las pruebas, algunos aplican pruebas exploratorias sin tener en cuenta algún proceso organizado, pero no existe ningún proceso de pruebas que sirva de patrón, para guiar las pruebas en los proyectos de SAS.

**Pregunta 4**

Para la evaluación de esta pregunta, solo se tomó la muestra de los estudiantes que respondieron que se aplican pruebas en sus proyectos. Del 50% de los estudiantes que respondieron que se aplican pruebas, solo un 11.1% reconocieron las fases de desarrollo en las que se hacían pruebas y ejemplificaron los tipos de pruebas en cada fase y el otro 38.9%, saben que se realizan pruebas en las fases, pero no conocen las pruebas que se realizan en cada fase de desarrollo del proyecto. Esto arroja un resultado un poco contradictorio, porque lo estudiantes conocen que se realizan pruebas en el proyecto, las fases en que se realizan y no conoce los tipos de pruebas que se aplican.

**Pregunta 5**

El 51.9% conoce acerca del Flujo de Trabajo de Pruebas y el 48.1% respondió que no conoce el proceso. Esto indica, que la mayoría de los encuestados no le conciernen mucha importancia al flujo de trabajo de pruebas, por lo que es importante, realizar un proceso de capacitación a los estudiantes del Área Temática acerca del tema, para que todos los estudiantes conozcan las actividades de pruebas que se debe realizar, ya que es de mucha importancia para contribuir a la calidad del software, en todo su ciclo de vida.

**Pregunta 6**

El 37% respondió, que en sus proyectos existen roles definidos para realizar las pruebas de software y el 63% de los estudiantes encuestados, tienen desconocimiento respecto al tema. En esta pregunta se puede apreciar la desinformación respecto a los roles para las pruebas y la necesidad de asignar roles para planificar todo el proceso de pruebas. Los que respondieron que existen roles dedicados a las pruebas, no coinciden las respuestas de estudiantes del mismo proyecto. Con la organización de un grupo de estudiantes y profesores dedicados a este tema, se puede lograr una mejor organización en todos los proyectos.

**Pregunta 7**

El 53.7%, conoce respecto a los artefactos que se deben tener en cuenta para realizar las pruebas y un 46.3, desconocen sobre los artefactos involucrados con el proceso de pruebas. Esta pregunta es muy importante, ya que refleja la falta de conocimiento sobre un tema tan importante, como son los artefactos de pruebas, porque a través de estos, se puede evaluar el software.

**Pregunta 8:**

De toda la muestra encuestada, el 96.3% está de acuerdo con que exista un procedimiento que guíe las pruebas de software, en aras de contribuir a la calidad de los productos que se implementan en SAS y el 3.7% tiene desconocimiento sobre el tema, ya que existen estudiantes en los proyecto, que son de cursos donde aún no se ha impartido el tema sobre pruebas. Es muy importante, que la mayoría coincida con la idea de crear un procedimiento de pruebas, que sirva de modelo para todos los proyectos.

### 2.5.3 Análisis de los resultados obtenidos

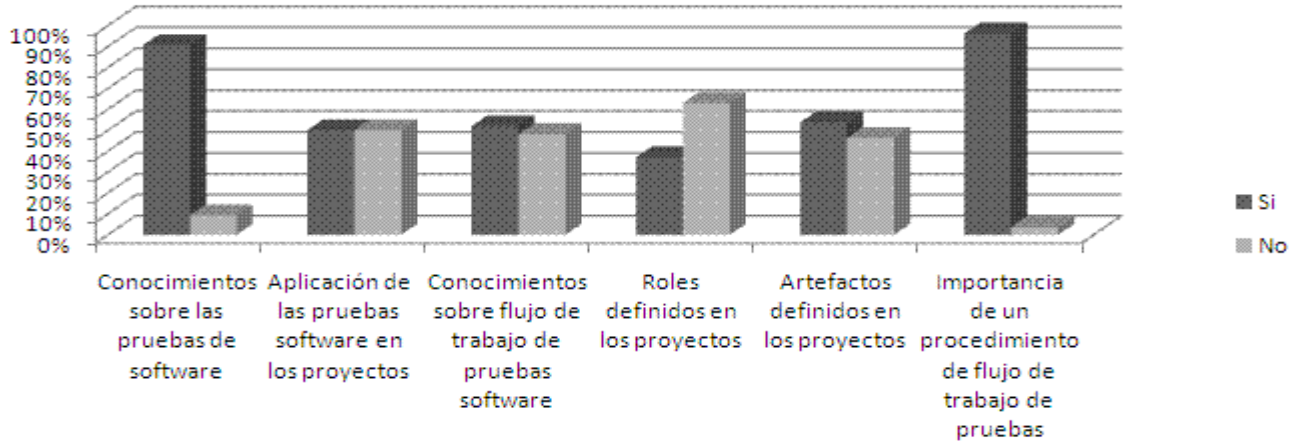



**Figura 2.11.Resultados obtenidos en el area tematica SAS**

A partir del análisis de cada una de las preguntas realizadas en la encuesta se evidencia que los estudiantes del área temática Sistema de Apoyo a la Salud poseen conocimientos sobre las pruebas de software, esencialmente de Caja Blanca y Caja Negra. A pesar de esto no en todos los proyectos se aplican las mismas en las diferentes etapas por la que transita el software que se esta implementando y no se tiene un conocimiento conciso de lo que es un flujo de trabajo de pruebas software, así como los roles y artefactos que de deben ser definidos. Sin embargo casi todos los estudiantes afirman que consideran importante la puesta en práctica de un procedimiento de flujo de

trabajo de pruebas para los proyectos que se desarrollan en el área temática para contribuir a la calidad de los mismos.

## 2.6 Conclusiones

Una vez analizados los temas abordados durante el capítulo, se han llegado a las siguientes conclusiones

- ✓ Las principales técnicas empleadas para realizar la investigación de este trabajo fueron: la entrevista y la encuesta.
- ✓ Se evidenció a través de una encuesta cómo los estudiantes de la Universidad de las Ciencias Informáticas y específicamente del área temática SAS poseen conocimientos sobre las pruebas de software y lo poco que son aplicadas en los productos software que se desarrollan en los proyectos productivos.
- ✓ Se evidenció que en muchos de los proyectos productivos de la UCI no se ponen en práctica procedimientos de pruebas que contribuyan a un mejor desarrollo de los productos software con la calidad requerida.
- ✓ Se demostró la necesidad de la puesta en práctica de un procedimiento de pruebas que contribuya a la calidad de los productos software que se desarrollan en el área temática Sistema de Apoyo a la Salud.

# Capítulo 3: Descripción de la solución propuesta

## 3.1 Introducción

Este capítulo surge a partir de un estudio del flujo de trabajo de pruebas que propone la metodología de desarrollo de software RUP, con la utilización de algunos artefactos y roles presentados en la Ayuda del Rational 2003 y la combinación de las buenas prácticas utilizadas por el Proceso Software Personal y el Proceso Software en Equipo. Se hizo énfasis en el Modelo de Pruebas W, ya que el mismo plantea una serie de pasos necesarios para la realización de las pruebas.

El objetivo de este capítulo es lograr una coherente vinculación entre los temas estudiados, para obtener un procedimiento de flujo de trabajo de pruebas que sirva de guía para realizar las mismas durante el ciclo de vida del software, ajustada a los proyectos productivos del área temática Sistemas de Apoyo a la Salud (SAS).

En esta sección se propone un procedimiento de pruebas que cumpla con las características requeridas para asegurar, en el mayor porcentaje posible, la ausencia de defectos en los productos que se liberan hacia el laboratorio de calidad de la facultad.

## 3.2 Propósito

Contribuir a la calidad de los productos software durante su ciclo de vida**.**

## 3.3 Alcance

Definir un flujo de trabajo de pruebas durante el ciclo de vida de los productos software del área temática Sistema de Apoyo a la Salud.

## 3.4 Acrónimos y Abreviaturas

**SAS**: Sistema de Apoyo a la Salud.
**EP**: Equipo de Pruebas.
**PSP**: Proceso Software Personal.
**TSP**: Proceso Software en Equipo.

## 3.5 Descripción general del procedimiento


**Figura 3.1. Procedimiento de pruebas para el área temática SAS**

Este procedimiento está dirigido a contribuir a la calidad de los productos software del área temática Sistema de Apoyo a la Salud de la Facultad 7 y está basado en la metodología RUP.

Para el cumplimiento de un proceso de pruebas de forma satisfactoria, se propone un flujo de trabajo de pruebas que se realice en paralelo con el proceso de desarrollo de los productos software que se implementan en el área temática SAS. Para llevar a cabo dicho flujo de trabajo, se propone la conformación de un Equipo de Pruebas (EP), compuesto por diversos roles con sus funciones definidas.

Las pruebas a realizar están organizadas en 8 períodos las cuales se realizan en dependencia de la fase (Inicio, Elaboración, Construcción y Transición) por la que esté transitando el proyecto. Estos períodos definen las pruebas a realizar, tanto a la documentación como al software, durante el ciclo de vida de los mismos y especifican en qué fase del proceso de desarrollo del proyecto deben ser aplicadas. Para la realización de cada una de las pruebas se especifican 4 actividades fundamentales y se generan artefactos de entrada y salida.

## 3.6 Propuesta del procedimiento

### 3.6.1 Objetivos del flujo de trabajo propuesto

- ✓ Definir roles y sus responsabilidades para el proceso de pruebas.
- ✓ Garantizar las revisiones a la documentación del proyecto y al software.
- ✓ Planificar las pruebas necesarias en cada fase del ciclo de vida del software, partiendo de los caso de uso del sistema.
- ✓ Garantizar que a los productos software del área temática SAS, se le hayan realizado las pruebas mínimas requeridas antes de ser liberados.

### 3.6.2 Estructura organizativa

Se propone que dentro del Equipo de Calidad del área temática SAS se conforme un Equipo de Pruebas (EP), el cual tiene como misión planificar y ejecutar todo el proceso de pruebas desde la etapa de inicio del software, hasta la entrega al cliente. Los integrantes de este equipo no deben estar directamente involucrados en el desarrollo del producto, pues así se detecta con mayor facilidad los posibles errores que puedan afectar la calidad del software.

Para definir los roles que integran el EP, se han escogidos algunos de los propuestos por TSP y por el flujo de trabajo de pruebas de la metodología RUP; como bibliografía, la Ayuda del Rational 2003, por ser la misma la más actualizada a la que se tiene acceso.

Se propone que la conformación de este EP se realice a través de un proceso de inscripción, en el cual, las personas interesadas en formar parte del mismo deben llenar una planilla de solicitud y presentarse a un examen. Luego de esto, se realiza un análisis de los resultados de dicho examen, con el objetivo de definir quienes formarán parte del EP. El Líder del Equipo de Prueba hace la distribución de roles dentro del grupo de acuerdo con las habilidades que posea cada uno.

**Roles definidos**

Los roles que se definen para integrar el EP, son dos de los que propone TSP y tres del flujo de trabajo de pruebas de la Ayuda del Rational 2003. Los mismos son: Líder del Equipo, Líder de Planificación, Analista de pruebas, Diseñador de Pruebas y Probador. Cada rol de este equipo se selecciona de acuerdo a las habilidades y conocimientos que posee el individuo sobre las pruebas de software, así como su modo de aplicarlas.

Se propone que exista un Líder del Equipo de Pruebas para toda el área temática y al menos un Líder de Planificación de pruebas en cada proyecto, el mismo será el encargado de dirigir el proceso de pruebas en su proyecto.

El Líder del Equipo de Pruebas es el encargado de determinar si deben existir más personas desempeñando el mismo rol, en dependencia de la complejidad del proyecto, además es importante destacar que como mínimo debe existir un individuo por rol en cada proyecto.

A pesar de realizar un proceso de selección para el desempeño de cada rol se plantea a continuación las habilidades que debe tener creadas el individuo para realizar sus funciones.

**Tabla 3.1. Roles y habilidades**

| **Roles** | **Año docente y habilidades necesarias a tener en cuenta para ejercer el rol.** |
|---|---|
| Líder del Equipo de pruebas | -Debe ser un especialista en el tema de pruebas. Y poseer experiencias prácticas de cómo liderar y llevar a cabo un proceso de pruebas. |

| | |
|---|---|
| Líder de Planificación de las pruebas | -Poseer conocimientos de alguna herramienta para la planificación y control de tareas.<br>-Tener conocimientos sólidos sobre pruebas e ingeniería de software. Se recomienda que sea un estudiante de cuarto o quinto año de la carrera. |
| Analista de Pruebas | -Tener conocimientos sólidos sobre pruebas e ingeniería de software. --<br>-Pueden ser estudiantes de tercer año a partir del segundo semestre si están bien capacitados, pero se recomienda que esta función sea desempeñada por estudiantes de cuarto y quinto año de la carrera. |
| Diseñador de Pruebas | -Tener conocimientos sólidos sobre pruebas e ingeniería de software. --<br>-Poseer habilidades en diferentes lenguajes de programación.<br>-Tener conocimientos sobre como diseñar casos de pruebas para realizar las pruebas del software.<br>-Pueden ser estudiantes de tercer año a partir del segundo semestre si están bien capacitados, pero se recomienda que esta función sea desempeñada por estudiantes de cuarto y quinto año de la carrera. |
| Probador | -Tener conocimientos básicos sobre las pruebas.<br>-Pueden ser estudiantes de segundo año en adelante. |

**Líder del Equipo de Pruebas**

El Líder del Equipo de pruebas debe lograr que su equipo funcione de forma efectiva, brindando la posibilidad de dar soluciones a las inquietudes que se presenten durante el proceso de pruebas de los proyectos del área temática. Controla el cumplimiento de las normativas para la regularización del proceso de pruebas. Determina, basándose en análisis realizados sobre los resultados obtenidos, cuándo se puede liberar el software para que le realicen las pruebas en el laboratorio de Calidad de la Facultad 7.

Responsabilidades

- ✓ Dirigir el proceso de selección de los integrantes del EP de cada proyecto y asegurar que cada uno de los roles esté desempeñado por una persona.
- ✓ Establecer un Plan de Trabajo para los proyectos del área temática SAS, con las actividades y fechas de entregas establecidas de forma general.
- ✓ Definir el Plan de Pruebas General del área temática.

- ✓ Controlar el cumplimiento de los planes de trabajo establecidos por el Líder de Planificación de cada proyecto.
- ✓ Revisar el cumplimiento del proceso de pruebas en cada proyecto.
- ✓ Configurar el ambiente de prueba para cada proyecto junto al Analista y el Diseñador.
- ✓ Dirigir el equipo encargado de liberar los productos del área temática al laboratorio de Calidad de la Facultad 7.
- ✓ Realizar la solicitud de cambios de los defectos encontrados al Líder del Equipo de Desarrollo.

**Líder de Planificación de las pruebas**

El Líder de Planificación de las pruebas es el encargado de planear todas las actividades que se realizan durante el proceso de pruebas de los proyectos de SAS, insertarlas dentro del cronograma de trabajo y mantener la información actualizada, para poder brindar los resultados de las pruebas a partir de las no conformidades detectadas por el EP.

Responsabilidades

- ✓ Definir y revisar el Plan de Trabajo de las tareas de cada integrante del EP en el proyecto.
- ✓ Organizar el proceso de pruebas del proyecto.
- ✓ Planificar en qué momento se realizan las pruebas en el proyecto.
- ✓ Controlar el cumplimiento de las tareas planificadas para cada persona en el proyecto.
- ✓ Identificar recursos necesarios para la realización de las pruebas y el alcance de las mismas.
- ✓ Supervisar que los artefactos generados como resultado de las pruebas cuenten con la calidad requerida.
- ✓ Evaluar junto al Analista del EP si la prueba realizada alcanza los objetivos y metas descritos en el Plan de Pruebas.
- ✓ Identificar las personas que están involucradas en el proceso de planificación de cada una de las pruebas.
- ✓ Participar en la liberación de los productos del área temática al laboratorio de Calidad de la facultad.

**Analista de pruebas**

El Analista es el encargado de identificar las pruebas a realizar en cada fase por la que transite el proyecto y detallar las actividades pertinentes para que las mismas tengan un resultado exitoso.

Responsabilidades

- ✓ Realizar un estudio de los objetivos y reglas del proyecto, para definir el alcance de las pruebas a efectuar.

- ✓ Identificar los aspectos fundamentales que se van a probar.
- ✓ Comprobar si los casos de uso del sistema son los adecuados, teniendo en cuenta los requisitos especificados.
- ✓ Definir y elaborar el Plan de Pruebas en dependencia del período en que se encuentre el proyecto.
- ✓ Identificar las pruebas necesarias.
- ✓ Identificar el propósito de la prueba.
- ✓ Definir revisiones a la documentación.
- ✓ Definir el enfoque de las pruebas en próximas iteraciones.
- ✓ Revisar la documentación mínima necesaria para entregar al laboratorio de Calidad de la Facultad 7.
- ✓ Evaluar los resultados de la pruebas en cada fase del proyecto.
- ✓ Monitorear el progreso de las pruebas.
- ✓ Evaluar junto al Líder de Planificación si la prueba realizada alcanza los objetivos y metas descritos en el Plan de Pruebas.
- ✓ Participar en la liberación de los productos del área temática al laboratorio de Calidad de la facultad.
- ✓ Configurar el ambiente de pruebas para el proyecto junto al Líder del Equipo, el Líder de Planificación y el Diseñador.
- ✓ Conformar el Informe de Solicitudes de Cambios.

**Diseñador de pruebas**

Es el responsable de verificar y diseñar las pruebas identificadas por el Analista, y en dependencia de estos tipos de pruebas, se las asigna a los probadores para realizar las mismas.

Responsabilidades

- ✓ Verificar el enfoque de la prueba.
- ✓ Definir la estrategia de prueba a utilizar en el proyecto.
- ✓ Identificar el método más adecuado para un tipo de prueba en específico.
- ✓ Planificar el orden para realizar las pruebas.
- ✓ Configurar el ambiente de prueba para el proyecto junto al líder del equipo, el Líder de Planificación y el Analista de pruebas.
- ✓ Diseñar las pruebas identificadas por el Analista de pruebas.
- ✓ Diseñar los casos de prueba a partir de los casos de uso del sistema.
- ✓ Evaluar la efectividad de las técnicas de pruebas utilizadas.

**Probador**

Es el responsable de ejecutar las pruebas que le son entregadas por el diseñador. Debe registrar todos los errores detectados durante el proceso en la plantilla propuesta para esta actividad y además realizar una evaluación final de la prueba efectuada.

Responsabilidades

- ✓ Ejecutar las pruebas diseñadas.
- ✓ Analizar y registrar los errores encontrados en el Informe de No Conformidades.
- ✓ Evaluar las pruebas efectuadas y registrar los resultados en el Documento de Evaluación de Pruebas.
- ✓ Entregar los resultados de las pruebas al Analista.

Al concluir el proceso de pruebas de los productos desarrollados en el área temática el Líder de Planificación, el Analista de pruebas y el Líder del Proyecto se reúnen para evaluar y definir si dicho producto reúne las condiciones requeridas para ser liberado. Como resultado de este proceso se conforma el Acta de Liberación del Producto Software del área temática. *Anexo 3*

**Actividades fundamentales del Equipo de pruebas**

**Seleccionar el equipo de pruebas**

Un factor de vital importancia para aplicar un procedimiento satisfactorio y en el tiempo requerido, es la selección del personal adecuado, por tanto en el proceso de captación se hace necesario elegir a las personas más capacitadas para desempeñar cada rol a partir de los conocimientos, habilidades y aptitudes que el mismo exige.

Para la selección del EP se captarán los interesados mediante solicitudes voluntarias, el cual se formaliza mediante una planilla en la cual se archivan los datos necesarios del individuo. En el proceso de solicitud, la persona escoge el proyecto al que desea pertenecer y el rol que le interesa ocupar. Luego, es agregado a la lista de candidatos dicho rol dentro del proyecto, se realiza una evaluación mediante un examen para medir sus conocimientos de acuerdo al rol que desea. Por último se realiza la asignación de roles de acuerdo a la evaluación realizada. Ver planilla de Solicitud de Ingreso. *Anexo 4*

**Capacitar al Equipo de pruebas**

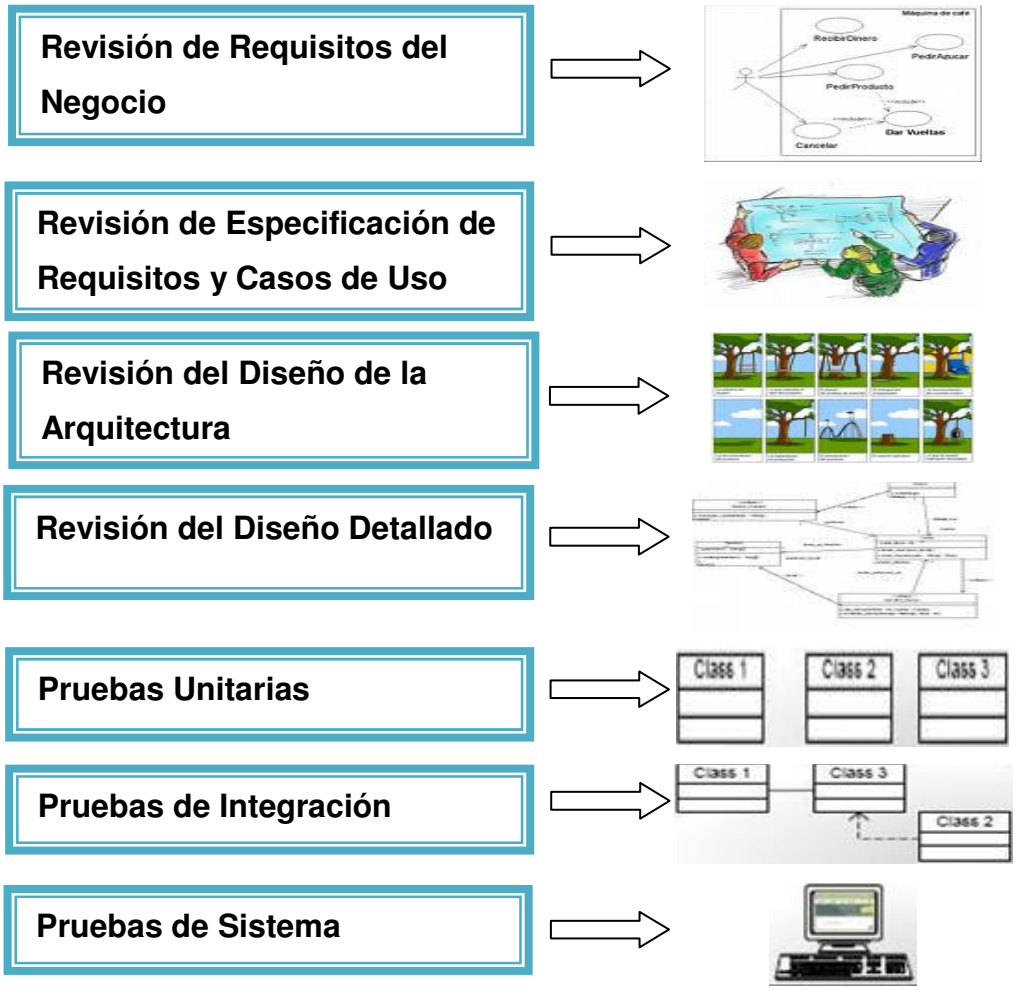
Se ha evidenciado que para lograr desarrollar un proceso de pruebas con la suficiente calidad y competitividad, es muy importante contar con un personal altamente calificado y motivado. Por tanto,

es de vital importancia preparar al personal que integra el EP, posibilitando con esto que se incremente la eficiencia en el trabajo. Para lograr este objetivo se proponen las siguientes estrategias:

- ✓ Buscar el personal capacitado para impartir cursos relacionados con las pruebas de software.
- ✓ Realizar cada 15 días encuentros con especialistas del tema, donde se puedan exponer inquietudes sobre una determinada idea a implementar.
- ✓ Realizar estudios de posibles combinaciones entre métodos, técnicas y metodologías que formulen mejores estrategias de pruebas.

### 3.6.3 Períodos de pruebas

En el flujo de trabajo propuesto se han definido 8 períodos de pruebas, los 4 primeros dedicados a la revisión de la documentación del proyecto y los restantes períodos dirigidos a las pruebas de software los cuales se realizan en paralelo con las fases que propone la metodología RUP.



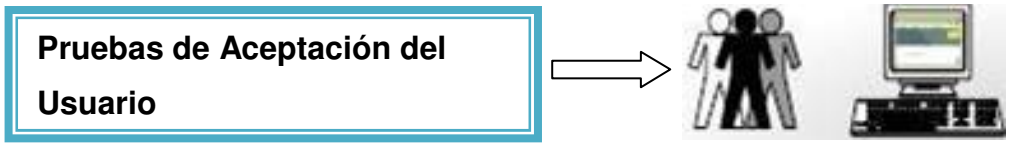



**Figura 3.2. Períodos de pruebas propuestos**

**Descripción de los períodos de pruebas**

**1- Revisión de Requisitos de Negocio**: Este período enmarca el análisis a los Modelos del Negocio. Las revisiones están encaminadas a comprobar si los procesos del negocio están correctamente identificados.

**2- Revisión de Especificación de Requisitos y Casos de Uso**: Este período está encaminado a revisar el Documento de Especificación de Requisitos y el Modelo de Caso de Uso. Las revisiones se realizan para verificar si los requisitos definidos se ajustan a los especificados por el cliente.

**3- Revisión del Diseño de la arquitectura**: Este período centra su mayor atención en la arquitectura definida para el software. Las revisiones están encaminadas a la verificación de la arquitectura teniendo en cuenta el Documento de Arquitectura y la Plantilla de Ambiente de Desarrollo del proyecto. Este período tiene gran relevancia, ya que una buena revisión de la arquitectura, posibilita que las funcionalidades obtenidas sean las esperadas por el cliente.

**4- Revisión del Diseño detallado**: Este período centra su mayor atención en el diseño de la interfaz visual del sistema, y en la verificación detallada del Modelo de Análisis y de Diseño. Las revisiones están encauzadas a verificar que la parte visual del sistema cumpla con las características definidas por el cliente y con los estándares de diseño propuestos para las aplicaciones desarrolladas en el área temática SAS, relacionados con el sector de salud**.**

**5- Pruebas Unitarias**: Luego de haber sido implementados los diferentes componentes del sistema, se puede pasar a ejecutar el período de Pruebas Unitarias, el cual tiene como propósito probar la lógica interna del código.

**6- Pruebas de Integración**: Este período enmarca su atención en la integración de los componentes del proyecto, por tanto, las pruebas están encaminadas a verificar que el sistema funciona correctamente una vez integrado.

**7- Pruebas de Sistema**: Este período centra su mayor atención en realizar pruebas al sistema, las cuales están dirigidas de forma general a comprobar si el mismo cumple con las propiedades o cualidades definidas para su funcionamiento.

**8- Pruebas de Aceptación del Usuario**: Las pruebas en este período son realizadas por el usuario en su entorno de trabajo, las mismas tienen el objetivo de verificar que el sistema funcione correctamente y cumpla con los requisitos especificados por el usuario.

### 3.6.4 Panorama de las actividades de pruebas en las fases del proyecto.

**Fase de Inicio**

En esta fase comienza el procedimiento de pruebas que se propone y se definen las premisas para que éste se implemente con éxito, teniendo en cuenta las primeras precisiones presentadas por el Equipo de Desarrollo. Cuando el proyecto esté transitando por esta fase, los períodos de pruebas a realizar son el de Revisión de Requisitos de Negocio y posteriormente el de Revisión de la Especificación de Requisitos y Casos de Uso. Teniendo en cuenta lo que propone la metodología RUP para la fase inicial de un software, se plantea que en esta fase del proyecto, las actividades de pruebas estén dirigidas a verificar si se han identificado de forma correcta:

- ✓ Casos de Uso (Negocio y Sistema)
- ✓ Actores que definen el ámbito (Negocio y Sistema)
- ✓ Descripción de los caso de uso (Negocio y Sistema)
- ✓ Casos de uso más críticos (es importante para obtener una arquitectura candidata correcta).

Posteriormente a esto se realizan revisiones al modelo de casos de uso de negocio y del sistema, teniendo en cuenta que exista una profunda descripción de cada caso de uso y concordancia con los requisitos, tanto funcionales como no funcionales impuestos por el cliente.

**Fase de Elaboración**

Para esta fase del proyecto el procedimiento propone verificar si la arquitectura definida y el diseño propuesto son correctos. Cuando el proyecto está transitando por esta fase, las actividades de pruebas que se realizan corresponden a los períodos de Revisión del Diseño de la Arquitectura y a la Revisión del Diseño Detallado. Teniendo en cuenta lo que propone la metodología de RUP para la fase de Elaboración de un software, se plantea, que en el período de Revisión de la Arquitectura se comprueben si se ha identificado, analizado, diseñado y realizado de forma correcta la descripción de

todos los casos de uso significativos desde el punto de vista de la arquitectura, para poder identificar la prioridad de cada caso de uso. Además se verifica que la arquitectura cumpla con los requisitos especificados en la plantilla de Ambiente de Desarrollo definida para los proyectos de SAS. En el período de Revisión del Diseño Detallado se verifica, que el diseño visual del sistema esté correctamente diseñado, comprobando si el prototipo no funcional cumple con las características requeridas.

**Fase de Construcción**

En esta fase luego del resultado de la implementación, el procedimiento propone verificar si el producto software en su versión operativa inicial, está apto para ser usado por el cliente. Cuando el proyecto está transitando por la fase de Construcción, las actividades de pruebas que se realizan corresponden a los períodos de Pruebas Unitarias y Pruebas de Integración, en estos períodos el objetivo fundamental es probar la parte funcional de software.

**Fase de Transición**

Cuando el proyecto está transitando por la fase de Transición, se obtiene el producto listo para la entrega al cliente. Las actividades de pruebas que se realizan en la misma, corresponden a los períodos de Pruebas de Sistema y Pruebas de Aceptación del Usuario. El período de Pruebas de Sistema en esta fase, tiene como principal objetivo, verificar si el software está listo antes de la entrega al cliente y el período de Pruebas de Aceptación del Usuario, está encaminado a corregir los problemas encontrados durante las pruebas en el entorno del usuario. La esencia de estos períodos en la Fase de Transición, es detectar y corregir los defectos, constituyendo las pruebas de regresión un elemento importante, para asegurar que estas modificaciones no introduzcan nuevos defectos.

### 3.6.5 Actividades para cada proceso de prueba

Para la realización de cada una de las pruebas que se definen en este flujo de trabajo de pruebas es importante que se efectúen en cada una de ellas las siguientes actividades

- ✓ Planificar la prueba.
- ✓ Diseñar la prueba.
- ✓ Evaluar y probar la prueba.
- ✓ Evaluar la calidad de la prueba.

**ACTIVIDADES DE LA PRUEBA**

**Planificar la prueba** →
- Identificar el personal involucrado.
- Definir la estrategia de prueba a utilizar.
- Identificar el propósito de la prueba.
- Identificar el método de prueba más adecuado.
- Identificar recursos necesarios y alcance.
- Especificar los requerimientos que se probaran.

**Diseñar la prueba** →
- Determinar los casos de pruebas a desarrollar durante el proceso planificado
- Diseñar los casos de pruebas determinados.
- Verificar el enfoque de la prueba.
- Configurar el ambiente de prueba.

**Probar y evaluar** →
- Ejecutar las pruebas diseñadas.
- Ejecutar suite de pruebas.
- Evaluar el producto.
- Determinar resultados de la prueba.
- Analizar fallos.
- Revisar artefactos resultantes**.**

**Evaluar la calidad de las pruebas** →
- Evaluar los resultados de las pruebas
- Evaluar si las pruebas realizadas alcanzan los objetivos y metas
- Realizar solicitud de cambio
- Definir enfoque en pruebas posteriores
- Definir elementos a probar
- Evaluar la efectividad de las técnicas de pruebas utilizadas

**Planificar la prueba**

La tarea fundamental que se realiza en esta actividad es planificar el proceso de la prueba que se va a realizar, que pueden ser pruebas a la documentación y pruebas al software directamente. En esta planificación se determinan las personas involucradas en el proceso, teniendo en cuenta el propósito principal que va a tener la prueba, se define la estrategia que se va a seguir, el orden las actividades necesarias para realizar la prueba y el tiempo dedicado para su ejecución. Es importante especificar el alcance que va a tener la prueba, para poder definir los recursos que sean necesarios en el transcurso de la prueba, especificando los requerimientos que se probaran.

Descripción de las actividades

**Identificar el personal involucrado**: El Líder de Planificación de las pruebas debe identificar todas las personas involucradas a lo largo de todo el proceso de la prueba a realizar.

**Identificar el propósito de la prueba**: En esta actividad el Analista de pruebas identifica los elementos que serán probados, para tener una visión general, de cuál es el objetivo principal de la prueba.

**Definir la estrategia de prueba a utilizar**: El Analista de pruebas es el encargado de estudiar las necesidades de pruebas del proyecto, para identificar todos los puntos que tiene que tener la estrategia propuesta en el Plan de Pruebas General, para cumplir con un excelente proceso de pruebas. Aquí se identifica el método de prueba, las técnicas de pruebas y los niveles por los que pasa la prueba.

**Identificar recursos necesarios y alcance**: En esta actividad el Líder de Planificación de las pruebas define el alcance que va a tener la prueba a realizar, las limitaciones pertinentes y los recursos que sean necesarios, esto permite tener una visión general del proceso de pruebas.

**Especificar los requerimientos que se probaran**: El Analista de pruebas describe detalladamente los requisitos que se van a probar, para que en el momento de hacer el diseño de la prueba esté delimitado lo que se quiere probar. *Anexo 5*

**Diseñar la prueba**

La tarea fundamental que se realiza en esta actividad es diseñar las pruebas que guiarán el proceso planificado, a partir de la revisión a la documentación, del diseño de los casos de pruebas y la

verificación al sistema; documentando la secuencia de pasos para ejecutar cada caso de prueba y los resultados esperados; para esto es necesario verificar el enfoque de la prueba seleccionada en la estrategia de prueba definida. El diseño de los casos de pruebas se realiza a partir de los casos de uso del sistema, comenzando con los definidos como más críticos, por tanto, es importante tener una buena descripción de los mismos.

Descripción de las Actividades

**Verificar el enfoque de la prueba:** El Diseñador de pruebas comprueba que los métodos y las técnicas propuestas sean las más adecuadas, para facilitar el trabajo del diseño de los casos de pruebas, además revisa el Plan de Pruebas (en dependencia del período en que se encuentre) y verifica que las herramientas seleccionadas sean apropiadas, teniendo en cuenta los recursos disponibles.

**Determinar los casos de pruebas a desarrollar durante el proceso planificado**: En esta actividad el Diseñador de pruebas tiene en cuenta los casos de uso, los requerimientos a probar previamente seleccionados y las técnicas de pruebas elegidas, para determinar los casos de pruebas a ejecutar.

**Diseñar los casos de pruebas determinados**: El propósito de esta actividad es, obtener un conjunto de casos de pruebas que tengan la mayor probabilidad de descubrir los defectos del software, estos casos de pruebas son diseñados por el Diseñador de pruebas a partir de los casos de uso de sistema.

**Pasos para el diseño de un caso de prueba a partir de los casos de uso del sistema**

**1-** Descripción general (Se realiza la descripción de todo el proyecto, para tener una visión general de cómo funciona cada proceso).

**2-** Describir los objetivos generales (Descripción específica de cada caso de uso, incluyendo todos los flujos alternos que puedan existir).

**3-** Elaborar las condiciones que tienen que existir, para que se pueda ejecutar el caso de uso.

**4-** Elaborar una tabla con la especificación de todas las clases válidas y las clases inválidas que tenga el caso de uso, también se representa el resultado esperado de la prueba, el resultado real de la prueba y las observaciones encontradas.

Se ha diseñado una plantilla donde se recogen todos estos datos**.** *Anexo 6*

**Configurar el ambiente de prueba:** Líder del Equipo de pruebas, el Analista y el Diseñador definen los elementos de hardware y software que servirán de soporte al proceso de pruebas, teniendo en cuenta las pruebas planificadas, de documentación y principalmente pruebas directamente al software.

**Probar y evaluar**

En esta actividad se ejecutan las pruebas diseñadas, con el fin de hacer una evaluación de los requerimientos a probar definidos anteriormente y determinar, si estos han sido cumplidos. Se archivan todos los resultados de la prueba, para facilitar la evaluación en curso del producto. Se hace un resumen de todos los fallos encontrados, esto ayuda a corregir los errores que puedan aparecer posteriormente. En esta actividad se realiza una evaluación de todas las técnicas propuestas para probar el software desechando las que no hayan sido satisfactoriamente usadas. Además también se verifican los artefactos resultantes.

Descripción de las Actividades

**Ejecutar las pruebas diseñadas:** En esta actividad el Probador ejecuta la prueba previamente diseñada, en un ambiente de pruebas disponible, ya sea manualmente o a través de una herramienta, según esté especificado en el diseño del caso de prueba.

**Evaluar el producto**: En esta actividad el Probador, evalúa el producto resultante haciendo uso del manual de usuario.

**Determinar resultados de la prueba**: En esta actividad el Probador hace un resumen de las evaluaciones de la calidad del producto, identifica y captura los resultados de prueba detalladamente y propone las acciones apropiadas para corregir fallos en la calidad. *Anexo 7*

**Analizar fallos**: En esta actividad el Analista de pruebas investiga los detalles de los registros de pruebas, y analiza los fallos que ocurrieron durante la implementación y ejecución de las mismas. Estos errores son registrados en el Informe de No conformidades por el Probador. *Anexo 8*

**Revisar artefactos resultantes:** El Analista de pruebas revisa los artefactos que se obtienen durante la ejecución de todas las actividades.

**Evaluar la calidad de las pruebas**

La función fundamental de esta actividad es mantener y mejorar la calidad de las pruebas. Esto es especialmente importante si la intención es reutilizar las ventajas desarrolladas en el actual ciclo de prueba y en ciclos de pruebas posteriores. Esta actividad está orientada a mostrar, a los involucrados en el proceso de pruebas, si los resultados del proceso cumplen con los objetivos y las metas trazadas en el Plan de Pruebas, de manera que, finalmente se hayan cumplido los requerimientos del cliente, de lo contrario, se define una nueva iteración para realizar pruebas.

Descripción de las Actividades

**Evaluar los resultados de las pruebas**: El Analista de pruebas examina todos los errores y fallos encontrados, durante la ejecución de las pruebas, y analiza el estado de los mismos, evaluando si ya han sido solucionados, o si hay que darle seguimiento.

**Evaluar si las pruebas realizadas alcanzan los objetivos y metas descritos en el Plan de Pruebas**: El Líder del Equipo de pruebas y Analista de prueba determinan si la prueba ha alcanzado plenamente las metas y objetivos trazados inicialmente en el Plan de Pruebas, y si se culmina el ciclo de pruebas. *Anexo 9*

**Realizar solicitud de cambio**: El Líder del Equipo de pruebas, sobre la base de las metas y los objetivos no alcanzados en las pruebas realizadas en el período de Pruebas de Aceptación del Usuario, solicita el tratamiento a errores no solucionados. *Anexo 10*

**Definir enfoque para pruebas posteriores**: El Analista de pruebas realiza un análisis a partir de los resultados obtenidos y establece los objetivos para la próxima iteración.

**Definir elementos a probar**: El Analista de pruebas define los elementos que necesitan ser probados nuevamente.

**Evaluar la efectividad de las técnicas de pruebas utilizadas**: El Diseñador de pruebas evalúa en que medida las técnicas de pruebas seleccionadas han cumplido con las metas trazadas en el Plan de Pruebas.

### 3.6.6 Artefactos definidos

Cuando se trabaja en equipo, la comunicación entre los integrantes del mismo es muy importante. Una de las vías que se ha implantado para lograrlo es a través de documentos en los cuales se guarda toda la información necesaria y detallada que pueda necesitar algún rol determinado, para utilizarlo en correspondencia a una función que deba ejecutar.

Luego de realizar un estudio de los artefactos propuestos por el flujo de trabajo de pruebas de Rup y la Ayuda del Rational 2003, se considera, que los que se proponen a continuación son los más importantes y necesarios para el Área temática SAS, teniendo en cuenta las características de sus proyectos. En el procedimiento se han clasificado los artefactos según dos criterios: de entrada y de salida.

**Especificación de los casos de uso del negocio:** Es un documento donde se registra la descripción textual de cada caso de uso del negocio, describiendo como colaboran los trabajadores y entidades del negocio para ejecutar el proceso. En este documento se explica detalladamente el flujo básico y alterno, así como las sesiones y escenarios de los requisitos de cada caso de uso. Este artefacto se utiliza en la fase de Inicio, además es importante tenerlos en cuenta en la elaboración del Plan de Pruebas Aceptación del Usuario.

**Especificación de los casos de uso:** Es un documento donde se registra la descripción textual de los caso de uso del sistema. En el mismo se explica detalladamente el flujo básico y alterno de los caso de uso del sistema. Este artefacto se genera en la fase de Inicio del software y se utilizan en la elaboración del Plan de Pruebas del Sistema.

**Glosario de términos**: Es un documento donde se tiene en cuenta determinada terminología que puede ser desconocida para los usuarios y otros personas. Este documento se genera en la fase de Inicio del software y se va actualizando durante el ciclo de vida del producto.

**Documentación de la arquitectura**: Es un documento donde se describen las 1+4 vistas de la arquitectura. Este se genera en la fase de Elaboración del software.

**Plantilla de ambiente de desarrollo**: En este documento se registran todas las herramientas utilizadas en el desarrollo del software, este artefacto se genera en la fase de Elaboración del software.

**Manual de usuario**: Es un documento que explica los pasos a seguir para usar la aplicación. Este se comienza a elaborar a partir de la primera versión del sistema y concluye en la fase de Transición.

**Especificaciones de requisitos**: Es un documento donde se registran todos los requisitos funcionales y no funcionales del sistema. Este se genera en la fase de Inicio del software.

**Plantilla de diseño de casos de prueba:** Es un documento donde se recogen un conjunto de condiciones, que determinan si los casos de uso del sistema definidos son correctos. Lo que caracteriza un escrito formal de un caso de prueba, es que hay una entrada conocida y una salida esperada, las cuales son formuladas antes de que se ejecute la prueba. La entrada conocida debe probar una precondición y la salida esperada debe probar una postcondición. Este documento incluye una descripción de la funcionalidad que se va a probar. El mismo se elabora durante la fase de Construcción del software por el Diseñador de pruebas. *Anexo 6*

**Plan de prueba general**: Es un documento donde se describe las estrategias de pruebas a seguir durante todo el proceso de pruebas, las pruebas a realizar, los recursos a utilizar y la planificación de las pruebas. Este se realiza teniendo en cuenta el Plan de desarrollo de los proyectos del área temática y es elaborado por el Líder del Equipo de pruebas. *Anexo 11*

**Datos reales de la Base de Datos**: Es un documento que contiene juegos de datos reales creados por los desarrolladores de la aplicación, para probar las funcionalidades del sistema. Este documento se genera en la fase de Transición del software.

**Documento Código Fuente**: Es un documento donde se archiva todo el código fuente de la aplicación. Este documento se genera en la etapa de desarrollo del software que haya código listo para probar.

**Pruebas previas**: Es un documento donde se registran, todas las pruebas realizadas previamente a la aplicación. Este documento se genera a partir de los períodos de pruebas al software y sirve para tener un seguimiento de las pruebas realizadas en iteraciones posteriores. Es elaborado por el Analista de pruebas a partir de los resultados de las pruebas anteriores. *Anexo 12*

**Lista de Chequeo**: Es un listado de preguntas en forma de cuestionario que sirve para verificar el grado de cumplimiento de determinadas reglas establecidas. El Analista de pruebas es el encargado

de confeccionar la lista para comprobar la validez de documento a verificar. Esta lista es aplicada por el Probador y se aplica en los períodos de pruebas a la documentación**.**

**Modelo de casos de uso del negocio**: Describe los procesos de negocio de una empresa en términos de casos de uso y actores del negocio, que se corresponden con los procesos del negocio y los clientes, respectivamente. Es elaborado por el Analista de procesos de Negocio. Contiene Actores del Negocio, Casos de Uso del Negocio y Metas del Negocio.

**Modelo de casos de uso**: Es un modelo del sistema que contiene actores, casos de uso y sus relaciones. Constituye un contrato entre clientes y desarrolladores. Este modelo es utilizado como una actividad de entrada fundamental en el análisis, diseño y prueba. Es elaborado por el analista de sistema.

**Modelos de análisis**: Contiene clases del análisis y sus objetos organizados en paquetes que colaboran. Elaborado por el Arquitecto de SW

**Modelos de diseño**: Es un modelo de objetos que describe las realizaciones de los CU y sirve como una abstracción del modelo de implementación y su código fuente. El modelo de diseño es utilizado como una actividad esencial de entrada para la implementación y las pruebas.

**Modelo de Datos**: Describe la representación física y lógica de los datos persistentes utilizados en la aplicación. En casos donde la aplicación utilizara una BD relacional, el modelo de datos debe incluir elementos del modelo para procedimientos almacenados, disparadores, restricciones, etc. que definan la interacción de los componentes de la aplicación con el RDBMS (sistema de gestión de BD relacionales).

**Modelo de Despliegue**: Muestra la configuración de los nodos procesadores y la comunicación que existe entre ellos, además de las instancias de los componentes y objetos que residen dentro de los nodos.

**Reglas del Negocio**: Describen políticas que deben cumplirse o condiciones que deben satisfacerse, por lo que regulan algún aspecto del negocio. El proceso de especificación implica que hay que "identificarlas" dentro del negocio, "evaluar" si son relevantes dentro del campo de acción que se está modelando e "implementarlas" en la propuesta de solución.

**Estándares de codificación**: Es un documento donde se registran todos los estándares de codificación. Este artefacto se debe tener en cuenta en la fase de Construcción del software.

**Estándares de Diseño**: Es un documento donde se registran todos los estándares de diseño de las aplicaciones para la salud. Este artefacto se debe tener en cuenta en la fase de Elaboración del software.

**Resultado de Prueba**: Es un documento donde se registran los resultados de las pruebas realizadas. Es elaborado por el Probador**.** *Anexo 7*

**Evaluación de Pruebas**: Es un documento que contiene una evaluación de los resultados de los esfuerzos de las pruebas, tales como la cobertura del caso de prueba y el estado de los defectos. Es elaborado por el Probador. *Anexo 9*

**Informe de solicitud de cambios**: Es un documento donde se registran todos los cambios que sean necesarios hacer en el sistema, luego de las pruebas realizadas por el cliente, estos son analizados y llevados a la mesa de negociaciones de ambas partes. Los cambios aceptados y pactados pasaran a ser nuevos requisitos. Es elaborado por el Analista de prueba. *Anexo 10*

**Informe de No Conformidades**: Listado que se genera luego de la aplicación de las pruebas a la documentación y las pruebas al software para registrar todas las no conformidades encontradas. Este artefacto es utilizado por el Probador. *Anexo 8*

**Ambiente de pruebas**: Es un documento donde se registran todos los elementos que se necesitan para ejecutar cada tipo de prueba en el entorno del usuario, simulando el entorno de despliegue. Estos elementos son: cantidad de computadoras, dispositivos, características de redes, sistemas operativos, navegadores, herramientas y antivirus. El mismo es elaborado por el Líder del Equipo de pruebas, el Analista de pruebas y el Diseñador de pruebas en la fase de Transición del software.

**Plan de Pruebas de Aceptación:** Es un documento donde se describe las estrategias de pruebas a seguir durante todo el proceso de pruebas del período de Pruebas de Aceptación del Usuario. Se definen las pruebas a realizar, los recursos a utilizar y la planificación de las pruebas teniendo en cuenta los requisitos del negocio especificados por el cliente. Es elaborado por el Analista y Diseñador de las pruebas. *Anexo 13*

**Plan de Pruebas del Sistema**: Es documento donde se describe las estrategias de pruebas a seguir durante todo el proceso de pruebas del periodo de Pruebas del Sistema. Se definen las pruebas a realizar, los recursos a utilizar y la planificación de las pruebas teniendo en cuenta los requisitos funcionales y no funcionales que debe cumplir el sistema. Es elaborado por el Analista y Diseñador de las pruebas. *Anexo 13*

**Plan de Pruebas de Integración de Subsistemas**: Es documento donde se describe las estrategias de pruebas a seguir durante todo el proceso de pruebas del período de Pruebas de Integración. Se definen las pruebas a realizar y la planificación de las mismas, así como los recursos a utilizar en el período de Revisión del Diseño de la Arquitectura. Es elaborado por el Analista y Diseñador de las pruebas. *Anexo 13*

**Plan de Pruebas de Unidad**: Es documento donde se describe las estrategias de pruebas a seguir durante todo el proceso de pruebas del período de Pruebas de Unidad. Se definen las pruebas a realizar y la planificación de las mismas, así como los recursos a utilizar en el período de Revisión del Diseño de la Arquitectura. Es elaborado por el Analista y Diseñador de las pruebas. *Anexo 13*

### 3.6.7 Pruebas a realizar

El proceso de pruebas paralelo al proceso de desarrollo del software, es la idea a seguir en el procedimiento que se propone, debido a que durante el avance de un software entre el 30% y el 40% de la actividad está relacionada con las pruebas. Por este motivo, es esencial plantear las actividades de pruebas al principio del proyecto, en lugar de hacerlo una vez terminada la codificación.

Se propone que las pruebas a realizar estén guiadas a través de las labores de pruebas que plantea el modelo de prueba W. Este modelo muestra cómo se relacionan las tareas de pruebas con las tareas en el modelo de desarrollo del software, refinando la prioridad de las mismas y la dependencia entre las actividades de desarrollo y las de pruebas. Aún siendo este método tan simple como el modelo V, el modelo W hace patente la importancia de ordenar las actividades individuales de pruebas. A continuación se presenta como queda el modelo de pruebas para el procedimiento que se propone.

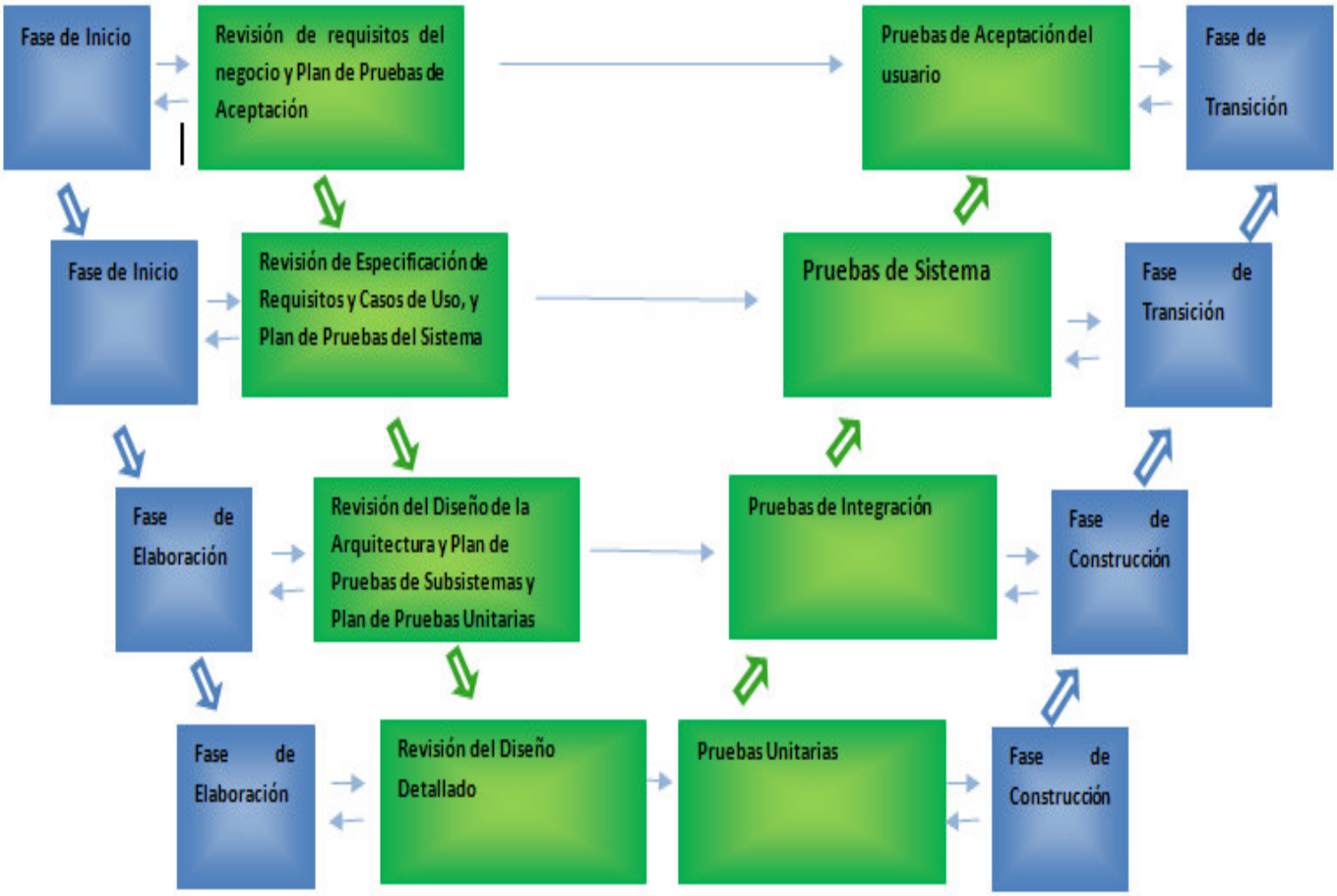



**Figura 3.3. Modelo de pruebas para el área temática SAS**

Se propone que el proceso de pruebas conste de 8 períodos, los cuales se dividen en dos partes, los cuatro primeros períodos dedicados a la revisión de la documentación y el resto a las pruebas del software, como se muestra a continuación:

**Tabla 3.2. Períodos de pruebas**

| **Revisión de la documentación** | **Pruebas de software** |
|---|---|
| 1. Revisión de Requisitos de Negocio | 5. Pruebas Unitarias |
| 2. Revisión de la Especificación de Requisitos y Casos de Uso | 6. Pruebas de Integración |
| 3. Revisión de la Arquitectura | 7. Pruebas de Sistema |
| 4. Revisión del Diseño Detallado | 8. Pruebas de Aceptación del usuario |

**Descripción de los períodos de revisión a la documentación**

Durante los períodos de revisión a la documentación el proceso de verificación de cada documento tiene una secuencia similar de actividades. A continuación se presenta el proceso de revisión de los documentos en una figura de forma general.

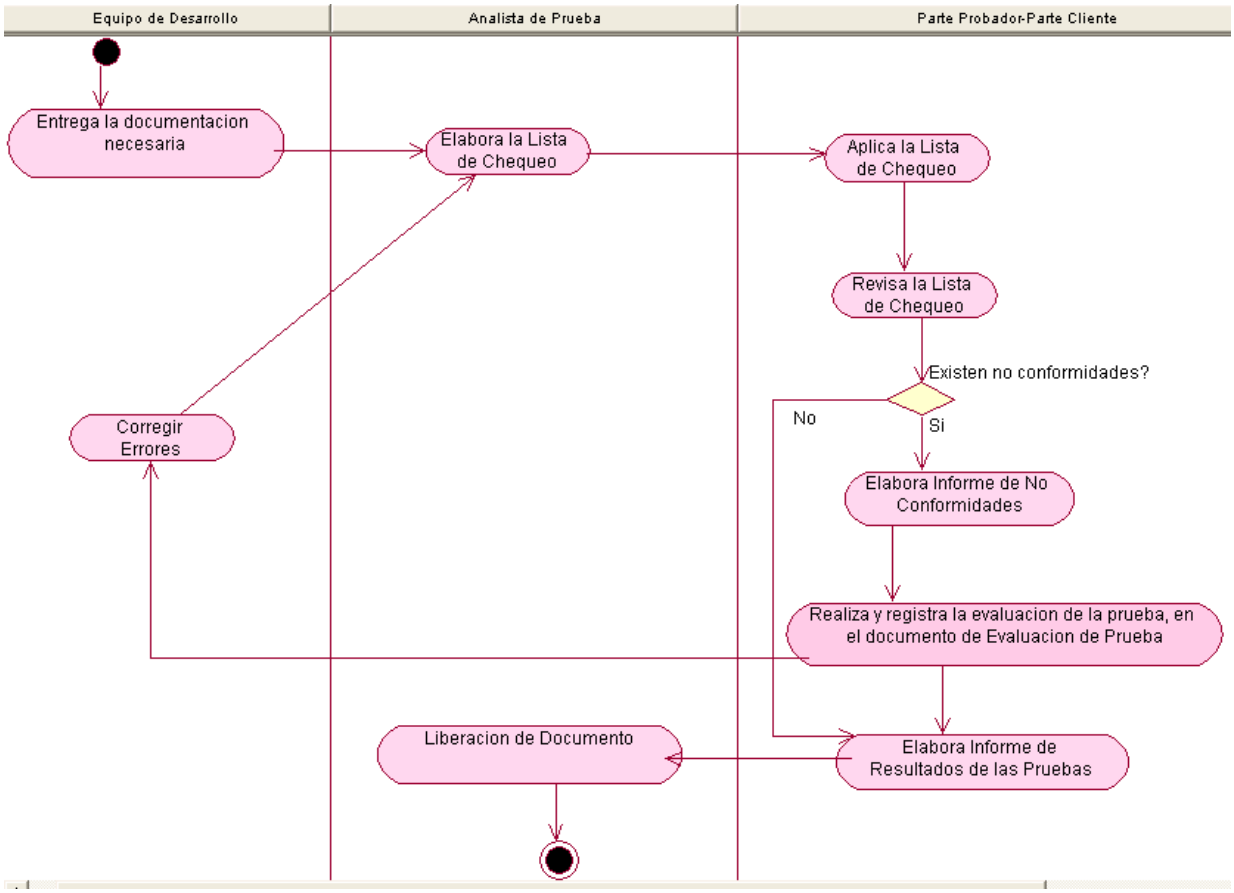



**Figura 3.4. Proceso de revisión de los documentos**

**Revisión de requisitos de Negocio**

Los requisitos del negocio marcan el inicio del desarrollo de un proyecto y es responsabilidad del Analista del proyecto definir los requisitos según las necesidades presentadas por el cliente, el alcance, las reglas y los objetivos del negocio presentado.

Las pruebas en esta primera actividad se centran en la revisión de los Modelos del Negocio. Estas revisiones se realizan a través de una lista de chequeo, la cual es elaborada por el Analista de prueba teniendo en cuenta las consideraciones pertinentes del Equipo de Desarrollo y el cliente, *Anexo 14.* Esta lista es aplicada por el Probador, para comprobar la validez del Modelo de Casos de Uso del negocio. Si existen errores, el Probador registra las no conformidades encontradas en el Informe de No

Conformidades de Requisitos del Negocio, las cuales son analizadas posteriormente en el Equipo de Desarrollo, donde se clasifica y se valora su respectiva respuesta.

Luego el Probador realiza una evaluación de las pruebas realizadas, la misma queda registrada en el Documento de Evaluación de Pruebas. Los resultados se contemplan en el resumen de Resultados de Pruebas por el Probador y discutidos con las partes interesadas llegando a un acuerdo de aprobación y firma de dicho documento. De esta forma concluye el ciclo de las pruebas en el período de Revisión de Requisitos del Negocio.

En este período el Analista de pruebas elabora el Plan de Pruebas de Aceptación, teniendo en cuenta los requisitos del negocio descritos por el cliente, el cual es utilizado en el períodos de Pruebas de Aceptación del Cliente.

Procedimiento para desarrollar la Verificación de los Requisitos del Negocio

1- Llenar los campos de la lista de chequeo, teniendo en cuenta las características peculiares del negocio planteado.
2- Efectuar la revisión de los parámetros a evaluar.
3- Conformar el Informe de No Conformidades.
4- Evaluar los resultados de las pruebas realizadas.
5- Elaborar con los resultados obtenidos el documento de Resultados de Pruebas.
6- Aprobar el Resultados de Pruebas y firmar dicho documento.

Artefactos involucrados en el período

- ✓ Lista de Chequeo de Requisitos del negocio (Entrada)
- ✓ Especificación de casos de uso del negocio (Entrada)
- ✓ Modelo de casos de uso del negocio (Entrada)
- ✓ Reglas del Negocio (Entrada)
- ✓ Glosario de término (Entrada)
- ✓ Informe de No Conformidades (Salida)
- ✓ Resultados de Pruebas (Salida)
- ✓ Evaluación de Pruebas (Salida)
- ✓ Plan de Pruebas de Aceptación (Salida)

**Revisión de la Especificación de Requisitos y Casos de Uso**

Después de lograr un buen entendimiento del negocio, se realiza la definición de los requisitos funcionales y no funcionales que debe de cumplir el sistema, más adelante se elabora el Modelo de Caso de Uso de Sistema, del cual se seleccionan los casos de uso críticos.

La revisión de la Especificación de Requisitos es la forma de verificar (probar) que los requisitos están correctos, la misma pretende comprobar tres principales atributos de calidad de los requisitos: corrección (carencia de ambigüedad), coherencia (especificación completa y clara del problema) y consistencia o exactitud de los requisitos (que no haya requisitos contradictorios). Estos aspectos se verifican con el fin de prevenir tantos errores como sea posible cuanto antes y de que se facilite el diseño y ejecución de los casos de pruebas. Este período se realiza cuando el software se encuentra transitando por la fase de Inicio.

Durante el proceso de inspección a la Especificación de Requisitos y Casos de Uso el Analista de pruebas conjuntamente con el Equipo de Desarrollo son los encargados de revisar el Documento de Especificación de Requisitos y Especificación de Casos de Uso, a partir del análisis de estos documentos, el Analista de pruebas elabora una lista de chequeo para cada documento, luego estas listas de chequeo son aplicadas por el Probador (*Anexo 15)*. En caso de que existan errores, el Probador elabora el Informe de No Conformidades de Especificación de Requisitos y Casos de Uso, los cuales son analizados posteriormente con el Equipo de Desarrollo.

Luego el Probador hace una evaluación de las pruebas realizadas, las mismas quedan registradas en el Documento de Evaluación de Pruebas. Los resultados son contemplados en el resumen de los Resultados de Pruebas por el probador y discutidos con las partes interesadas llegando a un acuerdo de aprobación y firma de dicho documento. De esta forma concluye el ciclo de las pruebas en el período de Revisión de Especificación de Requisitos y Casos de Uso.

El Analista de pruebas elabora el Plan de Pruebas del Sistema, el cual es utilizado posteriormente en el período de Pruebas de Sistema, teniendo en cuenta los requisitos funcionales y no funcionales definidos.

Procedimiento para desarrollar la verificación de Especificación de Requisitos

1- Llenar los campos de la lista de chequeo para las especificaciones propuestas, teniendo en cuenta las características peculiares de los Requerimientos y el Modelo de casos de uso del sistema.

2- Efectuar la revisión de los parámetros a evaluar.
3- Conformar el Informe de No Conformidades.
4- Evaluar los resultados de las pruebas realizadas.
5- Elaborar con los resultados obtenidos el documento de Resultados de Pruebas.
6- Aprobar los Resultados de Pruebas y firmar dicho documento**.**

Artefactos involucrados en el período

- ✓ Especificación de los Casos de Uso (Entrada)
- ✓ Especificación de Requisitos (Entrada)
- ✓ Lista de Chequeo de Especificación de Requisitos (Entrada)
- ✓ Lista de Chequeo de Especificación de Casos de Uso (Entrada)
- ✓ Glosario de término (Entrada)
- ✓ Plan de Pruebas del Sistema (Salida)
- ✓ Informe de No Conformidades (Salida)
- ✓ Informe de Solicitudes de Cambio (Salida)
- ✓ Resultados de las Pruebas (Salida)
- ✓ Evaluación de Pruebas (Salida)

**Revisión del Diseño de la Arquitectura**

Este período centra su mayor atención en el diseño de la arquitectura del sistema, las pruebas en el mismo están encaminadas a la revisión del diseño de la arquitectura. Para la revisión de la arquitectura es muy importante tener en cuenta la elección de las herramientas software que deben tener las PC clientes (sistema operativo, cliente grafico para administración de base de datos, lenguaje de programación, framework, entorno de desarrollo integrado, herramienta de modelado, navegadores) y la PC servidoras para el desarrollo del software (sistema operativo, sistema gestor de base de datos, sistema de control de versiones, servidor web), teniendo siempre presente que la mezcla de estos componentes deben cumplir con los requerimientos del sistema, y al mismo tiempo respetar las restricciones técnicas y económicas del proyecto.

Estas revisiones se efectúan a través de una lista de chequeo, que es elaborada por el Analista de pruebas y aplicada por el Probador, (*Anexo 16*). Para la elaboración de la misma se debe tener en cuenta los requisitos planteados en el Documento de Arquitectura y en la plantilla de Ambiente de Desarrollo del proyecto. Este período se realiza durante la fase de Elaboración del software.

En caso de que se encuentren errores el Probador es el encargado de conformar el Informe de No Conformidades, las cuales son analizadas posteriormente con el Equipo de Desarrollo, donde se clasifica y valora su respectiva respuesta. Luego el Probador hace una evaluación de las pruebas realizadas, la misma queda registrada en el Documento de Evaluación de Pruebas. Los resultados se contemplan en el resumen de los Resultados de Pruebas por el Probador y discutidos con las partes interesadas llegando a un acuerdo de aprobación y firma de dicho documento. De esta forma concluye el ciclo de las pruebas en el período de Revisión del Diseño de la Arquitectura.

En este período el Analista de pruebas elabora el Plan de Pruebas de Unidad y el Plan de Pruebas de Integración de Subsistema, estos planes se utilizan posteriormente en los períodos de pruebas de Unidad y de Integración.

Procedimiento para desarrollar la verificación del Diseño de la Arquitectura

1- Llenar los campos de la lista de chequeo, para verificar el diseño de la arquitectura, teniendo en cuenta las características peculiares del módulo o diseñar la lista en caso de que requiera de nuevos campos.
2- Efectuar la revisión de los parámetros a evaluar teniendo en cuenta los requisitos planteados en la plantilla de Ambiente de Desarrollo del proyecto y del Área Temática de SAS.
3- Conformar el Informe de No Conformidades.
4- Evaluar los resultados de las pruebas realizadas.
5- Elaborar con todos los resultados obtenidos el documento de Resultados de Pruebas.
6- Aprobar el Resultados de Pruebas y firmar dicho documento.

Artefactos involucrados en el período

- ✓ Lista de chequeo de la Revisión del Diseño de la Arquitectura (Entrada)
- ✓ Plantilla de Ambiente de Desarrollo. (Entrada)
- ✓ Plan de Pruebas de Integración de Subsistemas. (Salida)
- ✓ Plan de Pruebas de Unidad. (Salida)
- ✓ Informe de No Conformidades (Salida)
- ✓ Resultados de Pruebas (Salida)
- ✓ Evaluación de Pruebas (Salida)

**Revisión del Diseño Detallado**

Este período centra su mayor atención en el prototipo no funcional del sistema que se está construyendo. El Analista de pruebas a la hora de diseñar las mismas, debe tener en cuenta todas las peticiones que el cliente solicita que posea la parte visual del producto que se esta desarrollando, también tener en cuenta los Estándares de Diseño establecidos para los sistemas de salud desarrollados en el Área temática "Sistemas de Apoyo a la Salud". Esta revisión al diseño se realiza durante la fase de Elaboración del software.

Las revisiones están encaminadas a la verificación del prototipo no funcional y al diseño de la Base de Datos. Estas revisiones se efectúan a través de una lista de chequeo, la cual es elaborada por el Analista de pruebas teniendo en cuenta las consideraciones pertinentes del Equipo de Desarrollo y el cliente, (*Anexo 17*). La lista de chequeo es aplicada por el Probador, para revisar si el diseño de la Base de Datos es correcto y si el análisis y el diseño del prototipo no funcional cumplen con las características planteadas por el cliente y con los requisitos establecidos en el Documento de Estándares de Diseño del área temática SAS.

En caso de que se encuentren errores el Probador es el encargado de conformar el Informe de No Conformidades, las cuales son analizadas posteriormente con el Equipo de Desarrollo, donde se clasifica y se valora su respectiva respuesta. Luego el Probador hace una evaluación de las pruebas realizadas, la misma queda registrada en el Documento de Evaluación de Pruebas. Los resultados son contemplados en el resumen de los Resultados de Pruebas por el Probador y discutidos con las partes interesadas llegando a un acuerdo de aprobación y firma de dicho documento. De esta forma concluye el ciclo de las pruebas en el período de Revisión del Diseño Detallado.

Procedimiento para desarrollar la verificación del Diseño Detallado

1- Llenar los campos de la lista de chequeo para verificar el análisis y el diseño del prototipo no funcional y el diseño de la Base de Datos, teniendo en cuenta las características peculiares del módulo o diseñar la lista en caso de que requiera de nuevos campos.
2- Efectuar la revisión de los parámetros a evaluar teniendo en cuenta los estándares de diseño predefinidos en el Área Temática de SAS.
3- Conformar el Informe de No Conformidades.
4- Evaluar los resultados de las pruebas realizadas.
5- Elaborar con todos los resultados obtenidos el documento de Resultados de Pruebas.
6- Aprobar el Resultados de Pruebas y firmar dicho documento.

Artefactos involucrados en el período

- ✓ Modelos de Análisis (Entrada)
- ✓ Modelos de Diseño (Entrada)
- ✓ Modelo de caso de uso (Entrada)
- ✓ Modelo de Datos (Entrada)
- ✓ Lista de chequeo del Diseño Detallado (Entrada)
- ✓ Glosario de término (Entrada)
- ✓ Informe de No Conformidades (Salida)
- ✓ Resultados de Pruebas (Salida)
- ✓ Evaluación de Pruebas (Salida)

**Descripción de los períodos de pruebas al software**

Durante los períodos de pruebas al software el proceso de pruebas esta encaminado a la verificación de los requisitos funcionales y no funcionales del software. En cada uno de los períodos que se proponen se sigue una secuencia de actividades similares. A continuación se presenta una figura con la descripción de forma general del proceso.

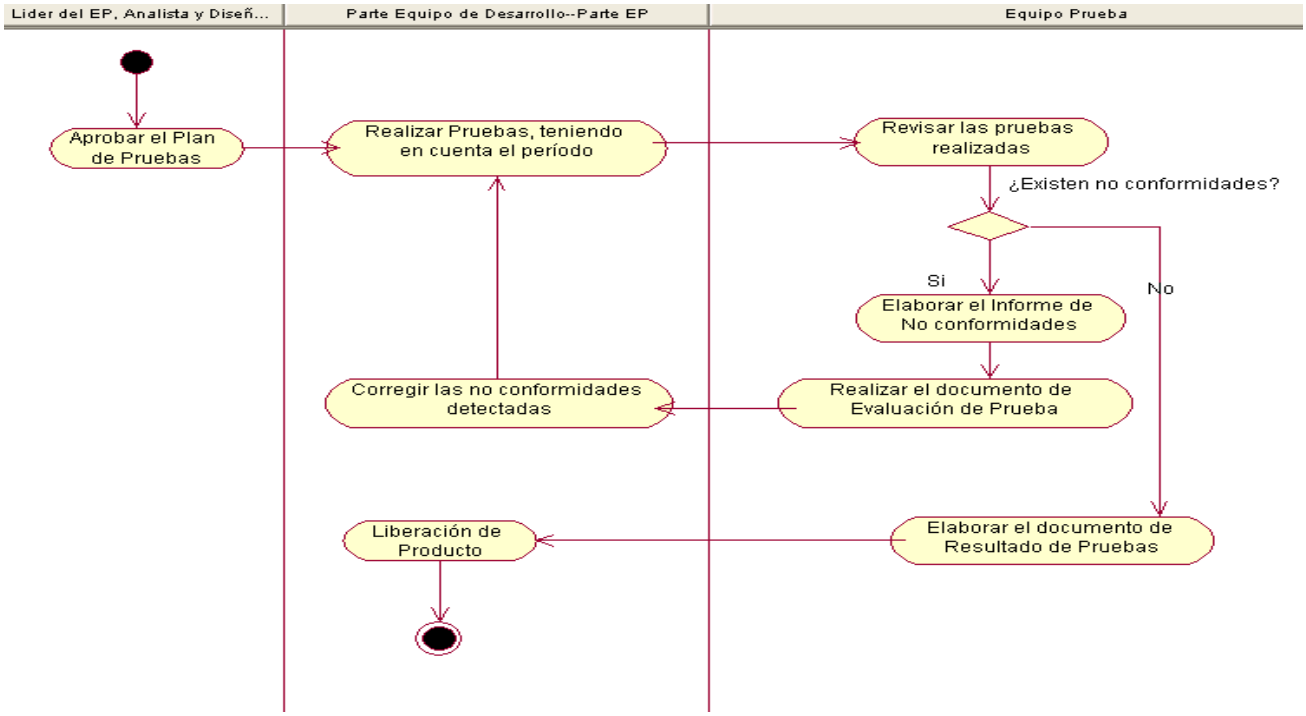



**Figura 3.5. Proceso de pruebas al software**

**Pruebas unitarias**

Es muy importante al inicio de la implementación establecer los estándares de codificación que deben tener en cuenta los programadores. Así se garantiza que otras personas puedan utilizar y comprender el código sin haberlo implementado y además posibilita que a la hora de ejecutar las pruebas resulte más fácil. Este período se realiza durante la fase de Construcción del software.

En la primera parte del período de Pruebas Unitarias, las pruebas están encaminadas a la revisión del código fuente del sistema que se esta desarrollando, debido a que es una forma de encontrar defectos rápidamente, siendo esta revisión una de las buenas prácticas propuestas por PSP.

La revisión del código consume tiempo, pero es mucho más eficiente realizarlas antes de comenzar a compilar o probar el sistema, debido a que a través de las mismas se ven los problemas y no los síntomas. Es decir, mientras se revisa el código se piensa en lo que el programa debe hacer, de esta manera cuando se nota algo que no es correcto, se puede ver el posible problema y verificar rápidamente el código, por tanto, las revisiones pueden ahorrar mucho tiempo si se tiene en cuenta que el tiempo transcurrido desde que se detecta el síntoma hasta que se llega al problema, es la mayor parte del coste de encontrar y corregir los defectos durante la compilación y pruebas. [68]

Esta revisión está a cargo del Diseñador de pruebas, el cual revisa el código, teniendo en cuenta los estándares de codificación establecidos para los proyectos de SAS y si la lógica de las funcionalidades implementadas está correcta.

Procedimiento para revisar el código de un producto

1- Imprimir un listado por cada módulo con todo el código fuente.
2- Revisar línea a línea el código.
3- Corregir los errores, tanto en las funcionalidades como en el cumplimiento de los estándares de codificación.
4- Documentar los errores encontrados en el Informe de No Conformidades.

La segunda parte del período centra su mayor atención en efectuarle pruebas unitarias a los objetos más pequeños del sistema. Las pruebas a realizar están encaminadas a los métodos, clases y módulos del sistema.

Las Pruebas Unitarias que se proponen ejecutar son:

- ✓ Método de Caja Blanca a través de la técnica del Camino Básico, con el objetivo de probar el código.
- ✓ Método de Caja blanca a través de la Prueba de Condición**,** con el objetivo de probar las condiciones lógicas contenidas en el código de un programa.

El Diseñador de Pruebas es el encargado de definir y verificar las pruebas a realizar, teniendo en cuenta la versión o iteración del producto y proporciona un orden lógico a las pruebas planificadas. El Probador ejecuta las pruebas diseñadas y registra los errores encontrados en el Informe de No Conformidades, las cuales son analizadas posteriormente con el Equipo de Desarrollo, donde se clasifica y valora la respectiva respuesta.

Luego de realizar las pruebas el Probador procede a hacer una evaluación de las mismas, conformando así el Documento de Evaluación de Pruebas, que luego es entregado al Jefe del proyecto. Los resultados serán contemplados en el Resumen de los Resultados de Pruebas por el Probador y discutidos con las partes interesadas, llegando a un acuerdo de aprobación y firma de dicho documento. De esta forma concluye el ciclo de las pruebas en el período de Pruebas Unitarias.

Procedimiento para efectuar el Método de Caja Blanca a través de la técnica del Camino Básico

1- Recibir el Código Fuente y la documentación necesaria

El Líder del Equipo de Pruebas, recibe de manos del líder del proyecto el código fuente del módulo o sistema al cual se le van a ejecutar las pruebas de Caja Blanca.

2- Evaluación de la documentación

El Líder del Equipo de Pruebas entrega la documentación al Analista de pruebas y este verifica si se definió un estándar de código, para hacer más legible y entendible el código, determinando si se necesita para aplicar la prueba uno o más probadores, eso está en dependencia de la magnitud del proyecto. El Probador se encarga de revisar y hacerle pruebas a todo el código.

3 - Evaluación de los Requisitos de hardware y software

Atendiendo a los requisitos especificados en la documentación recogida, se dispone en el EP de un determinado número de probadores y de computadoras que cumplan con los requisitos de hardware y software.

4 **-** Estructuración de las Particiones del Código

Aplicar la misma política de partición para todos los módulos del sistema.

Se siguen los siguientes criterios de partición:

a) Por funcionalidad de los métodos o formas: De cada funcionalidad del código, por ejemplo: insertar o eliminar, se divide en fragmentos independientes a probar.

b) Por regiones de código definidas: Si el programador ya definió regiones en el código fuente se hace de cada región una partición a probar.

c) Por caso de uso (si estos no son muy complejos): Cada caso de uso se prueba de forma independiente a través de su código fuente.

En cualquiera de los casos se debe seguir el mismo orden lógico del programador y por tanto del programa. [69]

5 - Obtención de los Grafos de Flujo

I. Para construir el grafo se debe tener en cuenta la notación para las diferentes instrucciones. *Anexo 18.*
II. Se numeran las instrucciones siguiendo estas notaciones y atendiendo a las particularidades del código fuente. *Anexo 19*
III. Se diseña el grafo siguiendo la numeración anteriormente establecida.[70]

6 - Cálculo de la Complejidad Ciclomática

Se determina el valor de la complejidad ciclomática. [71]

7 - Obtención de los Caminos Básico

Posibilita que se verifique la lógica de cada parte del código. [72]

<u>Procedimiento para efectuar el Método de Caja Blanca a través de la Prueba de Condición</u>

1- Imprimir un documento con los métodos fundamentales de todo el sistema.
2- Efectuar la revisión de las condiciones lógicas al código a través de la Prueba de Condición.
3- Conformar el Informe de No Conformidades.
4- Hacer una evaluación final de las pruebas realizadas.
5- Elaborar con los resultados obtenidos el documento de Resultados de Pruebas.
6- Aprobar los Resultados de las Pruebas y firmar dicho documento.

Artefactos involucrados en el período

- ✓ Documento Código fuente (Entrada)
- ✓ Estándares de Codificación (Entrada)
- ✓ Plan de Prueba de Unidad (Entrada)
- ✓ Pruebas previas (Entrada)
- ✓ Informe de No Conformidades (Salida)
- ✓ Evaluación de pruebas (Salida)
- ✓ Resultados de las pruebas (Salida)
- ✓ Evaluación de Pruebas (Salida)

**Pruebas de Integración**

Luego de terminado el proceso de implementación, el próximo paso es realizar la integración entre los módulos. Los proyectos que se desarrollan en el área temática SAS, intercambian los datos a través de servicios web, es por esto que es importante hacer la verificación de todas las funciones luego de la integración, para comprobar si todos los procesos siguen funcionando de forma correcta, esto es posible lograrlo efectuando las pruebas de integración, las cuales sirven para verificar si el producto integrado funciona bien, ya que de una buena integración depende de que la información que se procesa sea válida. Este período se lleva a cabo durante la fase de Construcción del software.

El Diseñador de pruebas es el encargado de definir y verificar las pruebas a realizar, teniendo en cuenta la versión o iteración del producto y proporciona un orden lógico a las pruebas planificadas. El Probador ejecuta las pruebas diseñadas y registra los errores encontrados en el Informe de No Conformidades, las cuales son analizadas posteriormente con el Equipo de Desarrollo, donde se clasifica y valora la respectiva respuesta.

Luego de realizar las pruebas el Probador procede a hacer una evaluación de las mismas, conformando así el Documento de Evaluación de Pruebas, que luego es entregado al Jefe del proyecto. Los resultados serán contemplados en el Resumen de los Resultados de Pruebas por el Probador y discutidos con las partes interesadas, llegando a un acuerdo de aprobación y firma de dicho documento. De esta forma concluye el ciclo de las pruebas en el período de Pruebas Integración.

Las pruebas que se proponen ejecutar son:

- ✓ Prueba de Referencia Cruzada

- ✓ Método de Caja negra

**Prueba de Referencia Cruzada**

La idea fundamental de esta prueba es tener en cuenta los flujos que conllevaron a no conformidades detectadas en un módulo, para ejecutarlos en el resto de los módulos y ver la reacción de este.

Objetivo

Verificar que los componentes genéricos implementados para su uso en todos los módulos funcionen correctamente, para ello se deberá realizar una lista de dichos componentes especificando la funcionalidad que implementa, para poder aplicar las pruebas.

Técnica

Al detectarse un error en una funcionalidad de un módulo y que pudiera haber sido implementado en cualquiera de los restantes, se tomarán las condiciones bajo las cuales ocurrió el mismo y se repetirá en el resto de los módulos, para comprobar si este es general a todo el sistema. [73]

Procedimiento para efectuar la Prueba de Referencia Cruzada

1- Aplicar la técnica de la prueba de referencia cruzada al módulo.
2- Registrar las no conformidades encontradas en el Informe de No Conformidades.
3- Hacer una evaluación final de las pruebas realizadas.
4- Elaborar con los resultados obtenidos el documento de Resultados de Pruebas.
5- Aprobar los Resultados de las Pruebas y firmar dicho documento.

**Método de Caja Negra**

Esta técnica se utiliza para probar las funcionalidades del sistema.

Objetivo

El propósito de este método es probar si las funcionalidades definidas por el cliente funcionan correctamente.

Técnica

Particiones o Clases de Equivalencia

Procedimiento para efectuar la Técnica de Particiones Equivalentes

1- Descripción general

En esta sección se hace una descripción general de los principales aspectos a tener en cuenta para realizar las pruebas funcionales de Caja Negra, partiendo de la técnica de Particiones Equivalentes.

2- Funcionalidades a revisar

Para cada una de las funcionalidades se especifican los siguientes elementos:

I. Descripción de la funcionalidad

Se hace una detallada descripción de la funcionalidad.

II. Flujo Central

Se describen los pasos que se deben realizar para llegar a la funcionalidad que se va a probar, lo que le permite al Probador que es el que va a ejecutar los Casos de Prueba saber que pasos debe realizar para llegar a la funcionalidad, por lo que no es necesario que conozca al detalle el Sistema, además si existe algún error será más fácil para los desarrolladores saber exactamente donde está.

III. Condiciones de ejecución

Se especifican los prerrequisitos que se deben cumplir para que se pueda ejecutar el Caso de Uso y la funcionalidad en cuestión.

IV. Iteraciones

En la siguiente tabla se registran los datos con los cuales se prueba la funcionalidad de todo el sistema, se especifica si los datos son válidos o no y la respuesta que se espera por parte del sistema, cuando se ejecuta el caso de prueba se coloca lo que realmente hizo el sistema y si hay alguna observación que hacer sobre la prueba realizada.

| **Clases válidas** | **Clases inválidas** | **Resultado esperado** | **Resultado de la prueba** | **Observaciones** |
|---|---|---|---|---|
| <Clases válidas > | <Clases inválidas > | <Resultado esperado de la prueba según lo especificado en el Flujo de Caso de Uso> | <Resultado de la prueba > | < Observaciones > |

3- Descripción del Caso de Uso para el diseño del caso de prueba

En este caso se refiere específicamente a una descripción detallada del caso de uso para el cual se diseña el caso de prueba, permitiendo describir detalladamente el caso de uso, en caso de que existan parámetros nuevos que no estén contemplados en ninguno de los casos anteriores.

4- Funcionalidades a probar

En esta sección se especifican las funcionalidades que se probarán, teniendo en cuenta la descripción del flujo central y de todos los flujos alternos que existan.

5- Condiciones de ejecución

Se expone con detalles las condiciones mínimas necesarias para que se pueda ejecutar la funcionalidad.

6- Completar la tabla del caso de prueba

Para la obtención del caso de prueba se completa la tabla con datos reales del caso de uso al que se le diseña el caso de pruebas.

Ejemplo resuelto de Diseño de Caos de prueba funcional. *Anexo 21*

Artefactos involucrados en el período

- ✓ Especificación de Casos de Uso (Entrada)
- ✓ Diseño de Casos de Pruebas (Entrada)
- ✓ Plan de Prueba de integración de subsistemas (Entrada)
- ✓ Pruebas previas (Entrada)
- ✓ Informe de No Conformidades (Salida)
- ✓ Evaluación de las pruebas (Salida)
- ✓ Resultados de las pruebas (Salida)
- ✓ Evaluación de Pruebas (Salida)

**Pruebas de Sistema**

Luego de verificar que las unidades más pequeñas del sistema y la integración entre sus módulos son correctas, el producto cuenta con las condiciones mínimas necesarias para realizarles pruebas más generales. Las pruebas al sistema se realizan durante la fase de Transición del software.

El Diseñador de pruebas es el encargado de definir y verificar las pruebas a realizar, teniendo en cuenta la versión o iteración del producto, de acuerdo a las pruebas especificadas y proporciona un orden lógico a las pruebas planificadas. El Probador ejecuta las pruebas diseñadas y registra los errores encontrados en el Informe de No Conformidades, las cuales serán analizadas posteriormente con el Equipo de Desarrollo, donde se clasifica y valora la respectiva respuesta.

Luego de realizar las pruebas el Probador procede a hacer una evaluación de las pruebas, conformando así el Documento de Evaluación de Pruebas, que luego es entregado al Jefe del proyecto. Los resultados son contemplados en el Resumen de los Resultados de Pruebas por el Probador y discutidos con las partes interesadas, llegando a un acuerdo de aprobación y firma de dicho documento. De esta forma concluye el ciclo de las pruebas en el período de Pruebas de Sistema.

Las pruebas que se proponen que se realicen son:

- ✓ Prueba de Volumen
- ✓ Prueba de Carga
- ✓ Prueba de Usabilidad
- ✓ Prueba de Stress
- ✓ Prueba de Configuración

**Prueba de Volumen**

Enfocada en verificar las habilidades de los programas para manejar grandes cantidades de datos, tanto como entrada, salida o residente en la base de datos. [74]

Objetivo

Verificar que la aplicación funciona adecuadamente con un máximo tamaño de Base de Datos.

Técnica

- ✓ Utilizar un tamaño máximo de Base de datos (actual o con datos representativos) y múltiples clientes para correr consultas simultáneamente para períodos extendidos, de esta manera se determinan la cantidad de datos con la cual el sistema falla.

**Prueba de Carga**

Es usada para validar y valorar la aceptabilidad de lo límites operacionales de un sistema bajo carga de trabajo variable, mientras el sistema bajo prueba permanece constante. [75]

Objetivo

Verificar que la aplicación funciona adecuadamente cuando existe un posible número de clientes (conectados o simulados) todos ejecutando la misma función por un período extendido.

Técnica

- ✓ Usar múltiples clientes, realizando las mismas peticiones al sistema, para determinar la carga máxima que el sistema soporta en un período dado.

**Prueba de usabilidad**

Prueba enfocada a factores humanos, estéticos, consistencia en la interfaz de usuario, ayuda sensitiva al contexto, asistencia a documentación de usuarios y materiales de entrenamiento. [76]

Objetivo

Medir que tan bien puede una persona usar el software, es decir que la navegación en la aplicación sea amigable.

Técnicas

Las técnicas que se proponen utilizar son las siguientes:

- ✓ El Probador del EP realiza una interacción con el producto a través del manual de usuario.
- ✓ Seleccionar a un grupo de usuarios y solicitarles que lleven a cabo las tareas para las cuales fue diseñada el software, este resultado es recogido por el Diseñador de pruebas y el Probador, los cuales registran los errores y dificultades con las que se encuentren los usuarios.

**Prueba de Stress**

La prueba de Stress está enfocada a evaluar cómo el sistema responde bajo condiciones anormales en un período de tiempo determinado (extrema sobrecarga, insuficiente memoria, servicios y hardware no disponible, recursos compartidos no disponible). [77]

Objetivo

Evaluar como el sistema responde bajo condiciones anormales.

Técnicas

- ✓ Se establecen condiciones no previstas en el entorno de trabajo en un tiempo dado para comprobar si el sistema es capaz de no perder la información procesada, o de enviar una falsa información. (falta de electricidad, pérdida de la red en el momento de iniciada una transacción de datos)

**Prueba de Configuración**

Enfocada a asegurar que funciona en diferentes configuraciones de hardware y software.

Objetivo

Evaluar como el sistema se comporta en diferentes configuraciones.

Técnicas

- ✓ Probar si el sistema funciona en diferentes Sistemas Operativos y navegadores.
- ✓ Probar si el sistema funciona con diferentes dispositivos.
- ✓ Probar si el sistema funciona con cada tipo de dispositivo de hardware y en cualquier configuración.

Procedimiento para ejecutar las pruebas del Sistema

1- Verificar si el entorno de trabajo es correcto para aplicar la prueba seleccionada (configuración de hardware y software)
2- Aplicar la técnica de la prueba seleccionada.
3- Registrar en el Informe de No Conformidades todos los errores detectados.
4- Hacer una evaluación final de las pruebas realizadas
5- Elaborar con todos los resultados obtenidos el documento de Resultados de Pruebas
6- Aprobar los Resultados de las Pruebas y firmar dicho documento

Artefactos involucrados en el período

- ✓ Plan de Pruebas del Sistema (Entrada)
- ✓ Plantilla de Ambiente de Desarrollo (Entrada)
- ✓ Modelo de despliegue (Entrada)
- ✓ Ambiente de Prueba (Entrada)
- ✓ Pruebas previas (Entrada)
- ✓ Informe de No Conformidades (Salida)
- ✓ Informe de Solicitudes de Cambio (salida)

- ✓ Evaluación de las pruebas (Salida)
- ✓ Resultados de las pruebas (Salida)
- ✓ Evaluación de Pruebas (Salida)

**Pruebas de Aceptación del Usuario**

Las pruebas de aceptación son las pruebas ejercitadas por el usuario en su entorno de trabajo y se realizan con el objetivo de validar si el producto software cumple con las funcionalidades requeridas. Estas pruebas tienen vital importancia pues los resultados proyectados deciden la aceptación o el rechazo del producto, son realizadas cuando el software se encuentra en la fase de Transición.

El ambiente donde se desarrollan las pruebas es uno de los elementos de mayor importancia a la hora de planificar los elementos mínimos necesarios. Por ello se toma como principio básico: llegar a simular el ambiente más real posible; garantizando un buen nivel de verificación de las funcionalidades de la aplicación.

Se define que todas las pruebas se realicen de forma manual; debido a que las pruebas son de aceptación con el cliente, no es aconsejable la utilización de algún tipo de herramienta que automatice las mismas, porque al cliente siempre le interesa comprobar por si mismo las condiciones y capacidades del producto por el cual está pagando.

Las pruebas de aceptación final del cliente se inician con la aprobación y firma del Plan de Pruebas de Aceptación, que incluye el plan de las pruebas a realizan, la relación de los datos del personal que intervendrán en las pruebas y la selección de los módulos a probar.

En este período las pruebas las realiza el cliente, aunque es recomendable la presencia del Probador. El propósito de este período es verificar el cumplimiento de los procesos elementales del negocio, los Casos de Uso del Negocio, así como las funcionalidades de cada uno de los módulos. Las no conformidades del cliente son recogidas por el Probador en el Informe de No Conformidades, las cuales son analizadas y valorada su solución, que luego es asignada al Equipo de Desarrollo. Posteriormente se procede a hacer una evaluación de las pruebas, conformando así el Documento de Evaluación de Pruebas, que es entregado al Jefe del proyecto. Las solicitudes de cambios que aparezcan serán recolectadas por el Analista de pruebas en su respectivo Informe de Solicitudes de Cambio, las cuales serán analizadas y llevadas a la mesa de negociaciones entre ambas partes. Los resultados serán contemplados en el resumen de los Resultados de Pruebas por el Probador y

discutidos con las partes interesadas llegando a un acuerdo de aprobación y firma de dicho documento. De esta forma concluye el ciclo de las pruebas en el período de Pruebas de Aceptación del Usuario.

Los tipos de pruebas que se proponen ejecutar son:

- ✓ Prueba de Función
- ✓ Prueba de seguridad y control de acceso.
- ✓ Prueba de integridad de los datos y la base de datos.
- ✓ Prueba de diseño informacional.

**Prueba de Función**

La prueba de Función se enfoca en requerimientos para verificar que se corresponden directamente a casos de usos o funciones y reglas del negocio. Este tipo de prueba se basa en técnicas de Caja Negra, que consisten en verificar la aplicación y sus procesos interactuando por medio de la interfaz de usuario y analizar los resultados obtenidos.[78]

**Objetivo**

El objetivo de esta prueba es verificar la aceptación de los datos, el proceso, la recuperación y la implementación correcta de las reglas del negocio.

**Técnica**

- ✓ Ejecutar cada proceso o función usando datos válidos y no válidos, para verificar lo siguiente:
  - ¿Se obtienen los resultados esperados cuando se usan datos válidos?
  - ¿Cuando se usan datos no válidos se despliegan los mensajes de error o advertencia apropiados?
  - ¿Se aplica apropiadamente cada regla del negocio?
- ✓ El cliente lleva a cabo la prueba de aceptación mediante la comparación del programa con lo acordado en el Documento Visión, para esto el diseñador de pruebas realiza diversos diseños de casos de pruebas usando datos no válidos para demostrar que el programa no cumple con el contrato, si estos casos de prueba no tienen éxito, el programa es aceptado.

**Prueba de seguridad y control de acceso**

La Prueba de Seguridad y Control de Acceso se enfoca a la seguridad en el ámbito de aplicación, incluyendo el acceso a los datos y a las funciones de negocios asociadas a cada rol de usuario, la cual asegura que, los usuarios solo accedan a los procesos de acuerdo al rol asignado.[79]

Objetivo

Verificar que un usuario pueda acceder solo a las funciones o datos para los cuales su tipo de rol tiene permiso.

Técnica

1. Identificar y hacer una lista de cada tipo de usuario, las funciones y datos sobre las que cada tipo tiene permiso.
2. Crear pruebas para cada tipo de usuario y verificar cada permiso creando operaciones específicas para cada tipo de usuario.
3. Modificar el tipo de usuario y volver a ejecutar las pruebas para los mismos usuarios. En cada caso, verificar que las funciones o datos adicionales están correctamente disponibles o sean denegados.

**Prueba de integridad de los datos y la base de datos**

Se enfoca en verificar la integridad de las funcionalidades de la base de datos. Esta prueba requiere un entorno de administración del Sistema de Gestión de Base de Datos o controladores, para ingresar o modificar información directamente en la base de datos. Los procesos deben ser invocados manualmente. Se deben usar bases de datos con una cantidad pequeña de información, para aumentar la facilidad de inspección de los datos y para verificar que no sucedan eventos no aceptables.[80]

Objetivo

Asegurar que los métodos y procesos de acceso a la base de datos funcionan correctamente y sin corromper datos.

Técnica

1. Revisar los métodos o procesos de acceso a la base de datos con datos válidos y no válidos.

2. Inspeccionar la base de datos para asegurarse de que se han guardado los datos correctos, que todos los eventos de la base de datos ocurrieron correctamente, o repasar los datos devueltos para asegurar que se recuperaron datos correctos por la vía correcta.

**Prueba de diseño informacional**

La prueba está encaminada a verificar que los requisitos del diseño sean los especificados por el cliente, teniendo en cuenta que todos los módulos cumplan con los estándares de diseño establecidos. [81]

Objetivo

Verificar que el diseño de la interfaz y de los reportes entre los diferentes módulos que componen al sistema, mantenga las mismas pautas de diseño gráfico.

Técnica

A partir de la definición de las pautas del diseño informacional aplicados al sistema, chequear que estas se cumplan en todos los módulos. Es importante considerar el cumplimiento de aspectos tales como:

- ✓ Color
- ✓ Tipo y tamaño de la letra tanto para la entrada de datos como para la salida de los mismos.
- ✓ Similitud en el diseño gráfico y distribución de los componentes en las interfaces y reportes.
- ✓ Contenido de los mensajes que emite la aplicación para eventos como: errores, alertas, información, ayudas, etc.

Artefactos involucrados en el período

- ✓ Especificación de requisitos (Entrada)
- ✓ Modelos del Negocio (Entrada)
- ✓ Plan de Prueba de Aceptación (Entrada)
- ✓ Pruebas previas (Entrada)
- ✓ Informe de No Conformidades (Salida)
- ✓ Informe de Solicitudes de Cambio (Salida)
- ✓ Evaluación de Pruebas (Salida)
- ✓ Resultados de Pruebas (Salida)

Procedimiento para ejecutar las pruebas de Aceptación del Cliente

1- Verificar si el entorno de trabajo es correcto para aplicar la prueba seleccionada (configuración de hardware y software)
2- Aplicar la técnica de la prueba seleccionada.
3- Registrar en el Informe de No Conformidades todos los errores detectados.
4- Hacer una evaluación final de las pruebas realizadas
5- Elaborar el Informe de Solicitudes de Cambio.
6- Elaborar con todos los resultados obtenidos el documento de Resultados de Pruebas
7- Aprobar los Resultados de las Pruebas y firmar dicho documento

## 3.7 Ventajas del procedimiento

- ✓ Permite que el área temática SAS cuente con un procedimiento de pruebas de software para sus productos durante su ciclo de vida.
- ✓ Mejora el proceso de pruebas que se lleva a cabo actualmente en el área temática.
- ✓ Garantiza que los productos que se desarrollan en el área temática SAS, al ser terminados, cuenten con la calidad mínima requerida; facilitando así el trabajo que se realiza en el laboratorio de Calidad de la Facultad 7.

## 3.8 Riesgos del procedimiento

- ✓ No tener conformado el Equipo de Pruebas con todos los roles definidos.
- ✓ No ejecutar en cada fase por la que este transitando el software las pruebas definidas en los períodos de pruebas.
- ✓ No realizar todas las actividades definidas para llevar a cabo una prueba.
- ✓ Comenzar a desarrollar las actividades de pruebas sin todos los artefactos de entrada definidos.
- ✓ Mala conformación de los artefactos de salida al ser generados y registrados.
- ✓ Existencia de poca comunicación entre los integrantes del Equipo de Pruebas
- ✓ Existencia de poca comunicación entre el Equipo de pruebas y el Equipo de Desarrollo.

## 3.9 Conclusiones

- ✓ En este capitulo se identificaron los elementos que conforman el procedimiento y la interacción entre los mismos.
- ✓ Se propone un Equipo de Pruebas, en el cual se definieron 5 roles con las respectivas actividades de pruebas que deben cumplir y se definieron períodos de pruebas para las fases por las que

transita el software, en los cuales se exigen diversos artefactos de entrada para realizar las pruebas y se generan importantes artefactos de salida.

- ✓ Para que el procedimiento de pruebas resulte satisfactorio, se debe cumplir con todas las actividades descritas en el flujo de trabajo de pruebas presentado, teniendo en cuenta todas las plantillas propuestas a realizar.

# Conclusiones

Con la presente investigación, luego de un estudio realizado para insertar un procedimiento de pruebas en el área temática Sistema de Apoyo a la Salud y en aras de apoyar la calidad de los proyectos que se desarrollan en la misma, se arribó a las siguientes conclusiones**:**

- ✓ Las pruebas son un aspecto determinante en la calidad de los productos software.

- ✓ Se demostró la necesidad de la puesta en práctica de un procedimiento de pruebas que contribuya a la calidad de los productos software que se desarrollan en el área temática SAS.

- ✓ Se ha cumplido el objetivo de la investigación, pues se realizó un procedimiento de pruebas de software para el área temática Sistemas de Apoyo a la Salud.

- ✓ Una vez puesto en práctica el procedimiento se contribuirá a la calidad de los productos del área temática, pues hasta el momento no se habían definido los elementos de pruebas en el desarrollo de los mismos.

# Recomendaciones

Dada la experiencia alcanzada con la confección de este trabajo, se recomienda para una mejor utilización del mismo:

- ✓ Aplicar y dar seguimiento al procedimiento propuesto en el área temática SAS.
- ✓ Realizar un análisis de los resultados obtenidos con la aplicación del procedimiento.
- ✓ Ejecutar el procedimiento propuesto en otras áreas temáticas.
- ✓ Proponer un Plan de Trabajo que contribuya a planificar mejor los tiempos de pruebas para los proyectos del área temática.
- ✓ Realizar un estudio de las pruebas específicas a efectuar a cada proyecto del área temática teniendo en cuenta sus características.

# Referencias Bibliográficas

[1] Desoft, Informatización de la Sociedad, 2002, Ciudad Habana, disponible en: http://www.mic.gov.cu/hinfosoc.aspx , octubre/2007.

[2] Ministerio de Relaciones Exteriores de la República de Cuba, La informatización en Cuba, 2004, disponible en: http://www.cubaminrex.cu/Sociedad_Informacion/Cuba_TIC/Informatizaci%C3%B3n.htm,2004,27/10/2007.

[3] Dónde aprender más sobre SCRUM, 2005, disponible en: http://www.agile-spain.com/agilev2/donde_aprender_mas_sobre_scrum, octubre/2007.

[4] Ídem [3].

[5] Schwaber Ken, SCRUM Development Process, disponible en: http://jeffsutherland.com/oopsla/schwapub.pdf,octubre/2007

[6] Spada Danilo, Usabilidad en el proceso de desarrollo de SCRUM, 26/octubre/2007

[7]Ansueta, SCRUM: metodología "ágil" para tus proyectos,2007 disponible en: http://pymecrunch.com/scrum-metodologia-agil-para-tus-proyectos , octubre/2007

[8] Gracia Joaquín, Gestión de proyectos con SCRUM, 2006 disponible en: http://www.ingenierosoftware.com/equipo/scrum.php, octubre 2007

[9] Duilio J. Protti, El método SCRUM, 2006 disponible en: http://www.exactas.org/index.php?name=Reviews&req=showcontent&id=6 , octubre 2007

[10]Fernández Escribano Gerardo, Introducción a Extreme Programming, Ingeniería de Software II, 2002, disponible en: http://www.info-ab.uclm.es/asignaturas/42551/trabajosAnteriores/Presentacion-XP.pdf , octubre 2007.

[11] Solis Calero Manuel, Una explicación de la programación extrema,2003, disponible en: http://www.willydev.net/InsiteCreation/v1.0/descargas/prev/explicaxp.pdf, octubre 2007.

[12]Anaya Villegas Adrian, A propósito de programación extrema XP, 2006, disponible en : http://www.monografias.com/trabajos51/programacion-extrema/programacion-extrema2.shtml#artef, octubre 2007.

[13] Ídem [12].

[14] Metodologías ágiles para el desarrollo de software, Letelier Patricio y Penadés Mª Carmen, Universidad Politécnica de Valencia, noviembre/2007.

[15] Jacobson Ivar, Booch Grady, Rumbaugh James, Proceso Unificado de Desarrollo, 2000, noviembre/2007.
[16] Ídem [15].
[17] Ídem [15].
[18] Ídem [15].
[19] Ídem [15].
[20] Ídem [15].
[21] Ídem [15].
[22]Ayuda del Rational 2003
[23]Kynetia, Madrid, info@kynetia.com, Metodología de Pruebas, 2007,disponible en : http://www.kynetia.es/calidad/metodologia-de-pruebas.html ,noviembre/2007
[24] Humphrey Watts S, Lecturas "A Discipline for Software Engineering ´´ Capitulo1, 1995, disponible en: http://www.fabricadesoftware.cl/download.php?id=1056&sid=a9f1432477ab803c082d0ae683269a59, noviembre/2007
[25] Humphrey Watts S, Introducción al Proceso Software Personal, 2005, La Habana, Editorial Félix Varela, noviembre/2007
[26] Ídem [25].
[27] Conferencia de gestión de software, noviembre/2007.
[28] Ídem [27].
[29] Ídem [27].
[30] Ídem [27].
[31] Diseño de la interfaz de usuario. Pruebas del software , 2007, disponible en: http://html.rincondelvago.com/diseno-de-la-interfaz-de-usuario_pruebas-del-software.html , noviembre/2007.
*[32]Informática. Control de calidad, disponible en: http://html.rincondelvago.com/prueba-de-software.html , noviembre/2007.*
[33] Pruebas de Software, disponible en:http://lsi.ugr.es/~ig1/docis/pruso.pdf, noviembre/2007.
[34] Pressman Roger S, Ingeniería de Software Un enfoque práctico,2005, La Habana, Editorial Félix Varela, noviembre/2007.

[35]Gutiérrez Javier, Introducción al proceso de Pruebas, disponible en: http://www.lsi.us.es/~javierj/cursos_ficheros/02.SR.pdf, noviembre/2007.

[36] Diseño de Pruebas, Grupo ARQUISOFT - Rojas Johanna - Barrios Emilio – 2007, disponible en: http://www.udistrital.edu.co/comunidad/grupos/arquisoft/fileadmin/Estudiantes/Pruebas/HTML%20-%20Pruebas%20de%20software/node13.html, noviembre/2007.

[37] Aplicación de un Método para Especificar Casos de Prueba de Software en la Administración Pública, Méndez Edumilis, Pérez María, Mendoza Luis E, 2007, disponible en :http://www.sistedes.es/TJISBD/Vol-1/No-4/articles/pris-07-mendez-amecp.pdf, noviembre/2007.

[38] Ídem [37].

*[39] Diseño de sistemas software en UML, disponible en:* http://books.google.com/books?id=X-bReT4PHcMC&pg=PA204&lpg=PA204&dq=pruebas+de+software+patrones+de+pruebas&source=web&ots=5u64pLv2q4&sig=ziqDR2aofrBMxk7sXKQxPnPyQLo#PPA203,M1, noviembre/2008.

[40] Ídem [22].

[41] Pruebas, disponible en: http://lsi.ugr.es/~arroyo/inndoc/doc/pruebas/pruebas_d.php, noviembre/2008

[42] MSc.Y, Fernández, Lic. L Cruz, Ing M González, Ing D, Acosta, 2006, ¨Proceso de pruebas de aceptación para un software de gestión ¨, noviembre/2008

[43]Pruebas de Unidad, Grupo ARQUISOFT - Rojas Johanna - Barrios Emilio, 2007, disponible en: http://www.udistrital.edu.co/comunidad/grupos/arquisoft/fileadmin/Estudiantes/Pruebas/HTML%20-%20Pruebas%20de%20software/node30.html, noviembre/2008

[44] Ídem [34].

[45] Ídem [34].

[46] Ídem [34].

[47]Pruebas de Aceptacion, Grupo ARQUISOFT - Rojas Johanna - Barrios Emilio, 2007, disponible en: http://www.udistrital.edu.co/comunidad/grupos/arquisoft/fileadmin/Estudiantes/Pruebas/HTML%20-%20Pruebas%20de%20software/node56.html , noviembre/2007.

[48]Pruebas de software, Acuña Cesar Javier,
Disponible en: http://kybele.escet.urjc.es/Documentos/ISI/Pruebas%20de%20Software.pdf
[49] Conferencia, Pressman, noviembre/2007.
[50] Ídem [34].
[51] Ídem [34].
[52] Ídem [34].
[53]Pruebas del Software, Letelier Patricio, Departamento Sistemas Informáticos y Computación Universidad Politécnica de Valencia https://pid.dsic.upv.es/C1/Material/Documentos%20Disponibles/Pruebas%20del%20SW.ppt, noviembre/2007.
[54] Ídem [22].
[55] Ídem [53].
[56] Tipos de Pruebas, Kynetia, 2007, disponible en: http://www.kynetia.es/calidad/tipos-de-pruebas.html , noviembre/2007.
[57] Ídem [55].
[58] Pruebas de software, Ingeniería del Software I, Universidad Rey Juan Carlos, César Javier Acuña, disponible en:http://kybele.escet.urjc.es/Documentos/ISI/Pruebas%20de%20Software.pdf, noviembre/2007.
[59] Ídem [55].
[60] Ídem [55].
[61] Ídem [58].
[62] Ídem [34].
[63] Ídem [34].
[64] Pedro Carlos Pérez Martinto, Teoría de Muestreo: población y muestra. Diseño experimental y métodos, disponible en: http://teleformacion.uci.cu/mod/resource/view.php?id=12636,27/2/2008, febrero/2008
[65] Ídem [64].
[66] María Lilia Rodríguez Batista, Propuesta de Procedimiento para el Aseguramiento de la Calidad del Software en los proyectos productivos de la Facultad 7, disponible en: http://bibliodoc.uci.cu/TD/TD_0376_07.pdf , febrero/2008

[67] Ídem [66].

[68] Ídem [25].

[69] Yaimí Márquez Alpízar, Yenni Valdés Hechavarría, Procedimiento General de pruebas de Caja Blanca aplicando la técnica del Camino Básico, disponible en: http://bibliodoc.uci.cu/TD/TD_0638_07.pdf, marzo/2008

[70] Ídem [34].

[71] Ídem [34].

[72] Ídem [34].

[73]Sanz Carmenates, Nadiesda; Pérez Rivas, Geiser Arecio , Pruebas de Aceptación Parciales del Cliente, disponible en: http://bibliodoc.uci.cu/TD/TD_0479_07.pdf,abril/2008.

[74] Ídem [22].

[75] Ídem [22].

[76] Ídem [22].

[77] Ídem [22].

[78] Ídem [73].

[79] Ídem [73].

[80] Ídem [73].

[81] Ídem [73].

# Bibliografía


1. Anaya Villegas Adrian, A propósito de programación extrema XP, 2006, disponible en: http://www.monografias.com/trabajos51/programacion-extrema/programacion-extrema2.shtml#artef, octubre 2007.
2. Ansueta, SCRUM: metodología "ágil" para tus proyectos,2007 disponible en: http://pymecrunch.com/scrum-metodologia-agil-para-tus-proyectos , octubre/2007.
3. Aplicación de un Método para Especificar Casos de Prueba de Software en la Administración Pública, Méndez Edumilis, Pérez María, Mendoza Luis E, 2007, disponible en :http://www.sistedes.es/TJISBD/Vol-1/No-4/articles/pris-07-mendez-amecp.pdf, noviembre/2007.
4. Ayuda del Rational 2003.
5. Beck, K.. "Extreme Programming Explained. Embrace Change", Pearson Education, 1999. Traducido al español como: "Una explicación de la programación extrema. Aceptar el cambio", Addison Wesley, 2000
6. Cockbun, A., Williams, L. "The Costs and Benefits of Pair Programming". Humans and Technology Technical Report. 2000.
7. Conferencia de gestión de software, noviembre/2007.
8. Conferencia, Pressman, noviembre/2007.
9. Desoft, Informatización de la Sociedad, 2002, Ciudad Habana, disponible en**:** http://www.mic.gov.cu/hinfosoc.aspx **,** octubre/2007.
10. Diseño de la interfaz de usuario. Pruebas del software , 2007, disponible en: http://html.rincondelvago.com/diseno-de-la-interfaz-de-usuario_pruebas-del-software.html , noviembre/2007.
11. Diseño de Pruebas, Grupo ARQUISOFT - Rojas Johanna - Barrios Emilio – 2007, disponible en: http://www.udistrital.edu.co/comunidad/grupos/arquisoft/fileadmin/Estudiantes/Pruebas/HTML%20-%20Pruebas%20de%20software/node13.html, noviembre/2007.
12. Diseño de sistemas software en UML, disponible en: http://books.google.com/books?id=X-bReT4PHcMC&pg=PA204&lpg=PA204&dq=pruebas+de+software+patrones+de+pruebas&source=web&ots=5u64pLv2q4&sig=ziqDR2aofrBMxk7sXKQxPnPyQLo#PPA203,M1, noviembre/2008.
13. Dónde aprender más sobre SCRUM, 2005, disponible en: http://www.agile-spain.com/agilev2/donde_aprender_mas_sobre_scrum, octubre/2007.

14. (PDF) Dr. Sutherland, J. (October 2004) Agile Development: Lessons learned from the first scrum.
15. Duilio J. Protti, El método SCRUM, 2006 disponible en: http://www.exactas.org/index.php?name=Reviews&req=showcontent&id=6 , octubre 2007.
16. Fernández Escribano Gerardo, Introducción a Extreme Programming, Ingeniería de Software II, 2002, disponible en: http://www.info-ab.uclm.es/asignaturas/42551/trabajosAnteriores/Presentacion-XP.pdf , octubre 2007.
17. Fowler, Martin, Foemmel M. "Continuous Integration". 2001, www.martinfowler.com/articles/designDead.html .
18. Gracia Joaquín, Gestión de proyectos con SCRUM, 2006 disponible en: http://www.ingenierosoftware.com/equipo/scrum.php, octubre 2007.
19. Gutiérrez Javier, Introducción al proceso de Pruebas, disponible en: http://www.lsi.us.es/~javierj/cursos_ficheros/02.SR.pdf, noviembre/2007.
20. Humphrey Watts S, Introducción al Proceso Software Personal, 2005, La Habana, Editorial Félix Varela, noviembre/2007.
21. Humphrey Watts S, Lecturas "A Discipline for Software Engineering ´´ Capitulo1, 1995, disponible en: http://www.fabricadesoftware.cl/download.php?id=1056&sid=a9f1432477ab803c082d0ae683269a59, noviembre/2007.
22. Informática. Control de calidad, disponible en: http://html.rincondelvago.com/prueba-de-software.html, noviembre/2007.
23. Jacobson Ivar, Booch Grady, Rumbaugh James, Proceso Unificado de Desarrollo, 2000, noviembre/2007.
24. Jeff Sutherland: http://jeffsutherland.com/scrum/.
25. Juan Palacio Bañeres. Gestión ágil de proyectos: Scrum. ppt.
26. Ken Schwaber: http://www.controlchaos.com/.
27. Kynetia, Madrid, info@kynetia.com, Metodología de Pruebas, 2007,disponible en: http://www.kynetia.es/calidad/metodologia-de-pruebas.html noviembre/2007.
28. María Lilia Rodríguez Batista, Propuesta de Procedimiento para el Aseguramiento de la Calidad del Software en los proyectos productivos de la Facultad 7, disponible en: http://bibliodoc.uci.cu/TD/TD_0376_07.pdf , febrero/2008.
29. Metodologías ágiles para el desarrollo de software, Letelier Patricio y Penadés Mª Carmen, Universidad Politécnica de Valencia, noviembre/2007.

30. Ministerio de Relaciones Exteriores de la República de Cuba, La informatización en Cuba, 2004, disponible en: http://www.cubaminrex.cu/Sociedad_Informacion/Cuba_TIC/Informatizaci%C3%B3n.htm,2004,27/10/2007.
31. MSc.Y, Fernández, Lic. L Cruz, Ing M González, Ing D, Acosta, 2006, ¨Proceso de pruebas de aceptación para un software de gestión ¨, noviembre/2008.
32. Pedro Carlos Pérez Martinto, Teoría de Muestreo: población y muestra. Diseño experimental y métodos, disponible en: http://teleformacion.uci.cu/mod/resource/view.php?id=12636,27/2/2008, febrero/2008
33. Pressman Roger S, Ingeniería de Software Un enfoque práctico,2005, La Habana, Editorial Félix Varela, noviembre/2007.
34. Pruebas, disponible en: http://lsi.ugr.es/~arroyo/inndoc/doc/pruebas/pruebas_d.php, noviembre/2008.
35. Pruebas de Aceptación, Grupo ARQUISOFT - Rojas Johanna - Barrios Emilio, 2007, disponible en: http://www.udistrital.edu.co/comunidad/grupos/arquisoft/fileadmin/Estudiantes/Pruebas/HTML%20-%20Pruebas%20de%20software/node56.html , noviembre/2007.
36. Pruebas de Unidad, Grupo ARQUISOFT - Rojas Johanna - Barrios Emilio, 2007, disponible en: http://www.udistrital.edu.co/comunidad/grupos/arquisoft/fileadmin/Estudiantes/Pruebas/HTML%20-%20Pruebas%20de%20software/node30.html, noviembre/2008.
37. Pruebas de software, Acuña Cesar Javier, Disponible en: http://kybele.escet.urjc.es/Documentos/ISI/Pruebas%20de%20Software.pdf
38. Pruebas de Software, disponible en: http://lsi.ugr.es/~ig1/docis/pruso.pdf, noviembre/2007.
39. Pruebas de software, Ingeniería del Software I, Universidad Rey Juan Carlos, César Javier Acuña, disponible en:http://kybele.escet.urjc.es/Documentos/ISI/Pruebas%20de%20Software.pdf, noviembre/2007.
40. Pruebas del Software, Letelier Patricio, Departamento Sistemas Informáticos y Computación Universidad Politécnica de Valencia

https://pid.dsic.upv.es/C1/Material/Documentos%20Disponibles/Pruebas%20del%20SW.ppt, noviembre/2007.

**41.** Rising, L., Janoff, N.S. (2000): http://jeffsutherland.com/oopsla/schwapub.pdf .

**42.** Sanz Carmenates, Nadiesda; Pérez Rivas, Geiser Arecio , Pruebas de Aceptación Parciales del Cliente, disponible en: http://bibliodoc.uci.cu/TD/TD_0479_07.pdf,abril/2008.

43. Schwaber Ken, SCRUM Development Process, disponible en: http://jeffsutherland.com/oopsla/schwapub.pdf,octubre/2007.

44. Solis Calero Manuel, Una explicación de la programación extrema,2003, disponible en: http://www.willydev.net/InsiteCreation/v1.0/descargas/prev/explicaxp.pdf, octubre 2007.

45. Spada Danilo, Usabilidad en el proceso de desarrollo de SCRUM, 26/octubre/2007.

46. Tipos de Pruebas, Kynetia, 2007, disponible en: http://www.kynetia.es/calidad/tipos-de-pruebas.html , noviembre/2007.

47. Yaimí Márquez Alpízar, Yenni Valdés Hechavarría, Procedimiento General de pruebas de Caja Blanca aplicando la técnica del Camino Básico, disponible en: http://bibliodoc.uci.cu/TD/TD_0638_07.pdf, marzo/2008.

48. Wake, W.C. "Extreme Programming Explored". Addison-Wesley. 2002
Fernández Escribano Gerardo, Introducción a Extreme Programming-Ingeniería del Software II, 2002

# Glosario de Términos

**Estándar**: Es una base para la comparación oficialmente aceptada.

**Estándar de codificación**: Es un conjunto de prácticas de codificación aceptadas, las cuales pueden servir como un modelo o guía para los programadores a la hora de escribir el código fuente del sistema que están desarrollando. Dichos estándares son útiles para prevenir defectos.

**Períodos de pruebas**: son actividades encaminadas a realizarles pruebas al software según la etapa (Fase) de desarrollo en la que se encuentre el proyecto.

**Artefactos**: resultados tangibles del proyecto que son producidos, modificados y usados por las actividades. Pueden ser modelos, elementos dentro del modelo, código fuente y ejecutables.

**Ciclo de vida del software**: conjunto de fases que agrupan disímiles actividades con el objetivo de obtener un producto final. Facilita el control sobre los tiempos en que es necesario aplicar recursos de todo tipo (personal, equipos, suministros, etc.) al proyecto.

**Modelo**: es una abstracción del sistema, especificando el sistema desde un punto y un determinado nivel de abstracción.

**Plan de iteración**: define las principales metas, objetivos, motivaciones, medidas, recursos, un horario y entregables de prueba durante cada iteración.