UNIVERSIDAD NACIONAL DE INGENIERÍA

FACULTAD DE INGENIERÍA INDUSTRIAL Y DE SISTEMAS

"PLAN DE ASEGURAMIENTO DE LA CALIDAD EN EL DESARROLLO DE SOFTWARE"

TESIS

PARA OPTAR EL TÍTULO PROFESIONAL DE:

INGENIERO DE SISTEMAS

Carlos Germán Quiroz Malca Muñoz
Luis Alberto Zapata Ruiz

Lima - Perú
1999

Por mi hijo Giancarlito, mi corazón.
Para mi madre Lucila por su apoyo
y a mi padre Demetrio en el recuerdo.
Carlos **Q**uiroz.

A mi madre Orlinda, a quien cumplo sus sueños de hacerme profesional; a mis tías Carmen, Alfira y Gricelda, de quienes recibí apoyo; a mi esposa Carmen, quien me impulsó hacer esto una realidad; a mi hijita Carmin, quien me llena de vida.
Luis **Z**apata.

AGRADECIMIENTOS

A la Casa de estudios que nos proporciono conocimiento, a nuestros Asesores quienes nos aconsejaron y maestros quienes nos dieron sus enseñanzas

Los Autores

Un Agradecimiento especial a Carmen mi esposa, quien me ayudo y empujo en la culminación de este trabajo.

Luis Zapata

INDICE

DESCRIPTORES TEMATICOS

- Calidad del Software
- Aseguramiento de la Calidad del Software
- Aseguramiento de la Calidad en el Desarrollo de Software
- Plan de Aseguramiento de la Calidad del Software
- Plan de Aseguramiento de la Calidad en el Desarrollo de Software
- Mejoramiento continuo del Proceso de Desarrollo de Software
- Factores de Calidad del Software
- Métricas de Calidad del Software

SUMARIO

La Gestión de la Calidad, es una disciplina nueva en nuestro medio que esta tomando creciente importancia en las empresas de nuestro país. Esta situación debido a la globalización y niveles de competitividad de la economía mundial nos compromete ha asumir estos cambios mediante innovaciones constantes en las organizaciones que propicien nuevas formas de enfrentar los retos del Tercer Milenio. Por ello las grandes empresas transnacionales invierten en la búsqueda de la calidad, esto tomando como punto de partida el análisis de la capacidad directiva, control de calidad, calidad de los insumos; así como, un adecuado manejo logístico de las empresas para ser competitivas.

En la Gestión de la Calidad todos los miembros de la organización deben estar involucrados, siendo un esfuerzo totalmente integrado hacia el mejoramiento del desempeño en todos los niveles. Este buen desempeño esta dirigido hacia la satisfacción de metas traducidas en términos de calidad, costos, tiempos, desarrollo del potencial humano y desarrollo de nuevos productos; conduciendo al mejoramiento de los procesos, obtención de productos de calidad y sobre todo a la satisfacción efectiva del cliente.

Debido a la situación descrita anteriormente, planteamos un esquema de trabajo, cuyos procedimientos permitan llevar a cabo un Aseguramiento de la Calidad en el Desarrollo de Software y mejora del Proceso de Desarrollo de Software; de tal manera, que al final del proceso de Desarrollo del Software se pueda tener una confianza en la Calidad del producto final. Por tal motivo, siguiendo el ciclo de vida del Desarrollo del Software proponemos un conjunto de revisiones, entregables, formas de evaluación y métricas que deben ser aplicadas.

Nuestra investigación esta basada en las normas internacionales ISO e IEEE y, modelos para el desarrollo de software como son: Prevención de Defectos, Sala Limpia. Asimismo, hemos considerado conveniente analizar los niveles de madurez que provee el CMM. Cabe mencionar que nuestra investigación se ciñe únicamente al proceso de Desarrollo de Software no estando dirigido a un proyecto en particular. Nosotros, pensamos que puede ser utilizado para todo tipo de desarrollo de software; exceptuando, el desarrollo de software de tipo crítico ya que deberá efectuarse revisiones, inspecciones más profundas y/o detalladas de acuerdo al tipo de proyecto.

Finalmente, esperamos que nuestra investigación permita a la comunidad informática mejorar los procesos en el desarrollo de software y obtener productos con mayor calidad; de tal manera, que pueda ser distribuido e implementado en las empresas contribuyendo con el desarrollo y calidad de los servicios que las mismas brindan al país.

INTRODUCCION

En la actualidad la Calidad del Software esta tomando mayor fuerza debido a los cambios tecnológicos, organizacionales y sobre todo a las altas inversiones que se realizan en desarrollar el software. La Tecnología de Información esta construyendo estos cambios, permitiendo que la información se asimile más rápidamente, mejorando la efectividad de las actividades empresariales. La búsqueda de estos cambios hace indispensable el uso de Sistemas de Información de alta Calidad que ayuden a mejorar el rendimiento y productividad de la organización, y las personas que lo componen. Pudiendo ser logrado con sistemas que cumplan con rigurosa exactitud, confiabilidad, precisión y velocidad.

Es por ello, consideramos necesario las empresas implementen un Plan para el Aseguramiento de la Calidad del Software y obtener software más completos y fiables para lo cual se hace necesario controlar los procesos de desarrollo y habilitar a las organizaciones en la predicción de sus resultados y producir software con características deseadas por los clientes. Esto permitirá alcanzar a las empresas sus metas y objetivos trazados de tal manera que empresas dedicadas al desarrollo de software se conviertan en empresas exitosas.

Capitulo 1

MARCO TEORICO

1.1 Importancia de la Calidad Total

La Calidad Total, integra todos los elementos de una organización: estrategia, política, planificación, sistemas de información, administración de proyectos y toda actividad requerida en las organizaciones de hoy. La capacidad de la Calidad Total juega un papel integrador, enfoca a la organización entera en una dirección, coordinando los planes y acciones, controlando los procesos y mejorando continuamente los negocios.

Existe una sinergia entre la Calidad Total y la Tecnología de la Información, cuando estas se aplican apropiadamente, pueden ayudar a la organización en la reducción de defectos, disminución del tiempo en los procesos, mejorar la seguridad, mejorar el soporte, incrementar la fiabilidad e incrementar el servicio y satisfacción del cliente.

En muchas empresas, los técnicos o jefes aplican tecnología de punta para llegar a ser innovadores sin conocer las metas y objetivos del negocio. Usan la tecnología como la solución instantánea del negocio, incorporando la tecnología a su estrategia y planes corporativos. Esto hace pensar que sin nueva tecnología

como estrategia corporativa la solución es cara y compleja. El peligro que encontramos con la tecnología sino cambiamos la manera de hacer los negocios a un nivel fundamental es que rápidamente hagamos las cosas mal.

La tecnología es cambiante así como los negocios cuyos ejecutivos examinan la forma como compiten y operan sus empresas para tener éxito. La Tecnología de Información esta construyendo estos cambios, permitiendo que la información se mueva rápidamente, aumentando la velocidad en que ocurren las actividades empresariales y el ritmo de trabajo de las personas en dichas organizaciones. Por esta razón los empresarios prestan mayor atención a los procesos comerciales claves de la organización, siendo los Sistemas de Información un recurso clave para mejorar el rendimiento de la organización.

Actualmente, las empresas desarrollan sistemas computacionales que permitan a los consumidores comprar productos / servicios / consultas, como se observa a través de Internet, en que los usuarios utilizan los servicios computacionales de las empresas permitiendo un comercio globalizado. La competencia a escala internacional está enfocada hacia la calidad. La calidad significa la disminución de defectos en los productos, actualmente esta dirigida a la calidad definida por el cliente. Además, las empresas no cuentan con mucho tiempo para desarrollar productos o servicios nuevos y colocarlos en el mercado ya que la competencia es fuerte y el tiempo es un factor esencial.

Dado los niveles de competitividad existentes en los negocios de hoy es indispensable el uso de Tecnología de la Información que ayude a mejorar el rendimiento de las organizaciones y las personas que lo componen. Esto se puede lograr siempre que estos sistemas cumplan con rigurosa exactitud, confiabilidad, precisión y velocidad; por tanto, se hace necesario obtener calidad en el software que componen los Sistemas de Información, para lo cual debemos aprender como aplicar la Calidad Total a la Tecnología de la Información para poder apoyar las estrategias del negocio y mejorar los procesos de las empresas.

1.2 Administración Estratégica basada en la Calidad

La Administración de la Calidad es la integración de todas las funciones y procesos de una organización con el fin de lograr un mejoramiento continuo de la calidad de los bienes y servicios que en ella se producen, siendo el objetivo final la satisfacción del cliente. Este estilo de administración es un enfoque de sistemas que considera todas las interacciones entre los diversos elementos de la organización.

Para la aplicación de este estilo de administración debemos plantear una estrategia basada en Calidad, para lo cual todos los miembros de la organización tendrán que conocer con claridad el concepto, la definición y la medición de la calidad.

1.3 Costo de la Calidad

"La calidad se mide por el costo del desembolso que implica el hecho de no actuar conforme a ella, es decir el costo de hacer las cosas mal" [11]

Philip Crosby
Quality is free

Philip Crosby es autor del libro "La Calidad es Gratuita", sostiene que el logro cero defectos es la norma absoluta del desempeño y que el costo de la calidad es el precio que se paga por el hecho de sujetarse a esta norma.

El costo de la calidad incluye todos los costos acarreados en la búsqueda de la calidad o en las actividades relacionadas con la obtención de la calidad; en términos generales, los costos de la calidad se asocian con la prevención, evaluación y fallos.

- Prevención: Actividades destinadas a suprimir y prevenir los defectos en el proceso de desarrollo, ejm: Planificación de la Calidad, Revisiones Técnicas Formales, Auditorias, Organización del Equipo de ACS.
- Evaluación: Son los costos que se hacen por detectar defectos cuando ya esta fabricado el producto; pero, antes que se distribuya a los clientes, ejm:

Inspecciones en el proceso y entre procesos, Pruebas de Aceptación, Pruebas, etc.

- Fallas Internas: Costos de fallas que se detectan antes de ser distribuido al cliente, ejm: Insatisfacción del Cliente, Servicios después del Servicio, Tiempo Perdido, Reinspecciones, Cambios de ingeniería, etc.
- Fallas Externas: Costos de fallas que se asocian a los defectos una vez enviado el producto al cliente, ejm: Resolución de Quejas, Soporte en Línea, Devolución del Producto, Trabajo de Garantía, etc.

1.4 Factores y Métricas de Calidad del Software

1.4.1 Factores de Calidad del Software

Son las características del software que afectan su calidad y que difieren según la aplicación y los clientes que las solicitan.

El International Standard ISO 9126 Software Product Evaluation define 6 atributos de calidad del software de alto nivel (funcionalidad, fiabilidad, usabilidad, eficiencia, mantenimiento, portabilidad y reusabilidad como factor adicional de importancia), de forma similar lo hace el Instituto de Ingeniería Eléctrica y Electrónica (IEEE). Cada nivel de calidad definido por ISO depende de un número de características o atributos de calidad de bajo nivel que no son mutuamente exclusivos en su impacto en la Calidad del Software.

FACTORES DE CALIDAD DEL SOFTWARE

Factor	Descripción
Funcionalidad	La existencia de ciertas propiedades y funciones que satisfacen las necesidades explícitas o implícitas de los usuarios. Este atributo depende funcionalmente de las propiedades de corrección y extensión por la cual la funcionalidad del sistema es exactamente caracterizado.
Fiabilidad	El grado en que se espera que un programa lleve a cabo sus funciones con la precisión requerida. Una definición más reciente define a la fiabilidad como "la probabilidad de operación libre de fallos de un programa de computadora en un entorno determinado y durante un tiempo especifico". El Código no estructurado representa un factor de alto riesgo para la fiabilidad. Las propiedades de estructuración y encapsulamiento significa reducir riesgos en la fiabilidad.
Usabilidad	Concerniente con la calidad de las interfaces del usuario, su diseño y características de performance. Es una medida del esfuerzo requerido por los usuarios para aprender y usar efectivamente un sistema. Desafortunadamente este factor no es evaluado frecuentemente por los que planifican los proyectos hasta la venta de un paquete, o cuando el uso de su sistema gerencial no cumple los niveles esperados. En este punto, los cambios en el software toman un carácter reactivo, involucrando un esfuerzo adicional y donde el costo subsecuente se incrementa. Las especificaciones pueden ser usadas para evaluar por adelantado la usabilidad de una versión de una aplicación. Sin embargo, esta evaluación es una medida relativa que cambia con la introducción de nuevos métodos de interfaces.
Eficiencia	Es la cantidad de recursos de computadora y de código requeridos por un programa para llevar a cabo sus funciones, bajo condiciones estables.
Mantenimiento	Se le define como el "esfuerzo requerido para localizar y arreglar los errores y hacer los cambios necesarios en un sistema existente". El mantenimiento de un producto existente puede incrementar impresionantemente sus costos de desarrollo. Hay ampliamente la creencia que el software que es fácil de mantener es software de alta calidad. El control efectivo del desarrollo de software hacia los objetivos de calidad establecidos requiere un tiempo adicional de desarrollo, sin embargo éste se verá compensado después por la reducción en el tiempo de mantenimiento.
Portabilidad	Es una medida de la habilidad de una aplicación para operar confiablemente en variados y diferentes sistemas. Los tres principales aspectos que afectan la portabilidad son: dependencia de la maquina, dependencia de la compilación, y dependencia de los sistemas operativos. Hay dos estrategias que pueden ser usadas para minimizar el impacto de la portabilidad: uso efectivo de parametrización, modularización y aislamiento de las dependencias del compilador/sistema/lenguaje. Un programa bien diseñado deberá tener la capacidad de poder ser transferido desde una configuración de hardware a otra. Esto es especialmente cierto, dados los rápidos cambios inherentes en el entorno actual del hardware. Por esta razón la portabilidad se ha convertido en un formidable negocio para los creadores de software, especialmente en grandes aplicaciones del ámbito empresarial.

SUB - FACTORES DE CALIDAD DEL SOFTWARE

Factor	Sub-Factor	Descripción
Eficiencia	Economía del Tiempo	Capacidad del software para ejecutar funciones especificas bajo condiciones explícitas o implícitas dentro de apropiados limites de tiempo.
	Economía de los Recursos	Capacidad del software de ejecutar funciones especificas bajo condiciones explícitas o implícitas dentro de apropiadas cantidades de recursos.
Funcionalidad	Completo	Grado por el cual el software posee las funciones necesarias y suficientes para satisfacer las necesidades del usuario.
	Corrección	El grado en que un programa satisface sus especificaciones y consigue los objetivos de la misión encomendada por el cliente. El Software deberá permitir registrar los datos, los que serán procesados, obteniéndose los resultados correctamente.
	Seguridad	Grado por el cual el software puede detectar y prevenir información errada, información perdida, uso ilegal y destrucción de recursos del sistema.
	Integridad	Grado por el cual se preservan los datos de alteraciones maliciosas o accidentales de usuarios no autorizados. La integridad se refiere al nivel de control sobre accesos no autorizados a los datos dentro de un sistema.
	Compatibilidad	Grado por el cual el nuevo software puede ser instalados sin cambiar ambientes y condiciones que fueron utilizados por el software a ser reemplazado.
	Inter- Operabilidad	Grado por el cual el software puede ser conectado fácilmente con otros sistemas y operado. Dado los constantes avances en la tecnología, los negocios frecuentemente encuentran la necesidad de complementar sus sistemas existentes, requiriendo así que éstos produzcan salidas, resultados, que sean válidos para los sistemas recién adquiridos.
	Intra - Operabilidad	Es la característica de poder comunicarse con otras aplicaciones que operen sobre el mismo sistema operativo. La intra–operabilidad, por sí misma no abarca ni incluye calidad de software, afecta la percepción del usuario sobre la aplicación y por consiguiente tiene un efecto real en la percepción del usuario sobre la calidad del producto.
Mantenimiento	Correctibilidad	Grado de esfuerzo requerido para corregir errores en el software y atender las quejas de los usuarios.
	Expandible	Grado de esfuerzo requerido para mejorar o modificar la eficiencia o funciones del software. Los dos aspectos principales a tener en cuenta para que el software sea expandible son: Diseño Simplificado y Modular.
	Evaluable	Grado de esfuerzo requerido para probar el software de forma que se asegure que realiza su función requerida, asegurar su confiabilidad, y poder detectar donde y cuando falla.

SUB - FACTORES DE CALIDAD DEL SOFTWARE

Factor	Sub-Factor	Descripción
Portabilidad	Independencia del Hardware	Grado por el cual el software no depende del ambiente de hardware
	Independencia del Software	Grado por el cual el software no depende del ambiente de software
	Instalable	Grado de esfuerzo requerido para adecuar el software al nuevo ambiente.
	Reusable	El grado en que un programa (o partes de un programa) se pueden reusar en otras aplicaciones. Para que un módulo tenga la calidad de reusable debe tener funcionalidad y ser fuertemente acoplado. Un factor que influye en la reusabilidad de una parte del software es la modularidad del diseño. Es un factor que está convirtiéndose rápidamente en un tema muy importante en el diseño de software porque considera la habilidad que debe tener un creador para introducir rápidamente productos nuevos dentro del mercado.
Fiabilidad	No Deficiente	Grado por el cual el software no contiene errores no detectados
	Robusto	El grado por el cual puede manejar circunstancias fuera de lo común. Estas circunstancias incluyen ingresos inapropiados de datos, falla en el hardware y manejo de errores. Un sistema robusto evita la perdida de datos críticos ante las fallas que se presenten. Si el sistema no es robusto será frágil y no podrá contemplar situaciones en el mundo real.
	Permanencia	Este factor tiende a medir la capacidad de la aplicación para mantener funciones críticas, aun cuando fallen los componentes principales. Aunque la operación puede ser afectada por fallas importantes, en muchos casos es imperativo que las fallas menores no paralicen el sistema.
Usabilidad	Entendible	Grado de esfuerzo que el usuario requiere para entender el software
	Fácil de Aprender	Grado de esfuerzo requerido para aprender a manejar el software, trabajar con él, preparar sus entradas e interpretar su salida.
	Operable	Grado por el cual las operaciones del software combina el propósito, ambiente, y las características psicológicas del usuario, incluyendo factores ergonómicos; tales como: color, sonido, forma, etc.
	Comunicable	El grado al cual el software es diseñado en concordancia con las características psicológicas del usuario.

FACTORES DE CALIDAD PARA EL PROCESO

Factor	Descripción
Performance	La Performance del Proceso se refiere a los valores de la característica que observamos cuando medimos atributos de productos y servicios que se obtienen de un proceso. La Performance del Proceso se puede describir de dos maneras: por medir atributos de productos que el proceso produce y por medir atributos del proceso mismo. Valores históricos describen cómo el proceso se ha ejecutado en el pasado.
Estabilidad	Características de procesos y productos que muestran variación cuando se mide sobre tiempo. Ésta variación tiene dos fuentes: - Fenómeno, es natural e inherente al proceso y cuyo resultado es común a todas las medidas de un atributo dado. - Variaciones, que tienen causas asignables que se podían haber prevenido.
Cumplimiento	Mide las razones por las que un proceso no puede ejecutarse cuando se requiere. Estas medidas proveen información del contexto para interpretación y acción de los datos que caracterizan la ejecución del proceso. Si se usó apropiadamente, el cumplimiento mide también advertencias anticipadas de procesos que están fuera de control.
Capacidad	Habilidad de la organización para comprender los procesos que realiza ayudado de herramientas, tecnología, personal. No puede ser medido si no se lleva un control estadístico de los procesos.
Mejoramiento	Habilidad de la organización para asegurar la mejora de sus procesos. Cabe señalar que no solo se necesita tener performance o ser capaz; sino que además, se debe entrenar al personal, adquirir nuevas herramientas, realizar investigaciones, entre otros.

1.4.2 Métricas de Calidad del Software

Según la IEEE, se define métrica como "una medida cuantitativa del grado en un sistema, componente o proceso que posee un atributo dado" [8].

Las Métricas son indicadores que buscan estimar el comportamiento de los factores de la calidad de software, debido a la dificultad de su medición directa. Es difícil desarrollar medidas directas de los factores de calidad. Por tanto, se debe definir un conjunto de métricas para los cuales se desarrollen expresiones matemáticas para cada uno de los factores de calidad.

Las medidas directas generalmente son orientadas al producto, tales como al tamaño, a la velocidad de ejecución, al número de defectos, etc. Además, existen medidas indirectas las que están orientadas a la productividad en el proceso de desarrollo y a la calidad del producto (funcionalidad, calidad, complejidad, eficiencia, fiabilidad, etc.)

Los desarrolladores de software tienen que identificar y construir medidas y parámetros pertinentes para sus propios productos con el fin de imponer la calidad de sus sistemas. Así, la calidad es un componente necesario y reconocido para un exitoso proceso de desarrollo de software, no obstante, es un término dinámico el cual podría asumir diferentes facetas dependiendo del contexto en el cual se le use, uno puede hacer lo posible por maximizar los beneficios de cada factor, para lo cual debe concentrarse en la esencia de estos factores.

Entre las razones por las cuales se deben efectuar mediciones al software tenemos:

1. Obtener un software de Calidad.
2. Una forma de verificar si estamos mejorando.

3. Evaluar la productividad del personal que desarrolla el software.
4. Evaluar los beneficios derivados del uso de nuevos métodos y herramientas de ingeniería de software.
5. Establecer una línea base de estimación.
6. Ayudar a justificar el uso de nuevas herramientas.

Medidas al Producto IEEE

Tipo	Medida	Uso	Conceptual	Requerimientos	Diseño	Implementación	Pruebas	Instalación Verificación	Operación Mantenimiento	Retiro
Errores Defectos Fallos	1. Densidad Fallas	2	▲	▲	▲	▲	▲	▲	▲	▲
	2. Densidad Defectos	3	▲	▲	▲	▲	▲	▲	▲	▲
	3. Perfil del fracaso acumulativo	1	▲	▲	▲	▲	▲	▲	▲	▲
	4. Número de Fallas por ida	0	▲	▲	▲	▲	▲	▲	▲	▲
	7. Seguimiento de los Requerimientos	3		▲	▲				▲	
	8. Indice Defectos	1	▲	▲	▲	▲	▲	▲	▲	▲
	12. Número de Requerimientos en conflicto	2		▲					▲	
	23. Cumplimiento de Requerimiento	1		▲					▲	
Tiempo entre fallas	30. Tiempo entre fallas	3					▲	▲	▲	
Indice de fallos	31. Indice de fallos	3					▲	▲	▲	
Crecimiento de la Fiabilidad & Proyección	10. Indice madurez Software	1	▲	▲	▲			▲	▲	
	18. Fiabilidad ejecución	2					▲	▲	▲	
	21. Nivel de la Pureza del software	1					▲	▲	▲	
	26. Función del crecimiento de fiabilidad	2					▲	▲	▲	
	28. Análisis de Fallas usando lapso de tiempo	3					▲	▲	▲	
	29. Suficiencia de Pruebas	0					▲	▲		
	30. Tiempo entre fallas	3					▲	▲	▲	
	37. Fiabilidad de la Performance del Sistema	2		▲	▲	▲	▲	▲	▲	
	38. Fiabilidad del proceso independiente	0						▲	▲	
	39. Disponibilidad operacional HW-SW	0					▲	▲	▲	
Producto Fallas	14. Medidas de la Ciencia del Software	3				▲			▲	
	22. Estimación numero de fallas restantes	2					▲	▲	▲	
	27. Contador de Fallas Residual	1					▲	▲	▲	
	28. Análisis de Fallas usando lapso de tiempo	3					▲	▲	▲	
	36. Exactitud de pruebas	1					▲			
Integridad Consistencia	5. Cobertura Pruebas modular o funcional	1					▲	▲	▲	
	6. Gráficos Causa - Efecto	2		▲	▲	▲	▲		▲	
	7. Seguimiento de los Requerimientos	3		▲	▲				▲	
	12. Número de Requerimientos en conflicto	2		▲					▲	
	16. Complejidad Ciclomática (Pruebas)	3				▲	▲		▲	
	23. Cumplimiento Requerimientos	1		▲					▲	
	24. Cobertura Pruebas	2		▲	▲		▲	▲	▲	
	32. Documentación y listados	2				▲	▲	▲	▲	
	35. Integridad	2		▲	▲				▲	
	36. Exactitud de la Prueba	1					▲			
Complejidad	13. Número de entradas/salidas por modulo	1			▲	▲			▲	
	14. Medida de la Ciencia del Software	3				▲			▲	
	15. Teoría - Gráfica, Complejidad para arquitectura	1		▲	▲					
	16. Complejidad Ciclomática	3			▲	▲			▲	
	17. Determinación caso de prueba mínima	2				▲	▲		▲	
	19. Estructura Diseño	1			▲				▲	
	25.-Complejidad Flujo información o Datos	1			▲	▲			▲	

Fuente : IEEE Software Engineering, 1994

Medidas al Proceso IEEE

Tipo Medida	Medida	Uso	Conceptual	Requerimientos	Diseño	Implementación	Pruebas	Instalación Verificación	Operación Mantenimiento	Retiro
Control Gestión	4. Número de Fallas por día	0	▲	▲	▲	▲	▲	▲	▲	
	8. Indice Defectos	1		▲	▲	▲	▲	▲	▲	
	9. Error Distribución	1		▲	▲	▲	▲	▲	▲	
	11. Horas hombre por defectos detectados	2		▲	▲	▲	▲	▲	▲	
	20. Tiempo en descubrir la prox. K falla	3					▲	▲	▲	
Cobertura	5. Cobertura Pruebas modular o funcional	1					▲	▲	▲	
	6. Gráficos Causa Efecto	2		▲	▲	▲	▲		▲	
	7. Seguimiento de los Requerimientos	3		▲	▲				▲	
	12. Número de Requerimientos en conflicto	2		▲					▲	
	23. Cumplimiento Requerimientos	1		▲					▲	
	24. Cobertura Pruebas	2		▲	▲	▲	▲	▲	▲	
	29. Suficiencia de Pruebas	0					▲			
	32. Documentación y listados	2				▲	▲	▲	▲	
	33. Fiabilidad Requerida del Software	1								
	35. Integridad	2		▲	▲	▲	▲	▲	▲	
	36. Exactitud de la Prueba	1					▲			
Riesgos, Beneficios Evaluación Costos	5. Cobertura Pruebas modular o funcional	1					▲	▲	▲	
	10. Indice madurez Software	1	▲	▲	▲			▲	▲	
	11. Horas hombre por defectos detectados	2		▲	▲	▲	▲	▲	▲	
	14. Medida de la Ciencia del Software	3				▲			▲	
	18. Fiabilidad ejecución	2					▲	▲	▲	
	19. Estructura Diseño	1			▲					
	20. Tiempo en descubrir la prox. K falla	3					▲	▲	▲	
	21. Nivel de la Pureza del software	1					▲	▲	▲	
	27. Contador de Fallas Residual	1					▲	▲	▲	
	31. Indice de fallos	3					▲	▲	▲	
	33. Fiabilidad Requerida del Software	1	▲	▲	▲	▲	▲	▲	▲	
	34. lectura del release de Software	0					▲	▲		

Fuente : IEEE Software Engineering, 1994

Matriz de Clasificación de Medidas IEEE

Medida	Medidas al Producto							Medidas al Proceso		
	Uso	Errores Defectos Fallos	Tiempo entre fallas Indice de fallos	Crecimiento de Fiabilidad & Proyección	Fallas restantes del Producto	Integridad Consistencia	Complejidad	Control Gestión	Cobertura	Riesgo, Beneficio, Evaluación Costo
1. Densidad Fallas	2	▲								
2. Densidad Defectos	3	▲								
3. Perfil del fracaso acumulativo	1	▲								
4. Numero de Fallas por día	0	▲						▲		
5. Cobertura Pruebas modular o funcional	1					▲			▲	▲
6. Gráficos Causa Efecto	2					▲			▲	
7. Seguimiento de los Requerimientos	3	▲				▲			▲	
8. Indice de Defectos	1	▲						▲		
9. Distribución del Error	1							▲		
10. Indice madurez Software	1			▲						▲
11. Horas hombre por defectos detectados	2							▲		▲
12. Número de Requerimientos en conflicto	2	▲				▲			▲	
13. Número de entradas/salidas por modulo	1					▲	▲			
14. Medida de la Ciencia del Software	3				▲		▲			
15. Teoría Gráfica, Complejidad para la Arquitectura	1						▲			
16. Complejidad Ciclomática (Pruebas)	3					▲	▲			
17. Determinación caso de prueba mínima	2					▲	▲			
18. Fiabilidad ejecución	2			▲						
19. Estructura Diseño	1						▲			
20. Tiempo en descubrir la prox. K falla	3									▲
21. Nivel de la Pureza del software	1			▲						
22. Estimación numero de fallas restantes	2				▲					
23. Cumplimiento Requerimientos	1	▲				▲			▲	
24. Cobertura Pruebas	2					▲			▲	
25. Complejidad Flujo información o Datos	1						▲			
26. Función del crecimiento de fiabilidad	2			▲						
27. Contador de Fallas Residual	1				▲					
28. Análisis de Fallas usando lapso d tiempo	3			▲	▲					

Matriz de Clasificación de Medidas IEEE

Medida	Medidas al Producto							Medidas al Proceso		
	Uso	Errores Defectos Fallos	Tiempo entre fallas Indice de fallos	Crecimiento de Fiabilidad & Proyección	Fallas restantes del Producto	Integridad Consistencia	Complejidad	Control Gestión	Cobertura	Riesgo, Beneficio, Evaluación Costo
29. Suficiencia de Pruebas	0			▲					▲	
30. Tiempo entre fallas	3		▲	▲						
31. Indice de fallos	3		▲							
32. Documentación y listados	2					▲				
33. Fiabilidad Requerida del Software	1								▲	▲
34. Lectura del releaseSoftware	0									▲
35. Integridad	2					▲				
36. Exactitud de la Prueba	1				▲	▲			▲	
37. Fiabilidad de Ejecución del Sistema	2			▲						
38. Fiabilidad del proceso independiente	0			▲						
39. Disponibilidad operacional HW-SW	0			▲						

Fuente: IEEE Software Engineering 1994

1.4.3 Proceso de Medición

"La medición es el proceso en el que se cuantifica los atributos de las cantidades del mundo real. Esta medida de atributos internos del producto le proporcionan al desarrollo de software una indicación en tiempo real de la eficacia del análisis, del diseño, de la estructura el código, la efectividad de los casos de pruebas y la calidad global del software"[8].

Los principios básicos de medición se pueden caracterizar en cinco actividades:

- Formulación: Obtención de medidas y métricas apropiadas para la representación de software en cuestión, de acuerdo a criterios de medición que soportan las metas del negocio.
- Colección: mecanismo empleado para acumular datos necesarios para obtener las métricas.
- Análisis: El cálculo de las métricas y la aplicación de herramientas automáticas.
- Interpretación: Evaluación de los resultados de las métricas en un esfuerzo por conseguir una visión interna de la calidad de la representación.
- Retroalimentacion: Recomendaciones obtenidas de la interpretación de métricas técnicas transmitidas al equipo de software.

Se debe identificar los factores críticos que determinan si o no tendremos éxito en cumplir nuestras metas, estos factores críticos son frecuentemente relacionados con los riesgos del negocio.

Las metas del negocio sirven para identificar y enfocar las mediciones necesarias para cuantificar el estado y la ejecución del proceso de software. Estas actividades de medición deben ser diseñadas para soportar las metas del negocio de la organización y proporcionar en

forma efectiva, económica y práctica información para la toma de decisiones.

1.5 Estándares, Metodologías y Normas para la Calidad del Software

1.5.1 Ciclo de Vida del Desarrollo Clásico

El ciclo de vida exige un enfoque sistemático, secuencial, del desarrollo del software que comienza en el nivel del sistema y progresa a través del análisis, diseño, programación, prueba y mantenimiento.

El paradigma clásico del ciclo de vida tiene un lugar definido e importante en el desarrollo de software y proporciona una forma para los métodos de análisis, diseño, programación, prueba y mantenimiento. Además, veremos que los pasos del paradigma clásico del ciclo de vida son muy similares a los pasos que se aplican a otros paradigmas de la ingeniería del software. El ciclo de vida clásico permanece como el modelo procedimental más ampliamente usado por los desarrolladores de software. A pesar de sus inconvenientes, es significativamente mejor que desarrollar sin rumbo el software.

El proceso de desarrollo del software contiene tres fases genéricas, independientemente del paradigma de ingeniería (o combinación) elegido. Las tres fases, definición, desarrollo y mantenimiento, se encuentran en todos los desarrollos de software, independientemente del área de aplicación, tamaño del proyecto o complejidad. Estas fases deben complementarse con documentación de cada uno de los pasos seguidos para asegurar que toda la información sobre el sistema y software estará disponible para su uso posterior.

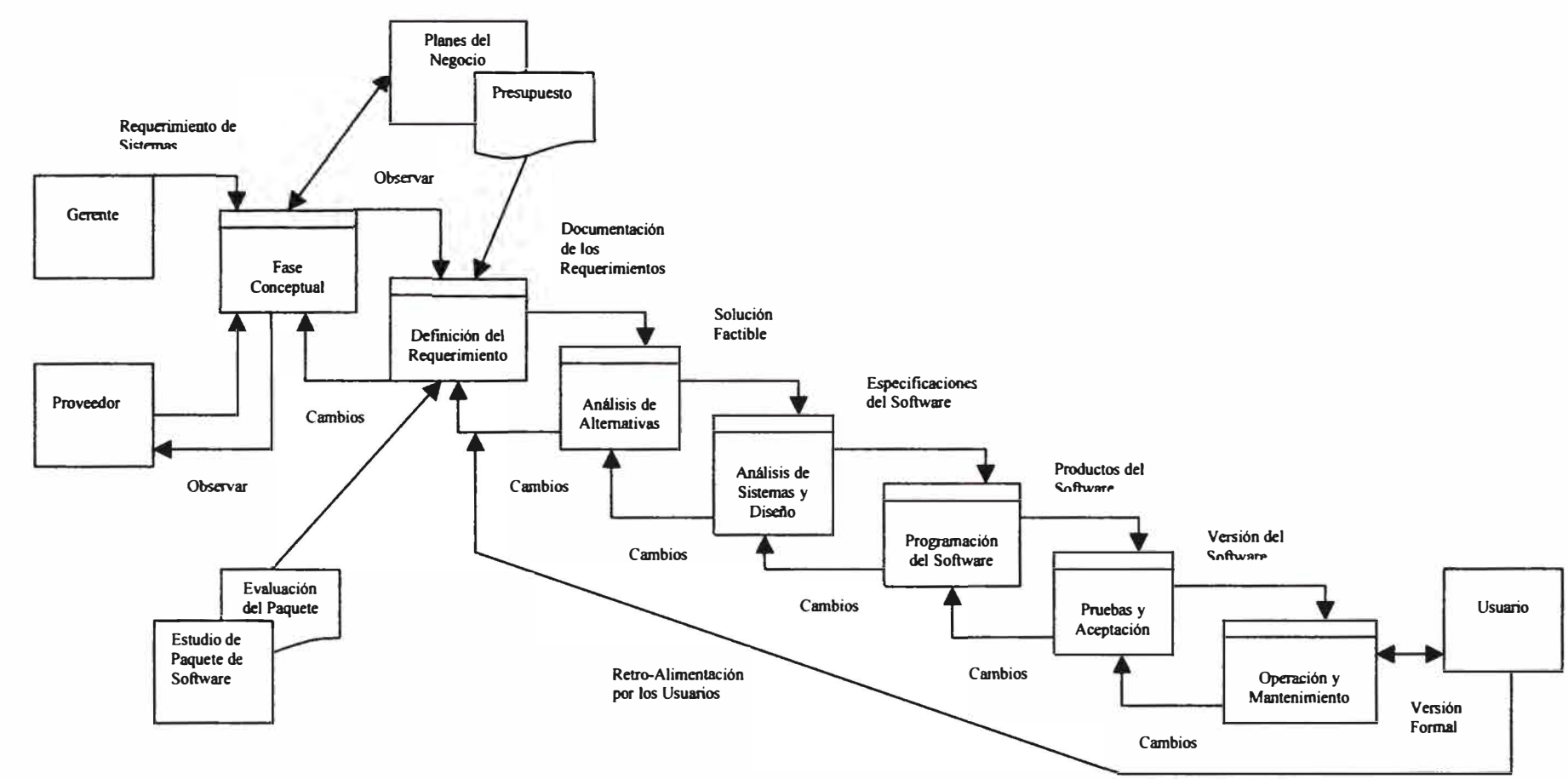


Fig. Proceso genérico del Desarrollo de Software [12]

1.5.2 Normas de Calidad ISO

Las Normas de Calidad ISO 9000 establecen los requisitos que deben cumplir un Sistema de Calidad para poner de manifiesto la capacidad de un proveedor para dar satisfacción a su cliente, cumpliendo con unos procedimientos especificados en la ejecución de los distintos procesos.

Los requisitos establecidos en esta norma tienen fundamentalmente por objeto evitar que se produzcan productos o servicios no conformes durante la fabricación, o ejecución de los mismos o, que si estos se producen, se detecten antes de la entrega del cliente, y la implantación de los medios necesarios para conseguir estos objetivos.

Las Normas ISO 9000 son un conjunto de normas editadas por el Comité ISO / TC 176 "Gestión de la Calidad y Aseguramiento de la Calidad" pertenecientes a Organización Internacional de Estandarización (ISO)

Las normas más representativas son:

- **ISO 9000 – 3 (93):** Quality management and quality assurance standards - Part 3: Guidelines for the application of ISO 9001to development, supply and maintenance of software.
- **ISO 9001 (94):** Quality systems - Model for quality assurance in design, development, production, installation and servicing.
- **ISO 9002 (94):** Quality systems - Model for quality assurance in production, installation and servicing.
- **ISO 9003 (94):** Quality systems - Model for quality assurance in final inspection and test.
- **ISO 9126 (95):** Software Product Evaluation.

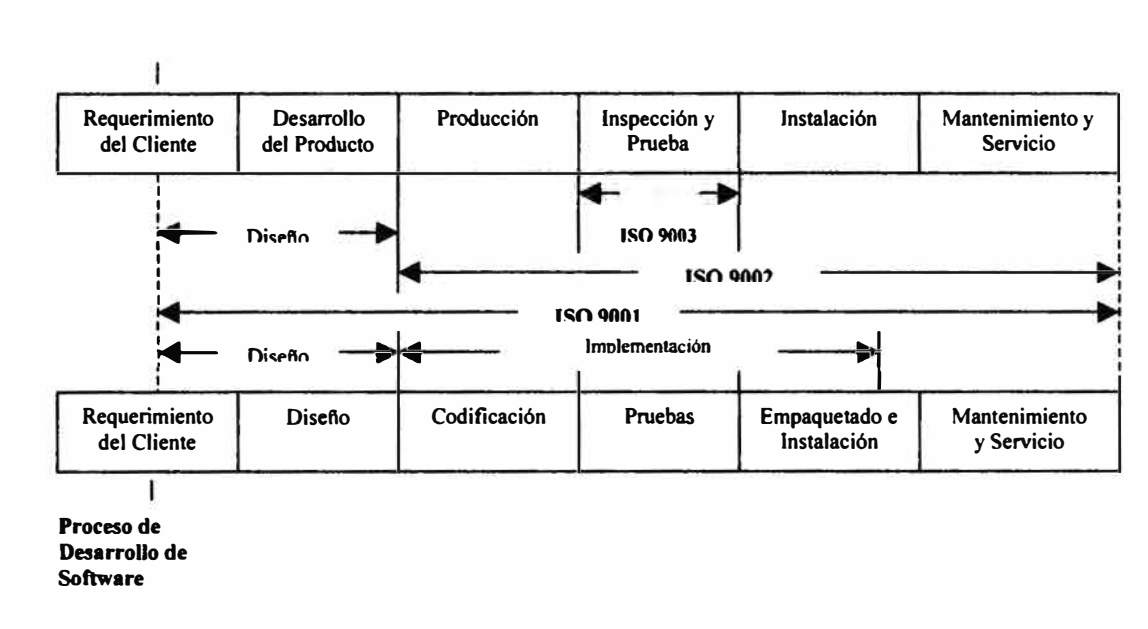


Fig. Relación de los Estándares ISO 9000 con el proceso de Desarrollo [13]

1.5.3 Normas de Calidad de IEEE

El propósito de estas normas es proveer un mínimo de requerimientos uniformes aceptables para la preparación y contenido de Planes para el Aseguramiento de la Calidad del Software.

Estas normas se aplican al desarrollo y mantenimiento de software crítico. Para software no crítico o por software desarrollado, se aplica un sub- conjunto de los requerimientos de estas normas.

Las normas más representativas son:

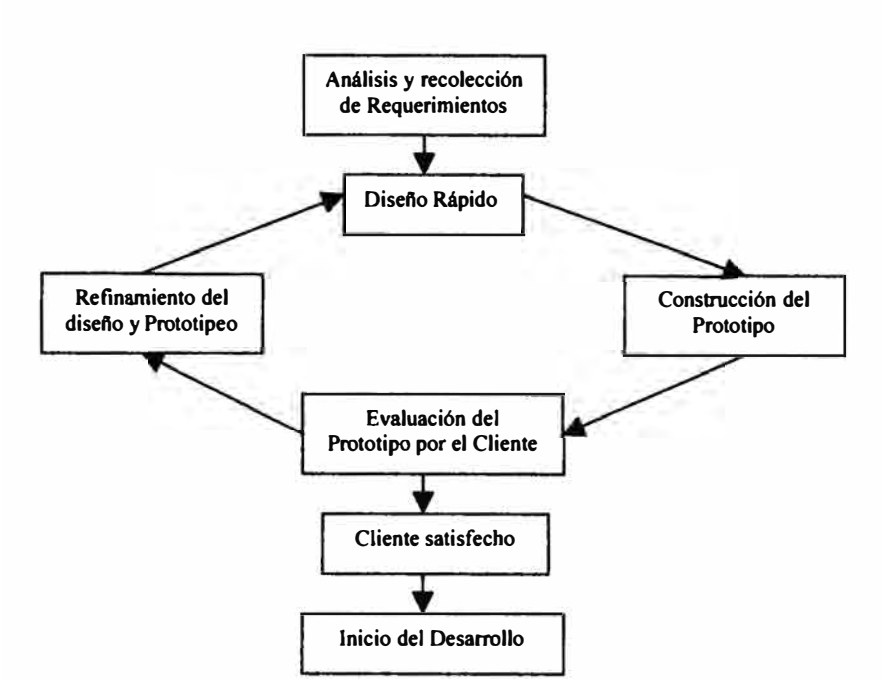
- **730-1998:** IEEE Standards for Software Quality Assurance Plans
- **730.1-1995:** IEEE Guide for Software Quality Assurance Planning Revision and Re-designation of IEEE STD 983-1986
- **730.2-1995:** IEEE Guide for Software Quality Assurance Planning
- **828-1998:** IEEE Standards for Software Configuration Management Plans
- **829-1998:** IEEE Standards for Software Test Documentation
- **830-1998:** IEEE Guide for Software Requirements Specifications

- **982.2-1988:** IEEE Guide for the Use of IEEE Standard Dictionary of Measures to Produce Reliable Software.
- **1008-1993:** IEEE Standard for Software Unit Testing
- **1044-1993:** IEEE Standard Classification for Software Anomalies
- **1045-1992:** IEEE Standards for Software Productivity Metrics
- **1061-1998:** IEEE Standards for a Software Quality Metrics Methodology

1.5.4 Prototipos

El proceso de elaboración de Prototipos en el desarrollo del software fue creado por Gomaa y Scott (1981) y se usan actualmente de manera extensa en la industria, sobre todo en el desarrollo de aplicaciones, usados normalmente en ambientes de alto riesgo o con elementos del sistema donde el desarrollo no tiene una comprensión profunda del problema. En esta aproximidad "rápida y sucia" se construyen prototipos verificados con los clientes y se continúa hasta lograr un prototipo satisfactorio, refinándose y se vuelve a verificar. En que el tiempo es un factor a considerar para comenzar con el desarrollo.

En la evolución del prototipeo se construye un prototipo basado en algunos requerimientos conocidos y comprensibles. Considerando que los prototipos son fáciles de elaborar son usados en aspectos del sistema que se entienda bien, siendo esto una fortaleza para el equipo del desarrollo ya que se puede clarificar los requerimientos del cliente y su interpretación. Estos prototipos se basan en la prioridad de los requerimientos, normalmente se expresan en forma lógica o física mostrando todas las interfaces externas. Para aplicaciones complejas no es razonable, ni barato elaborar prototipos para el desarrollo.

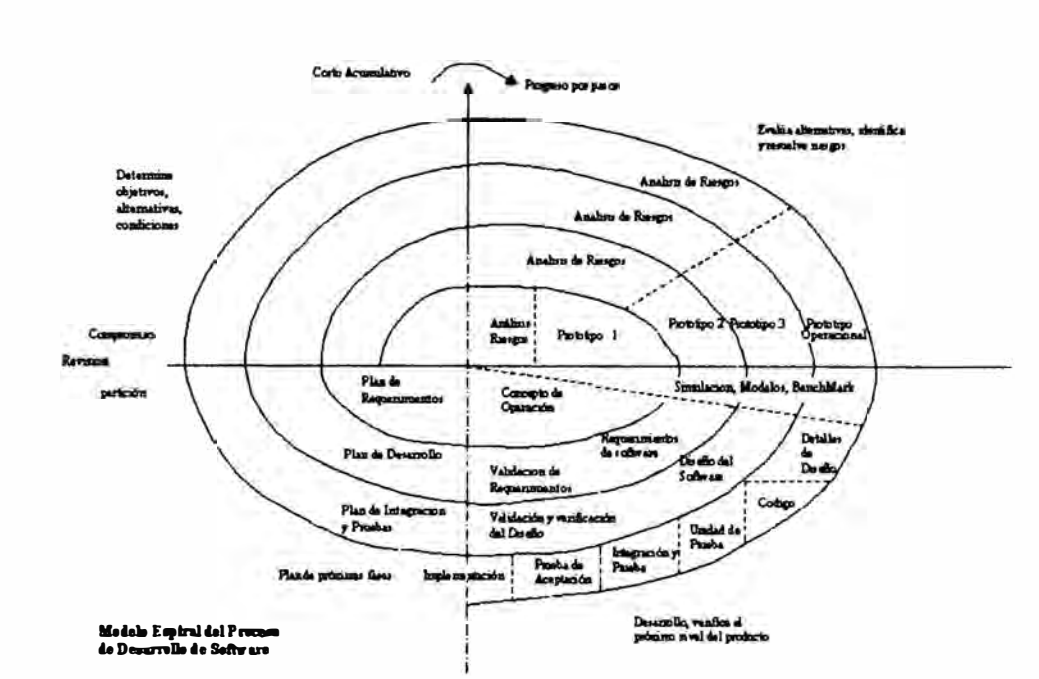


1.5.5 Modelo del Espiral

El Modelo Espiral para el desarrollo y perfeccionamiento del software, fue desarrollado por Boehm (1988), se basó en la experiencia con refinamientos sucesivos del modelo en cascada aplicado a proyectos de software del gobierno de los EEUU. Contar con prototipos y una buena administración de riesgos, es más flexible que el modelo en cascada.

El concepto del modelo se encuentra en cada porción del producto y en cada nivel de elaboración, efectuándose la misma sucesión de pasos (ciclo). El Análisis del riesgo y la administración de riesgos son características importantes del modelo espiral versus el manejo de documentos del modelo en cascada. La administración de riesgos y el prototipeo son herramientas importantes. Usualmente se aplica prototipeo a los elementos del sistema o las alternativas que tienen alto riesgo. Se pueden obtener prototipos que no satisfacen y cuando un prototipo satisface los requerimientos, la implementación puede empezar. Además del prototipeo, el modelo espiral también usa simulaciones, modelos y benchmarks para alcanzar una mejor alternativa. Finalmente, un rasgo importante del modelo espiral, con otros modelos, es que cada ciclo, se

completa luego de una revisión por los miembros claves de la organización involucrados con el producto.



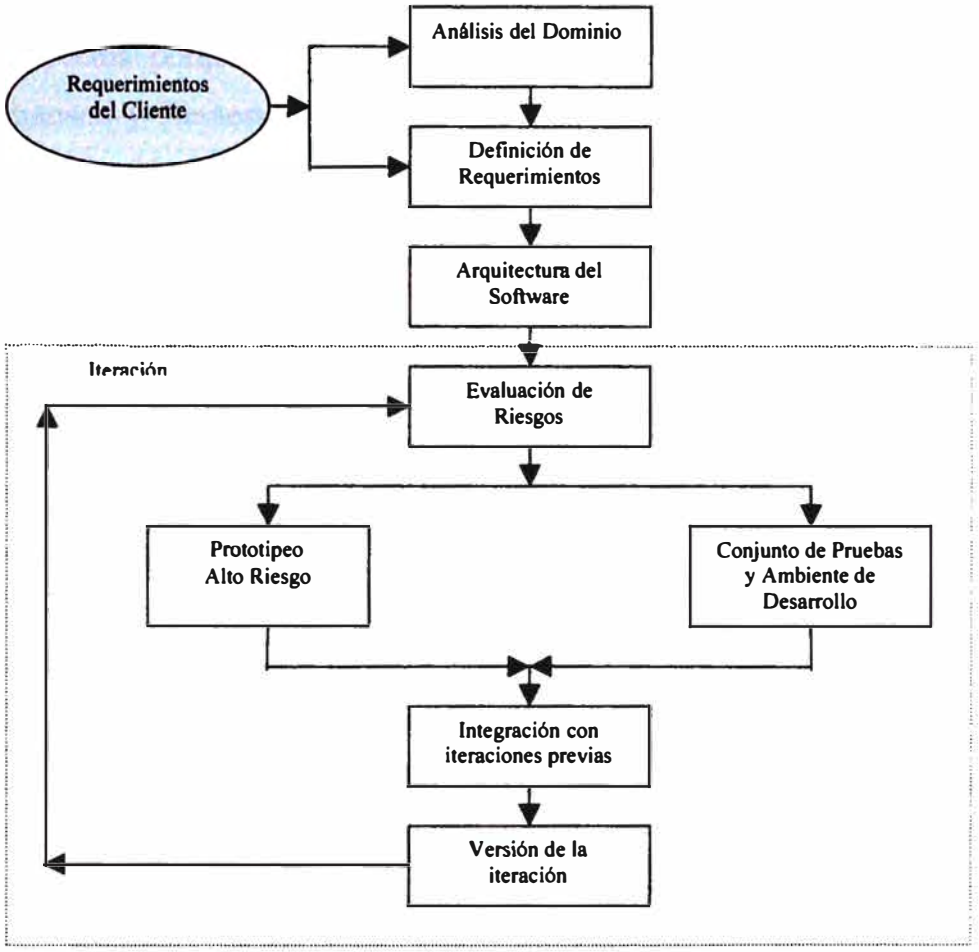
Modelo Espiral del Proceso de Desarrollo de Software

1.5.6 Modelo de Proceso de Desarrollo Iterativo

El Proceso de Desarrollo Iterativo, fue definido para comenzar con un sub- conjunto de requerimientos y desarrollar un sub- conjunto del producto que satisface las necesidades esenciales de los usuarios, provee una herramienta para el análisis y entrenamiento de los clientes, y un aprendizaje de la experiencia por el desarrollador. Basado en el análisis de cada producto intermedio, el diseño y los requerimientos son modificados sobre una serie de iteraciones para proveer un sistema a los usuarios que se desenvuelvan de acuerdo a las necesidades del cliente con mejoras del diseño basadas en la retro- alimentación y aprendizaje.

El Modelo de Proceso de Desarrollo Iterativo combina el prototipeo con el esfuerzo del clásico modelo en cascada. Otros métodos tales como el análisis y análisis de riesgos se pueden incorporar en modelo del proceso. El modelo tiene mucho en común con el modelo en espiral, sobre todo con respecto al prototipeo y manejo de riesgos. El Modelo Espiral se

puede considerar como un Proceso de Desarrollo Iterativo específico, mientras que el Proceso de Desarrollo Iterativo es un modelo más general sobre el cual pueden existir variantes del modelo. El modelo también provee una estructura para muchos sistemas modernos, métodos y técnicas de la ingeniería del software tal como, Desarrollo Orientado a Objetos y Prototipeo Rápido.



1.5.7 Proceso de Desarrollo Orientado a Objetos

El Proceso Orientado a Objetos para el diseño y la programación, se introdujo en los 80s, representa el mayor paradigma de cambio en el desarrollo del software, cuyo efecto en el cambio continuara por muchos años. Diferente de la programación tradicional, que separa datos y control, en la Programación Orientada a Objetos un objeto es un conjunto de datos definidos y un conjunto de operaciones (métodos) que pueden

ejecutar esos datos. Como el paradigma de diseño estructural y descomposición funcional, el Proceso Orientado a Objetos llegará a ser la mayor piedra angular en la ingeniería del software. Aunque hay una tendencia creciente en la Programación Orientada a Objetos (OOP) en la industria, a pesar de la gran cantidad de métodos de análisis y diseño desarrollados y documentados, existe poca información disponible acerca del proceso de desarrollo Orientado a Objetos. Muchas de las discusiones y metodologías existentes pueden ser implementadas en pequeños proyectos porque el proceso de desarrollo no siempre lo necesita. Branson y Herness (1992) propusieron un proceso OPP para grandes proyectos, este se centra sobre una metodología de ocho pasos soportado por un mecanismo de rastreo, una serie de inspecciones, un conjunto de tecnologías, y reglas de prototipeo y prueba.

1.5.8 Método Lógico de la Sala Limpia

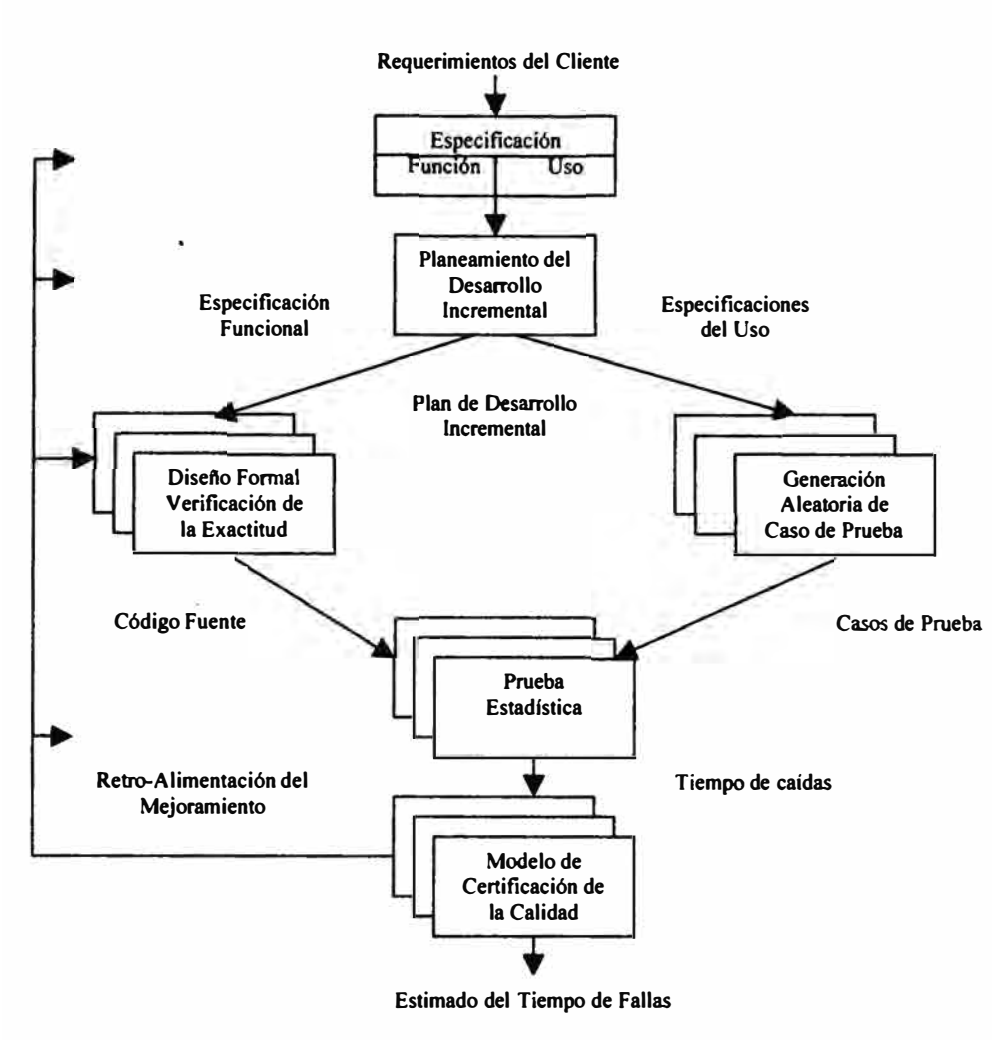
La Ingeniería del software de la Sala Limpia sobre el desarrollo de software es un proceso de ingeniería con fundamentación matemática en vez de un proceso de programación prueba - error (Linger y Hausler, 1992). El proceso de la Sala Limpia emplea teoría basada en tecnologías; tales como: estructura de la especificación de la función del usuario y arquitectura de objetos del sistema, función - teórica prueba del diseño y exactitud, y el uso de la estadística en las pruebas para la certificación de la calidad. La administración de la Sala Limpia se basa en el Desarrollo Incremental y certificación de la línea usuario - función aumentados en el producto final. Las operaciones de la Sala Limpia son llevadas a cabo para desarrollos pequeños, desarrollos independientes y certificación (Prueba) de equipos, con equipos de proyectos grandes.

El proceso de la Sala Limpia da énfasis a la importancia del equipo de desarrollo sobre el control intelectual del proyecto. Las bases del proceso son pruebas de exactitud (del diseño y la codificación) y certificación formal de la calidad vía pruebas estadísticas. Quizás el aspecto más polémico de la Sala Limpia es el equipo de pruebas de la exactitud,

comprueba la unidad de manera individual e independiente. Una vez que el código este desarrollado, está sujeto a la comprobación estadística para evaluar la calidad. Esto motiva tremendamente a los desarrolladores en entregar para las pruebas un código que se encuentre libre de error en su primera ejecución.

El proceso de la Sala Limpia proclama que la prueba estadística puede reemplazar las prueba de cobertura y de camino. En la Sala Limpia, todas las pruebas se basan anticipadamente al uso del cliente. Se diseñan pruebas de los casos de funciones usadas con mayor frecuencia. Por eso, los errores que causan frecuentes fallas a los usuarios se hallan primero. En cuanto a medida, la calidad del software se certifica en términos de veces de fallas.

El proceso de la Sala Limpia representa uno de los procesos formales en el desarrollo del software que ha empezado a ser aplicado en industria. La adopción de la Sala Limpia se emplea principalmente a proyectos pequeños. Como otros métodos formales, las preguntas acerca de su habilidad se descascara en proyectos grandes y el entrenamiento matemático requerido es cuestionado por muchos desarrolladores y gerentes de proyectos. No es sorprendente, que en algunos proyectos de Sala Limpia no evitan la práctica de los métodos tradicionales.



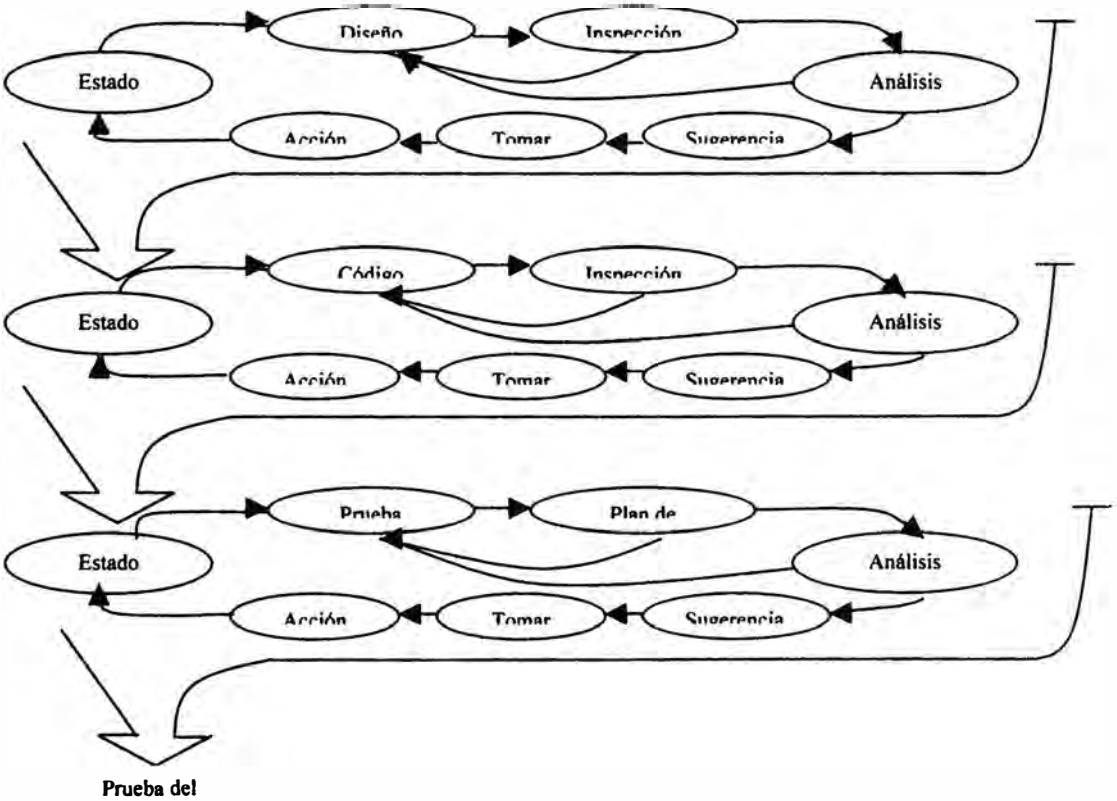
1.5.9 Proceso de Prevención de Defectos

El Proceso de Prevención de Defectos no es un proceso de desarrollo del software. Más bien, es un proceso de mejoramiento continuo del proceso de desarrollo. Se originó en el ambiente del desarrollo del software y se ha llevado a cabo principalmente dentro de organizaciones que desarrollan software.

El Proceso de Prevención de Defectos es un proceso en tiempo real, integrado en cada fase del proceso de desarrollo. Se incorpora en sub-procesos y fases dentro del proyecto. Se enfoca en acciones relacionadas a defectos y procesos orientados a acciones preventivas, lo cual es muy importante. Por la acción de los equipos y acción de herramientas y metodologías de rastreo, el Proceso de Prevención de Defectos provee un sistemático, objetivo, mecanismo basado en datos para la implementación de las acciones. Es un proceso de abajo ha arriba;

reuniones del análisis causal son conducidas por desarrolladores sin interferencia del administrador. De cualquier modo, el proceso requiere el apoyo de la administración y la participación directa vía la acción de los equipos.

Se puede aplicar el Proceso de Prevención de Defectos a cualquier proceso de desarrollo, al de Cascada, Prototipeo, Iterativo, Espiral, Sala Limpia y otros. Cuanto más defectos se registren, se puede ejecutar el análisis causal y trazar acciones preventivas e implementarlas. El principal rol del Proceso de Prevención de Defectos es en el proceso de mejoramiento del software reconocido por la comunidad del software. Finalmente, aunque se ha comenzado el proceso e implementación en los ambientes de desarrollo de software, se puede aplicar a cualquiera producto o industria.

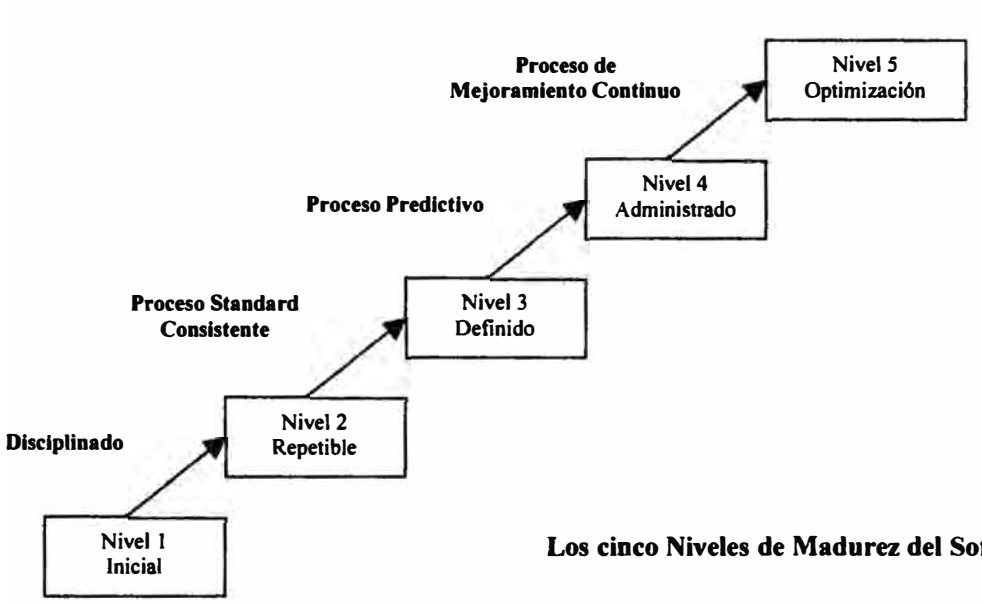


1.5.10 Modelo de la Capacidad de Madurez (CMM-SW)

El Software Engineering Institute (SEI) de la Universidad Canergie-Mellon desarrolló una estructura del proceso de madurez para el

desarrollo de software (Humphrey, 1989). La estructura SEI de evaluación de la madurez ha sido usada por agencias del gobierno y empresas de la industria del software en los EEUU. La estructura se usa con una metodología de evaluación, la cual cuenta con un cuestionario de 85 preguntas. Por cada pregunta, el SEI nivel de madurez asocia la pregunta con un indicador. Hay también preguntas especiales designadas como claves para cada nivel de madurez específico.

Nivel de Madurez	**Areas de Proceso Crítico**
Nivel 1 - Ninguno Desarrollo depende del esfuerzo individual	Ninguno
Nivel 2 - Repetible Utilización de procesos de gestión básicos para controlar costos, calendarios y funcionabilidad	1) Administración de Requerimientos 2) Planeamiento de Proyectos de Software 3) Control y Supervisión de Proyectos de Software 4) Administración de la Subcontratación de Software 5) Aseguramiento de la Calidad del Software 6) Gestión de Configuración del Software
Nivel 3 - Definido Proceso de SW es documentado, estandarizado y se integra dentro de un proceso de software de toda la Organización	1) Enfoque del Proceso de Organización 2) Definición del Proceso de la Organización 3) Programas de Entrenamiento 4) Administración de Integración del Software 5) Ingeniería del Producto de Software 6) Coordinación entre Grupos 7) Revisiones Periódicas
Nivel 4 - Administrado Recopilación de mediciones detalladas del proceso y de la calidad del producto	1) Administración Cuantitativa del Proceso, 2) Administración de la Calidad del Software,
Nivel 5 - Optimización Mediante resultado cuantitativo del proceso de SW, se utilizan criterios de mejoramiento	1) Prevención de Defectos 2) Administración de los Cambios Tecnológicos 3) Administración de los Cambios en los Procesos



Los cinco Niveles de Madurez del Software

1.5.11 Evaluación de SPR

La Software Productivity Research Inc (SPR), desarrolló el método de evaluación SPR, al mismo tiempo SEI desarrolló el modelo del proceso de madurez. Hay un alto grado de similitud y algunas diferencias substanciales entre ambos métodos. Algunos de los principales diseñadores de software usan ambos métodos concurrentemente. Mientras las preguntas de SEI se enfocan a la estructura de la organización del software y el proceso de software, las preguntas de SPR cubren ambas y adicionalmente la estrategia corporativa y las tácticas del proyecto que afecten la calidad, productividad y satisfacción del usuario. Comparando las 85 preguntas de SEI, el número total de preguntas en el SPR es un cuestionario de aproximadamente 400 preguntas. Además, las preguntas del SPR son preguntas de tipo enlace - múltiple - opción con cinco opciones para responder, mientras que el SEI usa un método binario (sí/no). El resultado de la evaluación del proceso por el SPR se expresa por un método de cinco opciones:

1. Excelente
2. Bueno
3. Promedio
4. Marginal
5. Pobre

Diferente a los cinco niveles de la madurez de SEI, el cual tiene criterios pre- definidos, las preguntas de SPR están estructuradas para que una evaluación de "3" sea aproximadamente el promedio para el tópico a ser explorado. El SPR ha desarrollado también una herramienta de software automatizada (CHECKPOINT) para evaluación, planificación de recursos y proyección de la calidad.

Con respecto a calidad del software y métricas, los tópicos a los que se dirigen las preguntas de SPR son:

Medidas de Calidad y Productividad.

Pre- prueba, experiencia en eliminar los defectos entre programadores.

Prueba, experiencia en eliminar los defectos entre programadores.

Calidad del Proyecto y fiabilidad de los Ingresos.

Pre- prueba, eliminar los defectos hasta el nivel del proyecto.

Proyecto de Pruebas, para eliminar los defectos.

Post-release, defectos ha eliminar.

1.5.12 Evaluación de Malcolm Baldrige

El Malcolm Baldrige National Quality Award (MBNQA) es el más prestigioso premio de calidad otorgado en los Estados Unidos. Se estableció en 1988 en la Sección de Comercio de los U.S. (fue nombrado después Secretaria Malcolm Baldrige), se da el premio anualmente para reconocer a las empresas americanas que tiene una mejor administración y realización de la calidad.

Los criterios del examen están divididos en siete categorías los que contienen 28 puntos:

Dirección

Información y análisis

Planificación de la calidad estratégica

Utilización del recurso humano

Aseguramiento de la calidad de productos y servicios

Resultados de la calidad

Satisfacción de los clientes

El sistema tiene un examen para cada uno de los siguiente tres aspectos: proximidad, desarrollo, y resultados. Cada elemento requiere información relativa a por lo menos una de estas dimensiones. La Proximidad, se refiere a los métodos que la empresa tiene para alcanzar sus propósitos dirigidos. Distribución, se refiere a la magnitud en que se aplica la proximidad. Resultados, se refieren a los resultados y efectos alcanzados en los propósitos dirigidos y aplicados.

El propósito de la evaluación Malcolm Baldrige es acerca (los elementos del examen y su evaluación) de cinco pasos:

1. Ayuda a elevar las normas de calidad y expectativas en el país.
2. Facilita la comunicación y comparte entre y dentro de organizaciones de todos los tipos basados en una comprensión común de requerimientos claves de calidad.
3. Sirve como una herramienta de trabajo para la planificación, entrenamiento, evaluación, y otros usos.
4. Provee la base para crear el premio.
5. Provee retro- alimentación de los solicitantes.

Hay 1000 puntos disponibles en los criterios del premio. Cada examen otorga un puntaje en porcentaje para cada elemento (rango de 0% a 100%). Un candidato para el Baldrige debe tener un puntaje en el rango del 70%. Se traduciría generalmente como sigue:

Por un elemento del examen de la proximidad, refinamiento continuo de proximidades está en lugar y una mayoría de ellos se une a otro.

Por un elemento del examen del desarrollo, desarrollo ha mejorado todas las áreas del negocio de las mayores empresas tan satisfactoriamente, como muchas áreas apoyan en conseguirlo.

Por un elemento del examen de los resultados, los resultados de las empresas en muchos de sus áreas son entre lo más alto en la industria. Debe haber evidencia también que los resultados son causados por la proximidad.

Mientras el puntaje es importante, lo más valioso de la evaluación es la retro- alimentación, el cual consiste en las fuerzas observadas y (más significantes) áreas mejoradas. No es raro que las grandes empresas reciban cientos de sugerencias para su mejoramiento. Por enfocar en y eliminar las altas prioridades semanales, pueden asegurar para la empresa un mejoramiento continuo.

Para ser el ganador MBNQA, los cuatro elementos básicos de la estructura de criterios del premio deben ser evidentes:

1. Conducción: Por la dirección del mayor ejecutivo.
2. Sistema: el juego de bien - definido y el buen - diseño de procesos para reuniones de la calidad de la empresa y requerimientos en la ejecución.
3. Medida de progreso: los resultados de las medidas de la calidad en - proceso de las empresas (apuntó a mejorar cliente tasa y ejecución de la empresa).
4. Meta: la puntería básica del proceso de la calidad es la entrega del valor del mejoramiento continuo para los clientes.

Capitulo 2

PROPUESTA PARA EL ASEGURAMIENTO DE LA CALIDAD DEL SOFTWARE

2.1 La Problemática en el Desarrollo de Software

"La industria del Software en el mundo genera cerca de US$221 billones de dólares y se han convertido en la más significante fuerza económica de nuestro tiempo" [10].

Sin embargo un estudio reciente elaborado por "Standish Group" confirmó las siguientes estadísticas:

Cerca del 30 % de los proyectos de software son cancelados antes de completar.

Cerca del 70 % del resto deja de entregar las características esperadas.

Estos datos son un reflejo de la problemática del desarrollo del software en los Estados Unidos debido a las siguientes causa:

- Pobre manejo de requerimientos: No tener un entendimiento claro del problema que intentamos resolver.

- Pobre manejo de los cambios: Los cambios son inevitables y raramente entendemos el impacto.
- Pobre Control de Calidad: Se tienen pobre medidas para la calidad del sistema y el mejoramiento del proceso de desarrollo.
- Poco Control de Calendario y Costos: El Planeamiento exacto es la excepción y las expectativas no realistas son la norma.

Alto costo al cometer errores en los requerimientos

Existe una amplia evidencia que el manejo de requerimientos efectivo que conduce ha ahorros en los costos del proyecto. Estudios efectuados por las corporaciones americanas como GTE, TRW y IBM, midieron y asignaron costos a los errores ocurridos en varias etapas del ciclo de vida del proyecto muestran el siguiente resultado:

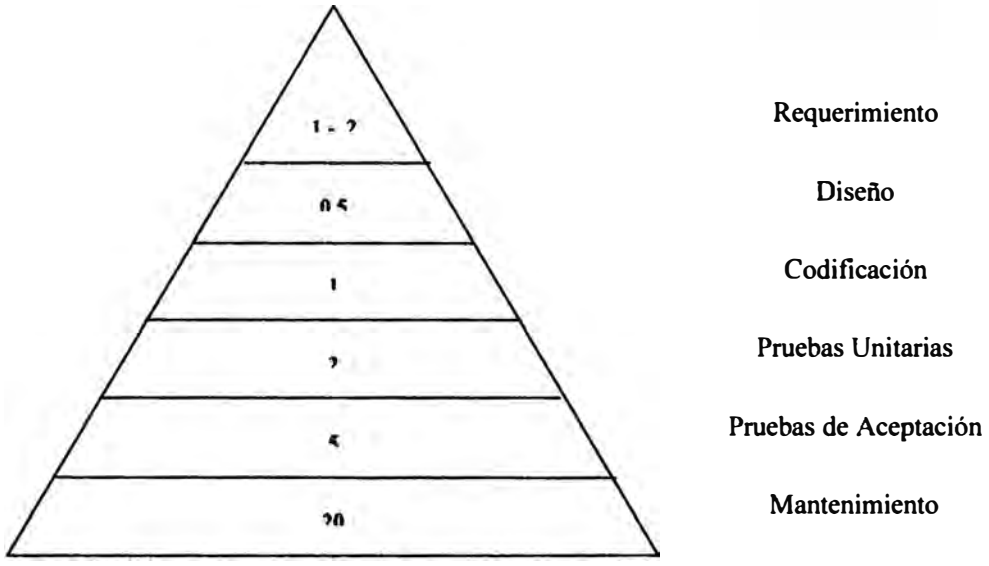
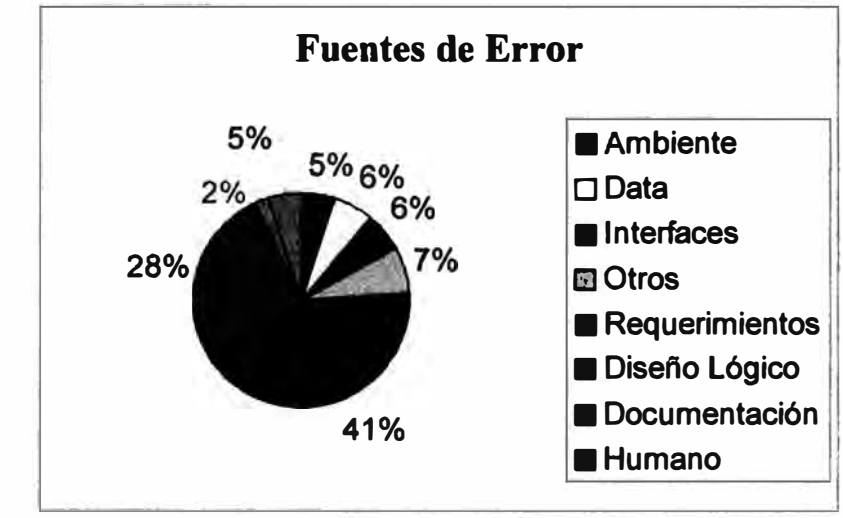


Fig. Costo relativo de reparación de un error

Estos estudios ilustran que los errores en la fase de requerimientos son más baratos de depurar. Un estudio de un gran proyecto de la Fuerza Aérea de USA determina la siguiente fuente de los errores:



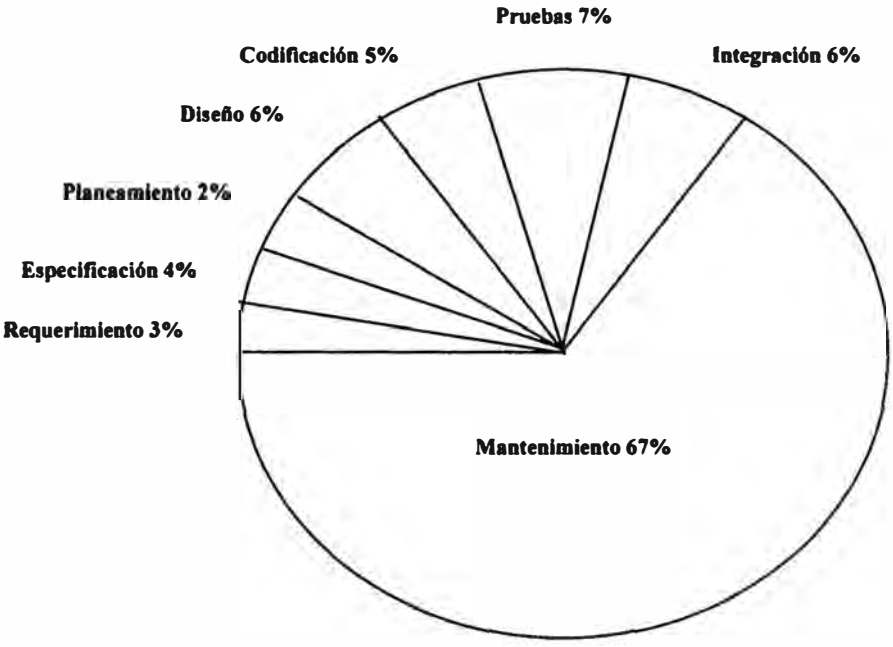


Fig. Necesita planear en el Proceso de Desarrollo de Software

Fuente: Computer Systems Engineering

Tales problemas son manifestaciones visibles de otras dificultades del software:

No hay tiempo de recopilar datos sobre el proceso de desarrollo del software. Sin datos históricos como guía la estimación no es buena y los resultados previstos muy pobres. Sin una indicación sólida de la productividad no

podemos evaluar con precisión la eficacia de las nuevas herramientas técnicas o estándares.

La insatisfacción del cliente con el sistema "terminado" se produce frecuentemente. Los proyectos de desarrollo de software se elaboran frecuentemente con una vaga indicación de los requerimientos del cliente. Normalmente, la comunicación entre el cliente y el que desarrolla el software es muy escasa.

La calidad del software es normalmente cuestionable, se debe dar verdadera importancia a la prueba sistemática y técnicas del software. Están empezando a emerger conceptos cuantitativos sólidos sobre la fiabilidad del software y la garantía de calidad.

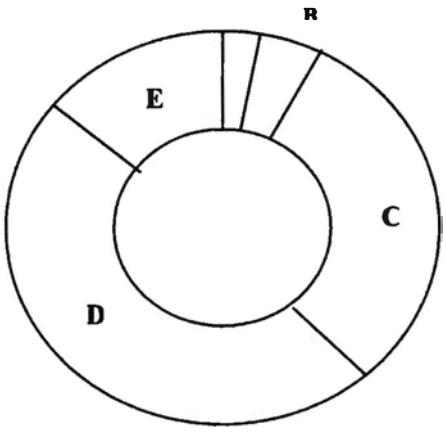
El software existente puede ser muy difícil de mantener. La tarea de mantenimiento de software se lleva la mayor parte de todo el dinero invertido en el software. El mantenimiento no se ha considerado como un criterio importante en la aceptación del software.

El software es un elemento lógico, si se encuentran fallos lo más probable es que se introdujeran inadvertidamente durante el desarrollo y no se detectarán durante la prueba. Reemplazamos las partes defectuosas durante el mantenimiento de software, incluyendo normalmente la corrección o modificación del diseño causados por defectos humanos.

Generalmente, los desarrolladores tienen muy poco entrenamiento formal en las nuevas técnicas de desarrollo de software. En muchas organizaciones cada individuo enfoca su tarea de escribir programas con la experiencia obtenida en trabajos anteriores. Algunas personas desarrollan un método ordenado y eficiente de desarrollo del software mediante prueba y error, pero muchos otros desarrollan malos hábitos que dan como resultado una pobre calidad y mantenibilidad del software.

En muchos casos las viejas actitudes y hábitos son difíciles de modificar, tal que en muchas empresas existen metodologías, procedimientos y estándares que no son utilizados o se utilizan parcialmente por el personal de sistemas.

Se ha hecho muy poco para mejorar la productividad de los desarrolladores de software. Los errores en los nuevos programas producen en los clientes insatisfacción y falta de confianza.



A. Usado como se entregó (2%).

B. Usado después de hacer cambios (3%).

C. Abandonado o Re-trabajado (33%).

D. Entregado pero no Usado (45%).

E. Pagado pero no Entregado (17%).

Fig. La Crisis del Software

2.2 Diagnóstico de la Problemática en el Desarrollo de Software

Los problemas que afligen al desarrollo del software se pueden caracterizar bajo muchas perspectivas diferentes, muchos de los cuales se centran sobre los aspectos de la planificación y estimación de costos, los cuales son muy imprecisos frecuentemente. Por lo observado consideramos dos aspectos importantes a tratar en nuestro análisis:

Análisis del Desarrollo de Software en la Organización

Existen múltiples problemas que se pueden identificar en el desarrollo del software los cuales han sido mencionados en la Problemática del Software (pto. 2.1) y representados en el gráfico Causa- Efecto, los cuales traen como consecuencia altos costos en el mantenimiento del Software, tiempos excesivos en el desarrollo de Software, baja calidad del Software, insatisfacción del cliente- usuario, entre otros.

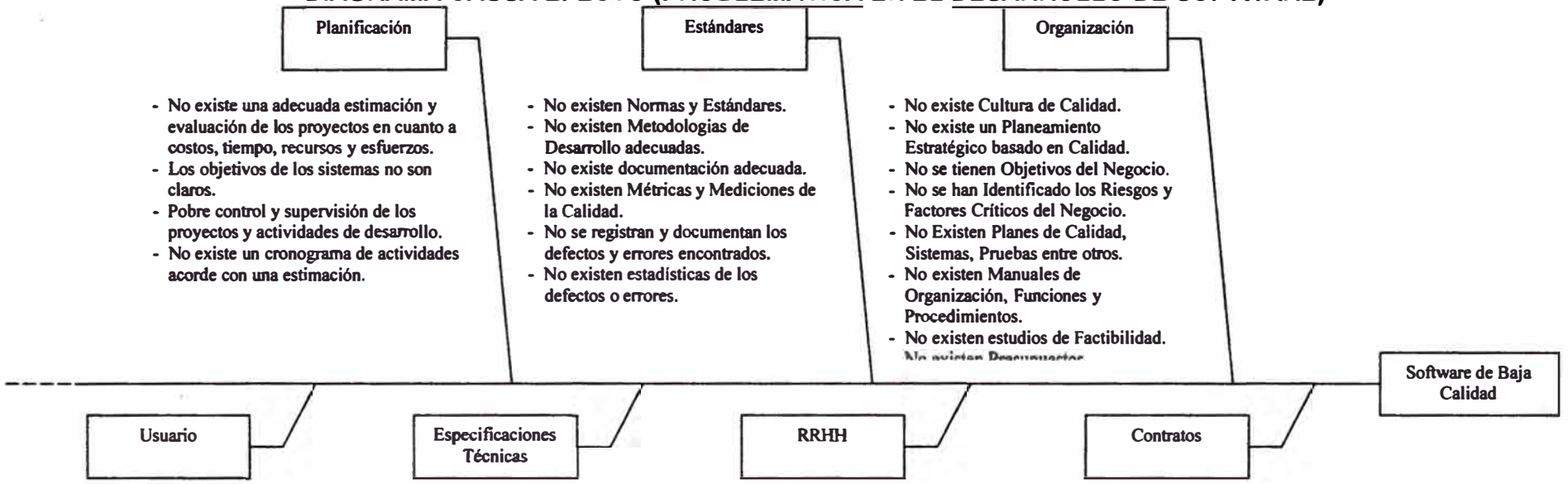
Análisis del Ambiente Externo al Desarrollo de Software

La necesidad de mejorar los negocios demanda que el software tenga mayor calidad, confiabilidad y robustez. La globalización de los negocios permite el

ingreso de Software más competitivo, los cuales ofrecen aplicaciones integradas de mayor calidad (ERP). Las alianzas estratégicas y el rediseño de empresas demandan rápida respuesta por parte del proveedor de Software. Estos son algunos de los eventos que suceden en el entorno de las empresas de nuestro país, las que se ven obligadas a optar por el cambio.

La forma de solucionar esta problemática es aplicar como filosofía de trabajo los conceptos de Calidad Total en el Desarrollo de Software, para obtener la satisfacción del cliente, mejorar los procesos, cumplir con los requerimientos solicitados y obtener un software de calidad.

DIAGRAMA CAUSA EFECTO (PROBLEMÁTICA EN EL DESARROLLO DE SOFTWARE)



Elaboración: Fuente Propia

2.3 Estrategia para el Aseguramiento de la Calidad del Software

Como estrategia consideramos a la Calidad como un factor competitivo importante, el cual debe ser tomado como tema central en la Planificación Estratégica, en el que se defina una misión que acepte como base la Calidad, para lo cual toda organización debe proponerse la siguiente misión:

MISION

Desarrollar productos y servicios de calidad basados en el mejoramiento continuo de los procesos del negocio para la satisfacción del cliente

Esto nos conllevo a trazarnos dos metas importantes para poder concretar nuestra misión, siendo estas:

METAS

- **Asegurar la Calidad del Software**
- **Mejoramiento continuo del Proceso de Desarrollo de Software**

Para cumplir con estas metas planteamos un Plan de Aseguramiento de la Calidad del Software, en el cual desarrollamos una serie de actividades de revisión, inspección, medición y registro de información que nos proporcionen una idea clara del nivel de Calidad del Software e indicadores que nos permitan mejorar el Proceso del Desarrollo de Software.

2.3.1 Asegurar la Calidad del Software

El asegurar la Calidad del Software nos permitirá obtener con mayor facilidad la aceptación del Software por parte del Cliente o Usuario; ya que nuestro Software cumplirá con las características técnicas y de funcionalidad solicitadas por el Cliente o Usuario.

A nuestro criterio los Factores de Calidad que nos permitirán conseguir la Calidad en el Software son:

Corrección

Necesitamos que el software cumpla con exactitud los requerimientos del Usuario o Cliente, para lo cual consideramos conveniente realizar una medida de los defectos del Software. Este factor por si solo no nos permitirá conocer los defectos ya que puede haber cumplido con exactitud con los requerimientos; pero, posiblemente todos los requerimientos del Usuario o Cliente no fueron incorporados al Software.

Completo

Por esta razón necesitamos conocer si todas las funciones en el Software han sido incorporadas. Pero, podremos decir que el Software no llegar a estar completo, si no tiene funciones adicionales que le permitan evitar la perdida de datos críticos, mantener en ejecución sus funciones criticas y lograr manejar sus errores.

Fiabilidad

Por lo mencionado, necesitamos que el Software no contenga errores no detectados, en caso contrario ocasionaría que el Software sea deficiente. El Software deberá saber manejar circunstancias del mundo real, logrando manejar las fallas y errores que se presente, si no lo hiciera aumentaría la fragilidad del Software. El Software no debe cesar en sus funciones inesperadamente o en caso contrario debería recuperarse con facilidad manteniéndose operativo.

Usabilidad

Hasta el momento podemos decir que tenemos un software digno de confianza, pero tenemos que tener en cuenta su uso. El Software debe ser fácil de aprender por un usuario típico y brindarle la ayuda necesaria. El Software no debe molestar al usuario; para lo cual, debe ser consistente en sus funciones y en el diseño de sus interfaces. Adicionalmente, su ejecución no debe ser inusualmente lenta o anómala, no debe tomar el control sin mostrar algún mensaje o indicación al usuario.

Mantenible

Finalmente, necesitamos que el Software sea fácil de mantener y facilite su crecimiento.

Una vez seleccionados los factores se hace necesario conocer como podemos medirlos; para esto, de cada uno de ellos seleccionamos los atributos que podemos medir y asignamos la métrica que corresponde, la cual nos dará una indicación del nivel de calidad para un factor dado. La asignación de atributos y métricas a los factores seleccionados las mostramos a continuación:

Factor de Calidad	Atributos a medir	Métricas	Actividades de medición
Corrección	# defectos	Indice de defectos	Inspección Código Verificación y Validación Análisis Estático Análisis Dinámico
Completo	# Requisitos # Funciones no implementados	Cumplimiento del Requerimiento. Métrica de Calidad de la Especificación.	Revisiones Inspecciones
Fiabilidad	# defectos # fallas # errores	Indice defectos. Numero días de Falla.	Revisiones Inspecciones Verificación y Validación
Usabilidad	Grado de aceptación del Usuario	Indice de aceptación	Check list de Aceptación
Mantenible	# líneas de Código # Cambios # Solicitudes de Cambio Tamaño Complejidad	Complejidad McCabe Ciencia Hasltead Indice de Madurez del Software.	Inspección del Código Fuente Revisión de Releases Revisión de Cambios

2.3.2 Mejoramiento Continuo del Proceso de Desarrollo de Software

Al mejorar los procesos nos aseguramos que los productos obtenidos en cada uno de ellos proporcionen la Calidad que se espera conseguir.

A nuestro criterio los factores que nos permitirán mejorar de manera continua los procesos en el Desarrollo de Software son:

Performance

La Performance de un proceso puede ser cuantificada, esto hace visible la habilidad de un proceso para generar productos con la calidad, tiempo y costos que los Clientes y Negocios exigen. Cuando la Performance de un proceso varia erráticamente y no es predecible en el tiempo, el proceso se hace no controlable originando una variabilidad inestable. Para lograr el control necesitamos que el proceso se haga estable para poder predecir los resultados.

Estabilidad

La Estabilidad es un factor importante en todo proceso, si la ejecución del proceso es errática y no predecible, se debe tomar las acciones del caso para estabilizar el proceso. La Estabilidad de un proceso depende del soporte y su fiel funcionamiento.

Cumplimiento

Para tener un proceso estable y controlable se necesita que sea correctamente ejecutado, se haga uso de herramientas, metodologías, prácticas y planes de trabajo. Por tanto, la organización debe tener la habilidad necesaria para ejecutar con éxito el proceso (Experiencia, habilidades, facilidades, herramientas, procedimientos documentados, etc.), con lo cual el proceso se hará capaz.

Finalmente, las metas del negocio y estrategias, así como los datos que se obtengan acerca de los factores de calidad que inciden sobre el producto y ejecución del proceso respectivamente son la clave que llevan a acciones que mejoran un proceso de software. Pero no sólo se deben identificar los cambios que se necesitan mejorar a un proceso, también se debe observar los beneficios que justifican los costos asociados al cambiar el proceso. Esto a menudo requiere medidas que van más allá de ejecución del proceso ya que debe pensarse en incluir los costos de

entrenamiento, de equipo adicional o nuevas herramientas de programación.

A continuación, de manera similar para el Aseguramiento de Calidad del Software asignamos los atributos y métricas a los factores seleccionados, los cuales se presentan a continuación:

Factores de Calidad	Atributos a medir	Métricas	Actividades Medición
Performance	Esfuerzo empleado Tiempo de CPU utilizado Tiempo Actividades del Proyecto Horas- Hombre empleadas Fuente de defectos detectados, reparados o reportados	Estadística de Defectos por Fase. Métricas de Productividad Niveles del Tiempo de CPU Tiempo de Desarrollo y Mantenimiento	Control Estadístico Revisiones e Inspecciones Análisis Causal
Estabilidad	# defectos de estándares, formatos y procedimientos	Estadística de Defectos	Revisión y Check-list Análisis Causal
Cumplimiento	Grado de Madurez en el Desarrollo Esfuerzo empleado Horas- Hombres empleados	Distribución del Error Nivel de Madurez Niveles de Entrenamiento y Experiencia del Personal Nivel del uso de Herramientas	Revisión y Check-list. Análisis de Riesgos de la Planificación del Proyecto Análisis Causal

Capitulo 3

GESTION Y ADMINISTRACION DE LA CALIDAD DE SOFTWARE

3.1 Propósito

Permitir que la formación de un Equipo de Aseguramiento de la Calidad de Software (ACS) asegure la calidad en el desarrollo software, mediante la ejecución de nuestro Plan en el que se han definido un conjunto de procesos, procedimientos y estándares a ser usados a lo largo de un proyecto, efectuando las mediciones y revisiones requeridas por las personas responsables en cada uno de los procesos.

Objetivos Específicos:

- Organizar, planificar e identificar los componentes del Equipo de ACS en la Organización.
- Definir métodos, revisiones, pruebas en las fases del desarrollo de software, con el propósito de asegurar el nivel de calidad del software.
- Determinar y evaluar factores y métricas de calidad adecuadas en cada punto de revisión.
- Los planes de trabajo de los Proyectos deben contemplar un cronograma de entregables y técnicas de revisión.

- Control de la documentación, en cada fase de desarrollo.
- Procedimiento para ajuste de estándares de desarrollo.
- Utilizar Estándares, Metodología, Modelos en el ACS, reconocidos, tales como: ISO 9001, ISO 9000-3, ISO 9126, CMM - Modelo, SPICE.
- Actividades que permitan efectuar revisiones al Software para verificar que ellos cumplen procedimientos y normas por los que se obtenga la calidad deseada; así como, documentar los resultados de las revisiones que se efectúen.
- Verificar la aceptación del software por parte del Usuario o Cliente de acuerdo a los requerimientos efectuados.

Metas

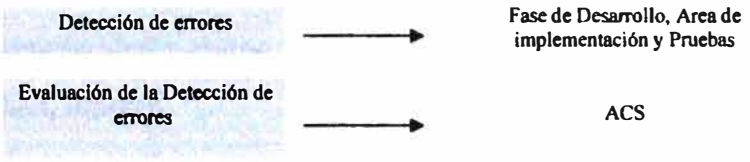
- Planear las actividades para el ACS.
- La adherencia de actividades y evaluación de los productos de software según las normas, procedimientos y requerimientos en forma objetiva.
- Los individuos y grupos afectados se informan de los resultados obtenidos en la revisión o actividades que garanticen la calidad del software.
- Los puntos de incumplimiento que no puedan ser resueltos dentro del proyecto de software son dirigidos al Gerente del Proyecto o Gerente General.

3.2 Funciones del ACS

1. Asegurar la utilización de herramientas y métodos técnicos durante todo el proceso de ingeniería de software.

2. Efectuar revisiones para:
   - Identificar necesidades de mejoras.
   - Identificar donde no se necesita mejoras.
   - Hacer más "manejable" y "uniforme" el trabajo técnico.

3. Prueba del software:



4. Ajustes a los estándares

   Mejoramiento de estándares basado en la incorporación del conocimiento adquirido en su implementación y al mejoramiento continuo de la calidad de software.

5. Control de cambios:

6. Mediciones: para evaluar la calidad actual del software y los efectos de los cambios.

7. Registro y generación de informes

Toda la documentación técnica y de ACS del Software

3.3 Organización del Equipo de ACS

Es necesario conocer la manera cómo la empresa está organizada y administrada en el terreno de la calidad y del control, localizando cualitativamente, los puntos fuertes y débiles que guiarán la acción de la calidad futura. Esta es la razón por la cual deberá implantarse un departamento que gestione y administre todo lo referente a la calidad (para nuestros fines el Software) dentro de una organización.

3.3.1 Departamento para el ACS

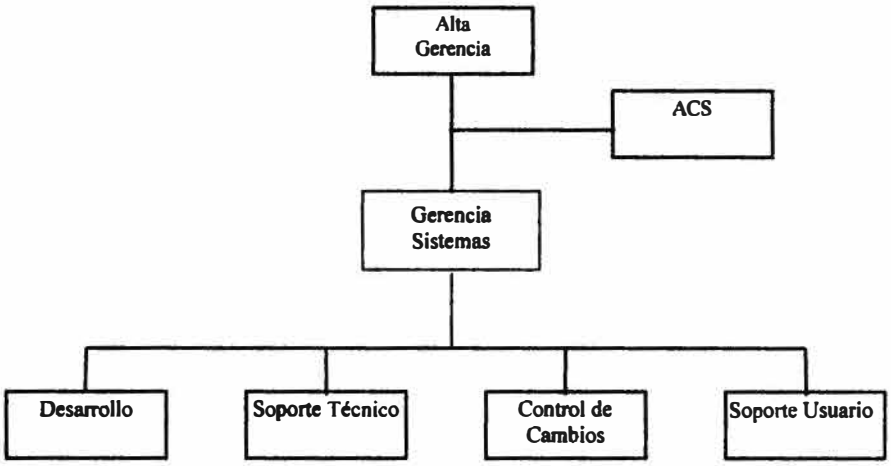
Se identifica como necesaria una unidad especial de ACS, basándonos en los siguientes argumentos:

El tamaño de los sistemas de la institución: grande.

La complejidad y riesgo de los sistemas: en línea, efecto monetario y jurídico, imagen.

El número de Proyectos que simultáneamente se desarrollan: más de 5 Proyectos.

La ubicación del departamento de ACS, se da al nivel Asesor de la Alta Gerencia, manteniendo independencia de los departamentos de Desarrollo, Implementación y Pruebas, Operaciones y Soporte Técnico. Pudiendo ser un ente externo que asesora en cuestiones de calidad a la Alta Gerencia o a la Gerencia de Sistemas.



La organización debe tomar verdadera importancia a la independencia, interpretación y seguimiento, particularmente para proyectos y organizaciones pequeñas. Todo plan de Calidad exige independencia cuando existen aspectos técnicos o sesgos de la organización que afecten la calidad o riesgos asociados con el proyecto. Por ejemplo:

1. El Equipo de ACS tiene un mayor alcance de información; siendo, independiente al del gerente del proyecto, al Equipo de Desarrollo y otros grupos relacionados con el desarrollo del software.
2. Para la aceptación del sistema el grupo de Prueba del Sistema deberá planear los procedimientos de prueba y ser independiente al Equipo de Desarrollo.

Se basa la necesidad de independencia del sistema y comprobación de la aceptación en consideraciones técnicas. Esta independencia asegura que las pruebas no sean inapropiadas o influyan en el plan y decisiones de implementación, como seria si fuese realizado por el Equipo de desarrollo o de Control de Cambios.

La independencia del Equipo de ACS es necesaria, así sus miembros pueden ejecutar sus trabajos sin ser influenciados por el cronograma del proyecto y presiones de costos.

Las Organizaciones deben determinar la estructura organizacional que apoye las actividades que requieren independencia, tal como ACS, en el contexto de sus metas del negocio, estrategias y ambiente del negocio.

Esta independencia debe:

Proveer al personal que ejecutan el ACS la libertad y responsabilidad para asegurar que los requerimientos de calidad sean implementados y mantenidos en el proceso de desarrollo del software.

Proteger al personal que ejecutan el ACS dar apreciación de la ejecución por el manejo del proyecto acerca de los involucrados en el desarrollo del sistema.

Proveer al gerente del ACS confianza en informar los objetivos durante el proceso e informar sobre productos obtenidos en el proyecto.

Tareas

1. El Equipo de ACS cubre las practicas necesarias para ejecutar la Función de Aseguramiento de la Calidad del Software. Las prácticas identifican actividades especificas y evaluación del software de trabajo que el Equipo de ACS revisara.
2. El Equipo de ACS trabaja con el Equipo de Desarrollo de Software durante las primeras fases para establecer planes, normas, y procedimientos que adicionaran valor al proyecto, y satisfacen las restricciones del proyecto y las políticas de la Organización.
3. El Equipo de ACS ayuda a asegurar la adaptación de las necesidades del proyecto y verificar que sean útiles para desempeñar las revisiones a lo largo del Ciclo de Vida del Desarrollo.
4. El Equipo de ACS revisa las actividades del proyecto y el Software a lo largo del Ciclo de Vida, y verifica que el Proyecto de Software se adhiere a los planes establecidos, normas y procedimientos.
5. Los puntos de cumplimiento son primero dirigidos dentro del proyecto de software y resueltos, para puntos no resueltos dentro del

proyecto del software, el Equipo de ACS direcciona el asunto a un nivel apropiado para su resolución.

6. Controlar los defectos encontrados y verificar la implementación de soluciones y acciones correctivas.
7. Controlar el procesamiento, distribución o instalación de productos no conformes hasta que la deficiencia o condición de insatisfacción halla sido corregida.

3.3.2 Equipo de ACS en Ingeniería del Sistema

El Equipo en Ingeniería del Sistema es la colección de individuos (gerentes y personal técnico) quien tiene responsabilidad por la planificación y ejecuta la comprobación del software en forma independiente, determina si el producto satisface sus requerimientos. Ejecuta su labor desde la fase de Planificación del Proyecto hasta la fase de Programación encargado dentro del Plan de Calidad del Software.

3.3.3 Equipo de ACS en Evaluación del Sistema

El Equipo de Evaluación del Sistema es la colección de individuos (gerentes y personal técnico) quienes diseñan e instrumentan las actividades para verificar la calidad del software. Evalúa cada proceso siguiendo normas, desde la fase de pruebas hasta su aceptación por el cliente o usuario, encargado dentro del Plan de Pruebas del Software.

3.3.4 Equipo de ACS en Administración de la Configuración

El Equipo para evaluar la Administración de la Configuración del Software es la colección de individuos (gerentes y personal técnico) tienen la responsabilidad por la planificación, coordinación y llevar a cabo las actividades de verificar la correcta Administración de la Configuración por el proyecto de software, encargado dentro del Plan de Configuración del Software.

3.4 Responsabilidades y Roles del Equipo de ACS

1. **El Equipo de ACS es responsable de coordinar e implementar el Aseguramiento de Calidad para el proyecto.**

   El Equipo de ACS es la colección de administradores, e individuos quienes tienen responsabilidad de un conjunto de tareas o actividades. Cuando se desarrolla o instala un software asignan tareas y actividades, definen el tamaño del proyecto, la estructura organizacional; así como, la cultura organizacional.

2. **El plan de ACS debe ser preparado para cada proyecto de software según un procedimiento documentado.**

   Este procedimiento deberá especificar que:

   - El plan de ACS es desarrollado en las fases iniciales y en forma paralela al proyecto total planificado.
   - El plan de ACS es revisado por los grupos afectados e individuos.

     Ejemplos de grupos afectados:

     El Gerente del proyecto de software.

     Otros gerentes o jefes involucrados en el proyecto.

     El representante del ACS.

     El Gerente General para quien el grupo de ACS reporta incumplimientos.

     El grupo de ingeniería de software (incluye todos los sub-grupos, tales como: diseño y desarrollo de software).

3. **Las actividades del Equipo de ACS son ejecutadas en conformidad con el Plan de Aseguramiento de Calidad del Software.**

   El plan cubre:

   - Responsabilidades y autoridades del Equipo de ACS.
   - Requerimientos de recursos para el Equipo de ACS (incluye personal, herramientas y facilidades).
   - Calendario y financiamiento de las actividades del Equipo de ACS.
   - Participación del Equipo de ACS en establecer el plan de desarrollo de software, normas y procedimientos para el proyecto.
   - Evaluaciones a ser ejecutadas por el Equipo de ACS.

Ejemplos de productos y actividades a ser evaluados son incluidos:

Software operacional y soporte de software,

Productos entregables y no entregables,

Software y documentos,

Desarrollo del producto y verificación de las actividades del producto (ejecutar casos de pruebas), y

Las actividades seguidas en la creación de productos.

- Las auditorias y revisiones son conducidas por el Equipo de ACS.
- Las normas y procedimientos son establecidos como la base para la revisión y auditoría del Equipo de ACS.
- Los procedimientos para documentación y control de incumplimientos a ser resueltos.
- Documentación que el Equipo de ACS requiere que produzca.
- Métodos y frecuencia para proporcionar retro- alimentación al Equipo de Desarrollo y otros grupos relacionados a las actividades de ACS.

4. El Equipo de ACS participa en la preparación y en la revisión del plan de desarrollo de Software, normas, y procedimientos.

- El Equipo de ACS proporciona consultas sobre la revisión de los planes, normas, estándares y procedimientos con respecto al:

  Cumplimiento de la política organizacional.

  Cumplimiento requerimientos externos y normas impuestas que deberán ser utilizados en el proyecto.

  Los temas que deben tenerse en cuenta en el plan de desarrollo de software.

  Las otras áreas asignadas al proyecto.

- El Equipo de ACS verifica que los planes, normas y procedimientos se encuentren almacenados en un lugar seguro para ser usados en futuras revisiones y auditarías del proyecto de software.

5. El Equipo de ACS revisa las actividades de la ingeniería de software para verificar el cumplimiento.

- Las actividades son evaluadas contra el plan de desarrollo de software, las normas y los procedimientos designados.

- Las desviaciones se identifican, documentan y se efectúa un seguimiento de las mismas.
- Las correcciones son verificadas.
- El software de trabajo es evaluado contra las normas de estándares, procedimientos y requerimientos contractuales.
- Los productos entregables del software se evalúan antes que se entreguen al cliente.

6. **El Equipo de ACS periódicamente reporta los resultados de sus actividades al Equipo de Desarrollo.**
7. **Las desviaciones identificadas en las actividades de desarrollo y en el software son documentadas y administradas según un procedimiento documentado.**

   Este procedimiento deberá especificar que:

   - Las desviaciones del Plan de Desarrollo de Software, a las normas del proyecto y los procedimientos son documentadas y resueltas con los respectivos lideres del proyecto, jefe del proyecto o gerentes del proyecto.
   - Las desviaciones del Plan de Desarrollo de Software, a las normas del proyecto y los procedimientos no resueltos con los lideres del proyecto, jefes del proyecto o gerentes del proyecto son documentados y presentados al gerente general o gerente designado para recibir los puntos incumplidos.
   - Los asuntos incumplidos son presentados al gerente general o gerente encargado para ser periódicamente revisados hasta que ellos sean resueltos.
   - La documentación de incumplimientos es administrada y controlada.
8. **Recursos y financiamiento adecuados es proporcionado para ejecutar las actividades de ACS.**
   - A un administrador se le asigna responsabilidades especificas para las actividades del proyecto.
   - Al administrador, quien conoce bien el rol del ACS y tiene la autoridad para tomar apropiadas acciones de vigilancia, es designado para recibir y

actuar sobre el incumplimiento en el software.

- Todos los gerentes en el ACS que informan al gerente general o gerente de sistemas conoce bien el rol del ACS, las responsabilidades, y autoridad.

9. **Herramientas para las actividades de ACS están disponibles.**
   Ejemplos de algunas herramientas, tenemos:
   Estaciones de trabajo,
   Programas de base de datos,
   Programas de hoja de calculo, y
   Herramientas de auditoria.
10. **Los miembros del Equipo de ACS son entrenados para ejecutar las actividades de Aseguramiento de la Calidad de software.**
    Ejemplos de entrenamiento:
    Técnicas y practicas de Ingeniería de Software;
    Roles y responsabilidades del Equipo de Desarrollo y otros grupos relacionados;
    Normas, procedimientos y métodos para el desarrollo de software;
    Dominio de la Aplicación del proyecto de software;
    Objetivos, procedimientos, y métodos de ACS;
    Involucrar al Equipo de ACS en las actividades de software;
    Métodos efectivos y herramientas de ACS;
    Comunicación interpersonal.
11. **Las actividades de verificación deben incluir inspección, prueba y monitoreo del desarrollo del software, diseño de revisiones.**
    Estas actividades deben ser llevadas a cabo por personal independiente de aquellos que tienen responsabilidad directa del trabajo que esta siendo ejecutado.
12. **El Equipo de ACS conduce las revisiones periódicas** de sus actividades y reuniones con el personal del cliente de ACS.
13. **Los miembros del proyecto de software reciben orientación en los roles, responsabilidades, autorizaciones y el valor del Equipo de ACS.**

3.5 Tareas y Responsabilidades de los miembros del Equipo de ACS

3.5.1 Gerente de ACS

Es el representante del cliente; es decir, los miembros de ACS deben mirar el software desde el punto de vista del cliente.

Dirige al grupo que define los estándares.

Organiza y conduce las reuniones recopilando los acuerdos suscitados en ellas para crear y mantener un legajo del proyecto y asegurar que se distribuya entre las personas involucradas en el proyecto (coordinación de fechas, elaborar agendas de reunión, conducir y documentar la reunión y comunicar posteriormente los resultados de la reunión a través de Actas de Reunión de ACS).

Revisa continuamente el cumplimiento de estándares, a través de visitas de inspección, las cuales están dirigidas a revisar los Folders de los Proyectos (ver si la documentación y los procesos establecidos se siguen en forma correcta) y evaluar si cada fase del desarrollo está cumpliendo con los estándares.

Prepara informes de evaluación de la calidad en forma periódica y al finalizar cada etapa del Proyecto.

Propone al Gerente de Sistemas las políticas, procedimientos y estrategias que se requieren para mejorar la calidad de software, así como las medidas correctivas necesarias para su cumplimiento.

3.5.2 Supervisor de ACS

El Supervisor de ACS en el desarrollo del Proyecto, tiene la responsabilidad de planear, coordinar e informar de la ejecución de las tareas asignadas al equipo. Tiene la misma responsabilidad que los otros miembros del equipo respecto a la calidad de su propio trabajo. Reporta y coordina directamente con el Gerente de ACS.

3.5.3 Analista de ACS

El Analista de ACS será el responsable de las revisiones, inspecciones, informa sobre las revisiones y resultados obtenidos. El Analista de ACS

puede dirigir las revisiones y, reporta y coordina directamente con el Supervisor de ACS.

Procedimiento
3.6 Revisión

Se efectúa para validar la existencia de un documento, verificar cumplimientos, detectar defectos, fallas o errores de interpretación con respecto a los requerimientos del negocio, cliente, presentados en la documentación elaborada por el Equipo del Proyecto. En este formato se deberá registrar información de las personas que aprobaron el procedimiento, cambios que se han efectuado y los departamentos en los cuales debe ser aplicado.

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Cambios

Revisión	Fecha	Persona Responsable	Descripción del Cambio
A		Nombre	Revisión Inicial

Lista de Distribución

(lista de los departamentos que reciben copia del procedimiento)

1. Propósito

- Descubrir defectos, errores en la función, la lógica e implementación de cualquier elemento del software.
- Verificar que el Software bajo revisión alcance sus requerimientos.
- Garantizar que el software ha sido representado de acuerdo con ciertos estándares predefinido.
- Conseguir un software desarrollado de forma uniforme.

2. Alcance

La revisión es una actividad de verificación que determina el estado actual del proceso, verifica cumplimientos y satisfacción de requerimientos, determina riesgos potenciales y permite tomar acciones correctivas. Debe ser realizado por el Equipo de ACS en apropiados intervalos de tiempo para asegurar la continuidad en forma satisfactoria y efectiva.

3. Roles

<Inspector> Encargado de dirigir y guiar el equipo de revisión, planificar la reunión, definir roles de los participantes y presentar los elementos a inspeccionar en la reunión. Sugerimos que se asigne el rol a un *<Supervisor de ACS>*.

<Revisor> Encargado de realizar el trabajo de revisión, aplicar lista de comprobaciones y detección de defectos. Sugerimos sea realizado por *<Analista de ACS>*.

<Autor> Es el que origina el producto a inspeccionar. Normalmente son los miembros responsables del desarrollo del proyecto o software.

<Secretario> Encargado del apoyo documentario, elabora lista de observaciones, lista de defectos encontrados en la reunión de inspección y/o alguna información de interés. Sugerimos que se asigne este rol a un *<Analista de ACS>*.

*<Cliente>*Es el representante de la organización que tiene amplio conocimiento de alguna actividad funcional del negocio. Normalmente son los miembros responsables de las áreas usuarias o funcionales en la empresa

4. Procedimiento

4.1 Planificación de la reunión

<El autor> se contactan con un jefe de revisión responsable de las revisiones del proyecto, cuando el producto esta terminado.

<Inspector> evalúa la disponibilidad del producto, genera copias del material si fuera necesario y lo distribuye a los *<Revisores>* para que se preparen por adelantado.

<Inspector> coordinara con las personas involucradas la fecha de revisión, las actividades, asignara roles, seleccionara las métricas de evaluación de acuerdo a la evaluación y prepara la agenda de la revisión.

4.2 Preparación

<Inspector> y el equipo de *<Revisores>* determinan los requerimientos necesarios para realizar la revisión.

4.3 Revisión

<Revisores> reúnen todos los documentos necesarios para la reunión de revisión como (contratos, documentos, programas fuentes, código, etc.) para encontrar los defectos, categorizarlos, registrarlos, pero no resolverlos.

<Revisores> ejecutarán la revisión, basándose en el carácter de la misma y el tipo de revisión.

4.4 Discusión

Luego de obtener los resultados de la revisión los *<Revisores>* coordinan el momento de la reunión de discusión si fuera necesario.

<Inspector> explicará la agenda de la revisión y una breve introducción a cargo del productor.

<Revisor> toma el tiempo necesario adicional para realizar discusiones sobre los riesgos potenciales y defectos encontrados, categorizarlos y registrarlos.

4.5 Reparación

Tiempo que tomará el *<Autor>* para corregir los defectos encontrados durante la revisión. El *<Autor>* recibirá las recomendaciones u objeciones sobre el producto, realizara el trabajo de reparación y coordinará otra fecha de reunión.

4.6 Seguimiento

Revisión del *<Inspector>* para determinar si los defectos han sido corregidos y no se han introducido otros.

Al final de la revisión, los *<Revisores>* deben decidir sí aceptan el producto sin posteriores modificaciones o rechazan el producto.

Tomada la decisión todos los participantes terminan firmando el documento de revisión, quienes deben estar de acuerdo con las conclusiones del equipo de revisión.

5. Documentación relacionada

<Documento de Revisión>.

6. Formatos

<Formato de Revisión> Este formato consta de 2 páginas, deberá ser utilizado desde la fase de elaboración de Planificación hasta la fase de Construcción y sirve para registrar datos generales sobre la revisión efectuada; tales como: nombre del proyecto, encargado de la revisión, resultados, observaciones, comentarios, documentos revisados. El contenido de este formato servirá de referencia para la elaboración de informes sobre las revisiones.

QZ	**Revisión de** _____ Num: ____ Pag: 1/2

Proyecto	Area Responsable de Revisión
Jefe Proyecto	Jefe de Revisión
Area	Fecha de inicio Total Horas Revisión
Etapa	Fecha Termino Total Revisores

Nombre	Cargo	Firma
Nombre	Cargo	Firma
Nombre	Cargo	Firma
Nombre	Cargo	Firma
Nombre	Cargo	Firma
Nombre	Cargo	Firma
Nombre	Cargo	Firma
Nombre	Cargo	Firma
Nombre	Cargo	Firma
Nombre	Cargo	Firma
Nombre	Cargo	Firma
Nombre	Cargo	Firma

Resultado de la Revisión:		Observaciones :
Aceptado	( )	
No Aceptado	( )	
Por Modificar	( )	
Nueva Revisión	( )	

Q Z	**Revisión de _____ Num: ____ Pag: 2/2**
Proyecto _____ Area _____ Etapa	Fecha _____ Hora Inicio _____ Hora Final

Documento Revisados	Observaciones
_____ _____ _____	_____ _____ _____

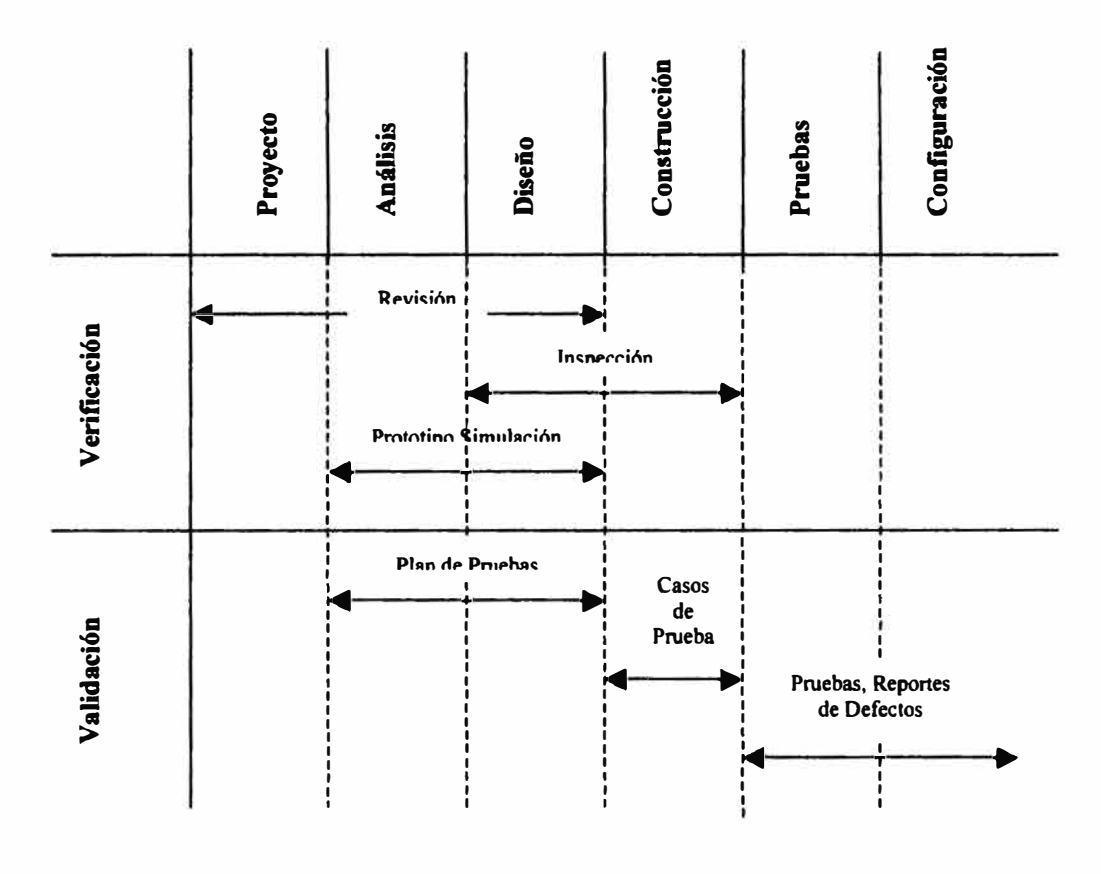


Fig. Proceso de Validación y Verificación

Fuente: Elaboración Propia

Procedimiento
3.7 Medición

Se efectúa para verificar los niveles en que se encuentran los factores de calidad y de productividad del software y del Equipo de Proyecto respectivamente.
En este formato se deberá registrar información de las personas que aprobaron el procedimiento, cambios que se han efectuado y los departamentos en los cuales debe ser aplicado.

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Cambios

Revisión	Fecha	Persona Responsable	Descripción del Cambio
A		Nombre	Revisión Inicial

Lista de Distribución

(lista de los departamentos que reciben copia del procedimiento)

1. Propósito

- Indicar la Calidad del Producto.
- Evaluar la productividad del personal y proyecto.
- Establecer una línea base de estimación.
- Identificar el actual nivel de performance en cada métrica
- Ayudar a justificar el uso de nuevas herramientas.
- Definir un criterio de aceptación de la performance del software, al utilizar herramientas automáticas de medición.

2. Alcance

La medición es una actividad técnica de Aseguramiento de la Calidad que establece los requerimientos de calidad, identifica, implementa, analiza, mide el nivel de calidad, y válida el proceso de desarrollo y del producto con métricas de calidad del software.

3. Responsabilidades

<*Supervisor de ACS*> Encargado de dirigir y guiar el equipo de inspección, y seleccionar la métrica adecuada.

<*Analista de ACS*> Encargado de realizar el trabajo de inspección, ejecución de la métrica, detección de defectos y registro de resultados. Es el responsable del uso y operación de las herramientas automáticas de medición, notificará al <*Supervisor de ACS*> sobre dificultades, observaciones y operatividad de la herramienta utilizada.

<*Autor*> Es el que origina el producto a inspeccionar. Normalmente son los miembros responsables del desarrollo del proyecto o software.

4. Procedimiento

4.1 Verifica los Objetivos de Calidad de la Organización

<Supervisor de ACS> revisa la existencia de objetivos de Calidad de la Organización para el proceso de desarrollo de software.

4.2 Establece los Requerimientos de Calidad del Software

<Supervisor de ACS> define las metas de calidad en términos de costo, esfuerzo, calendario para que puedan ser medidos. Identifica las características de calidad de software necesario para cumplir estos objetivos.

<Supervisor de ACS> determina las características deseadas del software. Para cada característica identifica metas de calidad demostradas por el software o medidas.

4.3 Identifica las métricas de Calidad del Software

<Supervisor de ACS> selecciona los factores de calidad de acuerdo a un criterio de selección. Enfoca una combinación de factores de calidad según las necesidades.

<Supervisor de ACS> determina el criterio de selección de la métrica que satisfaga las necesidades de calidad del software. Nosotros proponemos la detección de Defectos y el uso de Métricas de Calidad según el Anexo B.

<Supervisor de ACS> determina las mediciones que puedan ser usadas para mejorar el proceso de desarrollo y mantenimiento del software.

<Supervisor de ACS> determina las mediciones que puedan ser usadas para mejorar la calidad del software como producto.

<Supervisor de ACS> determina las mediciones que puedan ser usadas para controlar y mejorar el Proyecto.

<Supervisor de ACS> asigna a cada factor de Calidad la métrica que represente mejor al factor de acuerdo al criterio de selección, y análisis de costo y beneficio.

4.4 Prepara la Colección de Datos y ejecuta la Métrica apropiada

<Analista de ACS> revisa los procedimientos técnicos de ejecución de la métrica, define los procedimientos de colección de datos, selecciona el software necesario para realizar la medición. La información sobre el uso de las métricas se encuentra en el Anexo B.

<Supervisor de ACS> selecciona las métricas de acuerdo al tipo de revisión o inspección.

<Analista de ACS> ejecuta el cálculo de la medición de acuerdo a los procedimientos y métricas del producto, según los procedimientos técnicos de ejecución de la métrica, señalados en el Anexo B.

4.5 Evalúa los resultados de la Herramienta Automática en uso

<Analista de ACS> registra los resultados en el *<Reporte de Inspección>* y las conclusiones respecto al uso de la *<herramienta>*. Informa y recomienda al *<Supervisor de ACS>* si fuese necesario la selección de otra herramienta.

4.6 Analiza los resultados de las métricas aplicadas

<Supervisor de ACS> analiza los resultados de las métricas del software interpretando los resultados, identificando la calidad del software, haciendo predicciones de la calidad del software, comparando con valores históricos de referencia para determinar si es necesario realizar nuevas pruebas.

<Supervisor de ACS> analiza las mediciones para asegurar que la calidad del software desarrollado satisface los objetivos de calidad y son aceptables.

<Supervisor de ACS> chequea la consistencia de los criterios de aceptación y la suficiencia de las pruebas para demostrar que los objetivos se han cumplido.

4.7 Monitorea las Mediciones

<Analista o Supervisor de ACS> monitorea el progreso hecho durante el desarrollo.

<Analista o Supervisor de ACS> analiza las medidas para determinar si los resultados son suficientes para satisfacer las metas de calidad.

<Analista o Supervisor de ACS> verifica si una medida ayuda ha determinar la calidad del producto o proceso.

<Analista o Supervisor de ACS> toma acciones correctivas cuando sean necesarias para mejorar las características deseadas del software.

4.8 Registra Información Histórica

<Analista o Supervisor de ACS> registra los resultados de las métricas del software en el *<formato de defectos>* los que serán relacionados al Proyecto de Software para ser usados en futuros proyectos.

4.9 Criterios de Aceptación

<Supervisor de ACS> se reúne con él *<Autor>* para evalúar los resultados de las mediciones y métricas ejecutadas.

<Supervisor de ACS> chequear la consistencia de los criterios de aceptación y la suficiencia de las pruebas para demostrar que los objetivos se han cumplido.

Al finalizar la inspección todos los participantes terminan firmando el documento de *<formato de defectos>*.

5. Documentación relacionada

<Documento de Revisión>

<Formato de Defectos>

6. Formatos

<Formato de Inspección> Este formato consta de 2 páginas, deberá ser utilizado desde la fase de Diseño hasta la fase de Construcción y sirve para registrar datos generales sobre la inspección efectuada; tales como: nombre del proyecto, encargados de la inspección, resultados, observaciones, comentarios, defectos o errores encontrados, herramientas utilizadas. El contenido de este formato servirá de referencia para la elaboración de informes sobre las inspecciones.

Q Z	**Inspección de** _____ Num: ____ Pag: 1/2
Proyecto _____ Responsable Inspección _____ Etapa	Descripción Producto _____ Release Versión _____ _____ Fecha Inicio Fecha Final

		Tiempo Preparac	Tiempo Inspec.	Fecha Inicio	Fecha Fin
Inspector					
Analista					
Analista					
Analista					
Analista					
Analista					
Analista					

Métrica		# Unidades Inspeccionadas	
		# Paginas Inspeccionada	
		# Líneas Inspeccionadas	
Tools		Tiempo Total Preparación	
		Tiempo Total Inspección	
		Tiempo Total Usado	
		Total Inspectores	

Resultado de la Inspección:		Observaciones :
Aceptado	( )	_____
No Aceptado	( )	_____
Por Modificar	( )	_____
Nueva Revisión	( )	_____
		_____

Q Z	**Inspección de _____ Num: ____ Pag: 2/2**

Proyecto	Descripción Producto
Responsable Inspección	Release Versión
Etapa	Fecha Inicio Fecha Final

Producto				**Defecto**					**Revisión**	
Tipo	Pag.	Lin	Unid	Cod.	Severidad	Categoría	Fase Origen	Descripción	Completo	Verificado

Métrica	Herramientas

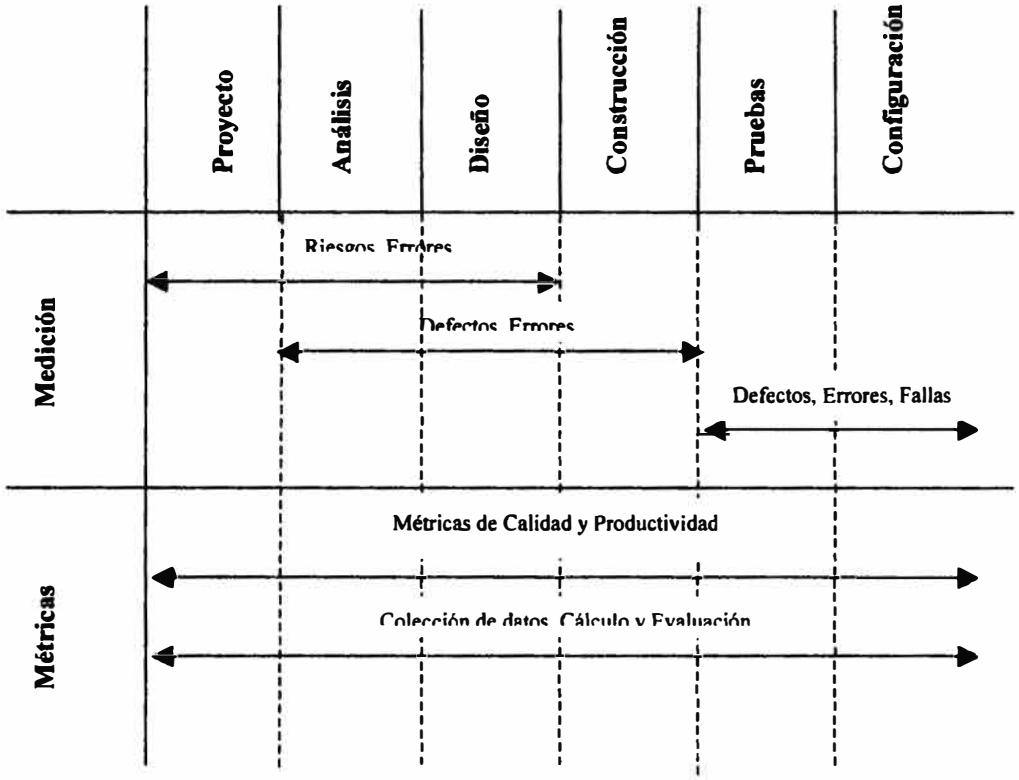


Fig. Proceso de Medición

Fuente: Elaboración Propia

Procedimiento
3.8 Registro de Calidad

Se utiliza para el registro de los niveles de Calidad del Software, esto permitirá tener valores históricos de los proyectos en los cuales se haya efectuado mediciones.
En este formato se deberá registrar información de las personas que aprobaron el procedimiento, cambios que se han efectuado y los departamentos en los cuales debe ser aplicado.

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Cambios

Revisión	Fecha	Persona Responsable	Descripción del Cambio
A		Nombre	Revisión Inicial

Lista de Distribución

(lista de los departamentos que reciben copia del procedimiento)

1. Propósito

- Registrar los resultados de las métricas y mediciones del Proyecto.
- Almacenar los archivos de calidad por un período especificó.
- Identificar que el almacenamiento de los archivos de calidad se encuentre protegido de daños y facilitar su identificación, recuperación y disposición.
- Asegurar que los archivos se encuentren legibles, fechados (incluyendo fecha de la revisión), limpios, identificables y mantenidos de una manera ordenada.

2. Alcance

Este procedimiento es aplicado a todos los archivos que intervenga en las revisiones de calidad.

3. Responsabilidades

<Equipo de ACS> es responsable de mantener los archivos de calidad definidos en este procedimiento.

4. Procedimiento de Registro de la Calidad

4.1 General

Las métricas de calidad y productividad serán resumidas por *<Equipo de ACS>* en un registro de Calidad de acuerdo al *<formato de Calidad>*.

<Equipo de ACS> determina el índice de defectos de todas las etapas del proyecto, las cuales permitirán definir una línea base de estimación en futuros proyectos.

<Equipo de ACS> obtiene la información necesaria de los reportes de las revisiones e inspecciones realizadas a lo largo del proyecto.

<Equipo de ACS> diseña un Registro de Calidad y formatos adecuados que corresponda a las expectativas y objetivos de medición del negocio o tipo de proyecto.

4.2 Registra las mediciones de Calidad

<Equipo de ACS> consolida las mediciones realizadas en las diversas etapas del Desarrollo y los registra en el formato de Calidad adjunto.

4.3 Almacenaje

Los archivos de calidad son almacenados en *<localización>*. El tiempo es determinado por el *<Gerente de ACS>* e indicado en las tablas al final del procedimiento.

Los archivos de calidad son protegidos de los daños siguientes:

- Roturas
- Manchas
- entre otros

Los archivos de calidad son protegidos de pérdidas siguientes:

- Por incendios
- Por extravíos
- entre otros

Los archivos de calidad son legibles y leíbles. Cuando son computarizados, los archivos de calidad siguen un procedimiento de backup establecido.

<Operador> es responsable de obtener un backup de los archivos si estos han sido automatizados.

4.4 Archivos

Los formatos anexos al procedimiento registran donde se almacenan los archivos de calidad y el tiempo de almacenaje.

5. Documentación relacionada

<Documentos de Revisión>
<Formatos de Revisión e Inspección>
<Formatos del Registro de Calidad>

6. Formatos

<Formato de Registro de la Calidad> Este formato consta de 3 páginas, será utilizado durante todo el ciclo de desarrollo del proyecto, servirá de base

histórica de futuras comparaciones y estimaciones; asimismo, servirá como base para la obtención del nivel de calidad del Software que se esta desarrollando.

<*Formato de Archivos de la Calidad*> Este formato consta de 1 página, deberá ser utilizado en todo momento para tener una relación actualizada de toda la documentación relacionada con las actividades del <*Equipo de ACS*> para un proyecto dado.

Q Z	**Registro de Calidad**	Num: ____ Pag: 1/3
Proyecto Software _____ Responsable del Proyecto _____ Tipo Proyecto		Proyecto de ACS _____ Responsable de ACS _____ Fecha Inicio Fecha Final

(antes de su instalación y/o distribución, el mismo formato puede ser utilizado en la etapa de post- evaluación)

PROYECTO	#	**Defectos**	#	%
# personas Equipo de Desarrollo		Fase de Proyecto		
# días Total Trabajo Equipo de Des.		Fase de Análisis		
# personas Equipo de ACS		Fase de Diseño		
# días Total Trabajo Equipo de ACS		Fase de Codificación		
# Total páginas documentación		Pruebas Unitarias		
Indice Calidad Software QZ		Pruebas de Sistema		
Indice de Madurez del Software		Pruebas de Aceptación		
Productividad Desarrollo		# total de defectos		
Tiempo Total Equipo de Desarrollo				
Tiempo Total del Equipo de ACS				
Tiempo Total Supervisor de ACS				
Tiempo Total Analista de ACS				

REQUERIMIENTOS	#	**Mediciones**	#
# Unidades Inspeccionadas		# defectos Serios	
# Paginas Inspeccionada		# defectos Medios	
# Líneas Inspeccionadas		# defectos Triviales	
# Paginas de Documentación		# Total defectos	
# Total de Analista del Proyecto		Indice Defectos de la Fase	
# Total Inspectores		Densidad Defectos	
Tiempo Total Equipo de Desarrollo		H-H por defectos	
Tiempo Total Preparación de ACS		Estadística errores por fase	
Tiempo Total Inspección de ACS		Facilidad de Uso	
Tiempo Total Usado de ACS			

DISEÑO	#	**Mediciones**	#
# Unidades Inspeccionadas		# defectos Serios	
# Paginas Inspeccionada		# defectos Medios	
# Líneas Inspeccionadas		# defectos Triviales	
# páginas documentación		# Total defectos	
# Total de Analista del Proyecto		Indice Defectos de la fase	
# Total Inspectores		Densidad Defectos	
Tiempo Total Equipo de Desarrollo		H-H por defectos	
Tiempo Total Preparación de ACS		Estadística errores por fase	
Tiempo Total Inspección de ACS		Promedio Estructura del Diseño	
Tiempo Total Usado de ACS		Promedio Integridad	

INSPECCION CODIGO	#	**Mediciones**	#
# Programas		# defectos Serios	
# Unidades Inspeccionadas		# defectos Medios	
# Paginas Inspeccionada		# defectos Triviales	
# Líneas Inspeccionadas		# Total defectos	
# páginas documentación		Indice Defectos de la fase	
# Total de Programadores		Densidad Defectos	
# Total Inspectores		H-H por defectos	
Tiempo Total Equipo de Desarrollo		Estadística errores por fase	
Tiempo Total Preparación de ACS		Promedio Indice Ciencia de Healstead	
Tiempo Total Inspección de ACS			
Tiempo Total Usado de ACS			

PRUEBAS UNITARIAS	#	**Mediciones**	#
# Módulos		# defectos Serios	
# Unidades Inspeccionadas		# defectos Medios	
# Paginas Inspeccionada		# defectos Triviales	
# Líneas Inspeccionadas		# Total defectos	
# Casos de Pruebas		Indice Defectos de la fase	
# fallas sembradas		Densidad Defectos	
# fallas sembradas detectadas		H-H por defectos	
# Total de Analistas y/o Programador		Estadística errores por fase	
# Total Inspectores		Indice Cobertura de Pruebas modular	
Tiempo Total Equipo de Desarrollo		Cobertura de Pruebas	
Tiempo Total Preparación de ACS		Exactitud de Pruebas	
Tiempo Total Inspección de ACS			
Tiempo Total Usado de ACS			

PRUEBAS INTEGRALES	#	**Mediciones**	#
# Unidades Inspeccionadas		# Casos de Pruebas	
# Fallas Total sembradas		# defectos Serios	
# Fallas Total detectadas		# defectos Medios	
# Fallas después de pruebas		# defectos Triviales	
# módulos integrados		# Total defectos	
# módulos de configuración		Indice Defectos de la fase	
# Total de Analistas y/o Programador		Densidad Defectos	
# Total Inspectores		H-H por defectos	
Tiempo Total Equipo de Desarrollo		Estadística errores por fase	
Tiempo Total Preparación de ACS		Suficiencia de Pruebas	
Tiempo Total Inspección de ACS		Cobertura de Pruebas	
Tiempo Total Usado de ACS		Exactitud de Pruebas	

PRUEBAS SISTEMAS	#	**Mediciones**	#
# Unidades Inspeccionadas		# defectos Serios	
# Fallas Total sembradas		# defectos Medios	
# Fallas Total detectadas		# defectos Triviales	
# Fallas después de pruebas		# Total defectos	
# Casos de Pruebas		Indice Defectos de la fase	
Capacidad implementadas		Densidad Defectos	
Capacidades requeridas		H-H por defectos	
Entrada de programas		Estadística errores por fase	
Total entradas de Programas			
# Total de Analistas y/o Programador			
# Total Inspectores			
Tiempo Total Equipo de Desarrollo			
Tiempo Total Preparación de ACS		Cobertura de Pruebas	
Tiempo Total Inspección de ACS			
Tiempo Total Usado de ACS			

PRUEBAS ACEPTACION	#	**Mediciones**	#
# Unidades Inspeccionadas		# defectos Serios	
# Fallas Total sembradas		# defectos Medios	
# Fallas Total detectadas		# defectos Triviales	
# Fallas después de pruebas		# Total defectos	
# Casos de Pruebas		Indice Defectos de la fase	
Capacidad implementadas		Densidad Defectos	
Capacidades requeridas		H-H por defectos	
Entradas de Programas (casos de pruebas, capacidades funcionales)		Estadística errores por fase	
Tiempo Total Equipo de Desarrollo			
Tiempo Total Preparación de ACS			
Tiempo Total Inspección de ACS		Cobertura de Pruebas	
Tiempo Total Usado de ACS			
CONTROL DE CAMBIOS	#		
# Pers asignado al Mant. Correctivo		# Orden Cambio de Mant. Correctivo	
# Pers asignado al Mant. Adaptativo		# Orden Cambio de Mant. Adaptativo	
# Pers asignado al Mant. Perfectivo		# Orden Cambio de Mant. Perfectivo	
# horas involuc en Mant. Correctivo		# Prog/Orden Cambio x Mant. Correc	
# horas involuc en Mant. Adaptativo		# Prog/Orden Cambio x Mant. Adapt	
# horas involuc en Mant. Perfectivo		Tiempo Total medio (h) Orden Camb	

Revisado	Observaciones :
Gerente de ACS	
Firma	
Fecha	

QZ	**Archivos de Calidad**	Num: ____ Pag: 1/1

Proyecto _____ Responsable Proyecto de ACS	Recepción _____ Fecha

#	Documento: Código:	Tipo Documento	Código Almacén	Tiempo Almacenaje

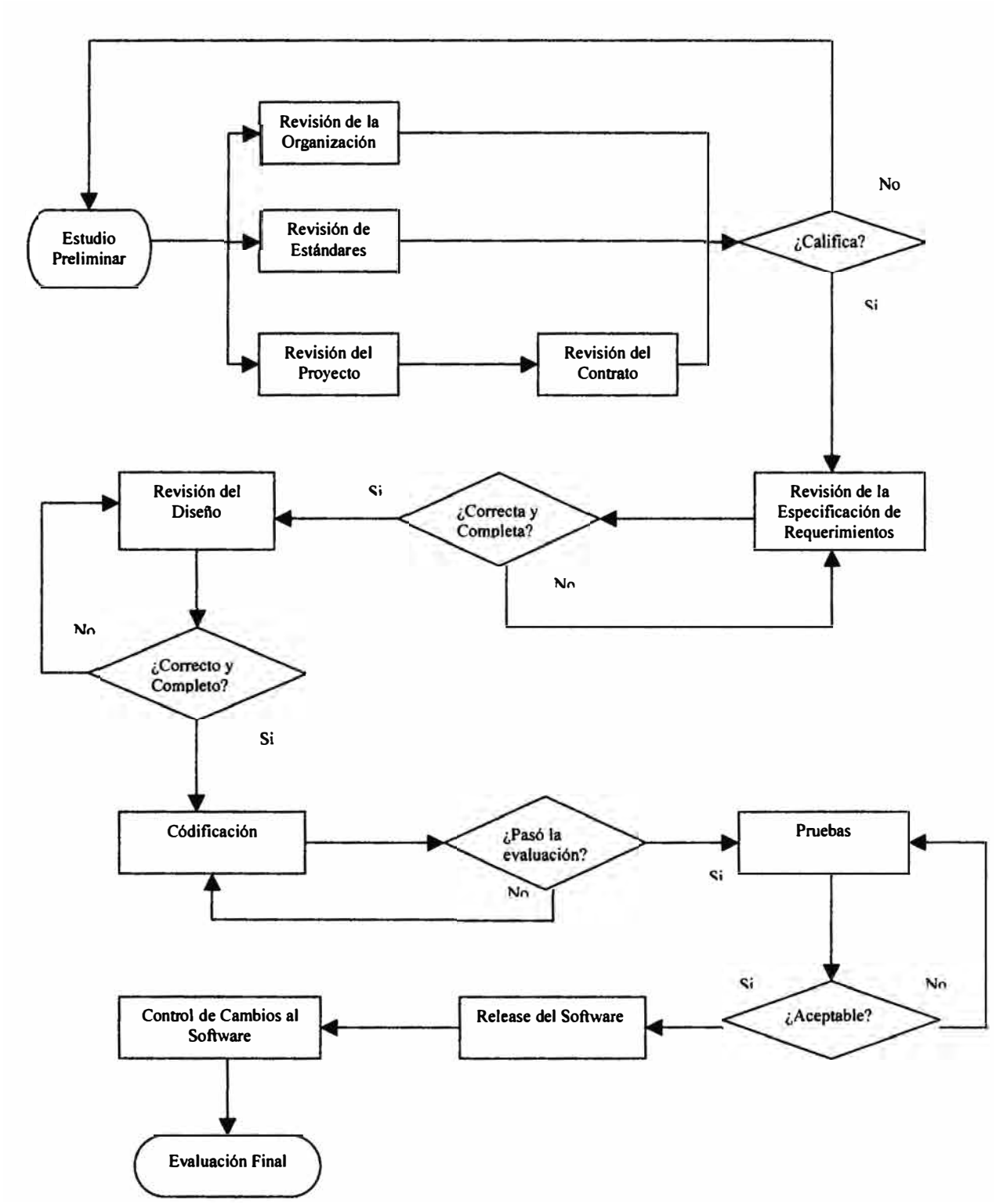
Responsable Almacén _____ _____ _____ Firma	Observaciones _____ _____ _____ _____

3.9 Actividades del Plan de ACS

A continuación indicamos las actividades que contendrá el Plan de Aseguramiento de la Calidad del Software, las que serán definidas en los capítulos correspondientes:

- **Estudio Preliminar**
- **Plan de Calidad del Software**
  - Revisión de la Planificación
    - Revisión de la Organización
    - Revisión de la Planificación del Proyecto
    - Revisión del Contrato
    - Revisión de Estándares
  - Revisión del Análisis de Requerimientos
  - Revisión del Diseño
    - Revisión del Diseño Lógico
    - Revisión del Diseño Físico
  - Revisión de la Construcción
- **Plan de Pruebas del Software**
  - Elaboración del Plan de Pruebas
  - Casos de Prueba
  - Pruebas Unitarias
  - Pruebas de Integración
  - Pruebas de Sistema
  - Pruebas de Aceptación
- **Plan de Configuración del Software**
  - Revisión del Ambiente de la Configuración
  - Revisión del Control de Cambios
  - Revisión del Control de Versiones
  - Revisión del Control de Releases
- **Evaluación Final**

DIAGRAMA DE FLUJO DE ACTIVIDADES

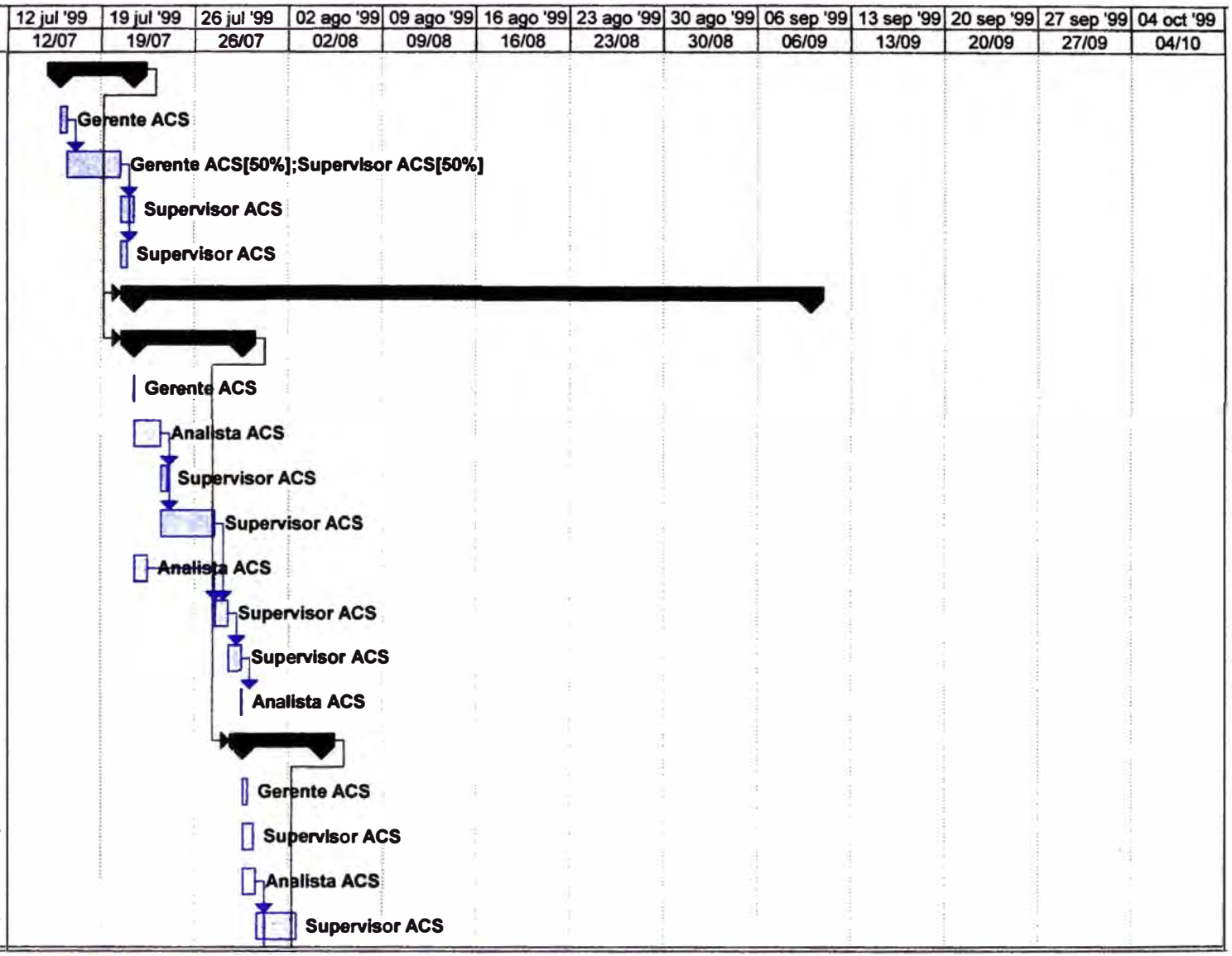


Actividades del Plan de ACS

ENTRADAS	▪ Plan Estratégico ▪ Plan de Sistemas ▪ Plan de Contingencias ▪ Manual de Organización y Métodos ▪ Manual de Procedimientos ▪ Plan del Proyecto ▪ Cronograma de Actividades del Equipo del Proyecto ▪ Análisis de Riesgos ▪ Análisis de Estimación de Esfuerzos, Recursos. ▪ Criterios de Aceptación ▪ Criterios de Validación	▪ Requerimientos del Negocio ▪ Plan de Sistemas ▪ Documentación del Proyecto ▪ Cronograma de Actividades ▪ Especificación de requerimientos ▪ Documentación de Requerimiento de Interfaces ▪ Documentación de Usuario ▪ Análisis de estimación de esfuerzos, riesgos, etc. ▪ Plan de Pruebas ▪ Lista de Comprobación	▪ Estándares ▪ Especificación de Requerimientos. ▪ Requerimientos Funcionales y no Funcionales. ▪ Especificación de requerimientos de Diseño ▪ Documentación de Requerimiento de Interface ▪ Documentación de Diseño de Interface ▪ Documentación de usuario ▪ Plan de Pruebas ▪ Lista de Comprobación	▪ Estándares ▪ Especificación de requerimientos de Diseño Funcional o Físico ▪ Cronograma de Actividades ▪ Estimación del esfuerzo ▪ Listado de Código fuente ▪ Código ejecutable ▪ Documentación de diseño de interface ▪ Documentación de usuario ▪ Plan de Pruebas ▪ Lista de Comprobación	▪ Listado de Código Fuente ▪ Código ejecutable ▪ Documentación de usuario ▪ Lista de Comprobación	▪ Cronograma de Actividades ▪ Documentación Planificación ▪ Especificación de requerimientos de análisis ▪ Documentación requerimientos interfaces ▪ Especificación de requerimientos de diseño ▪ Documentación Diseño Interfaces ▪ Listado Código Fuente ▪ Código ejecutable ▪ Documentación de usuario ▪ Cambios aprobados/propuestos ▪ Reporte de anomalías
	PROYECTO	**REQUERIMIENTO**	**DISEÑO**	**CONSTRUCCION**	**PRUEBAS**	**CONFIGURACION**
TAREAS	▪ Identificación de los objetivos del Proyecto ▪ Revisión de los Planes Estratégicos, de Sistemas, de Contingencias, Proyecto ▪ Evaluación de la Configuración ▪ Verificar los Criterios de Aceptación y Verificación ▪ Evaluar y medir riesgos del Proyecto	▪ Revisión de la Especificación de requerimientos. ▪ Revisión del Análisis de requerimientos. ▪ Verificar los Criterios de Validación. ▪ Generación de Plan de Pruebas: 1. De Sistema 2. De Aceptación	▪ Revisión del Diseño. ▪ Seguimiento de la especificación de Requerimientos. ▪ Generación de Plan de Pruebas: 1. Unitarias 2. De Integración ▪ Generación de diseño de Pruebas: 1. Unitarias 2. De Integración 3. De Sistema 4. De Aceptación	▪ Seguimiento del Diseño ▪ Evaluación del Código ▪ Análisis de Interfaces ▪ Generación de casos de pruebas: 1. Unitarias 2. De Integración 3. De Sistema 4. De Aceptación ▪ Generación de procedimientos de pruebas: 1. Unitarias 2. De Integración 3. De Sistema ▪ Ejecución de prueba Unitarias	▪ Generación de Procedimiento de Pruebas de aceptación ▪ Ejecución de Pruebas: 1. De Integración 2. De Sistema 3. De Aceptación	▪ Revisión del Plan de Pruebas. ▪ Evaluación de anomalías ▪ Evaluación de Cambios propuestos
SALIDAS	▪ Informe de revisión ▪ Análisis y evaluación de riesgos	▪ Plan de Pruebas (actualizado) 1. De Sistema 2. De Aceptación ▪ Reporte de anomalías ▪ Análisis y evaluación de riesgos y defectos.	▪ Plan de Pruebas (actualizado) 1. Unitarias 2. De Integración ▪ Diseño de Pruebas 1. Unitarias 2. De Integración 3. De Sistema 4. De Aceptación ▪ Reporte de anomalías ▪ Análisis y evaluación de riesgos.	▪ Casos de Pruebas 1. Componente 2. Integración 3. Sistema 4. Aceptación ▪ Procedimientos de Prueba: 1. Componentes 2. Integración 3. Sistema ▪ Reporte de anomalías ▪ Informe de Inspección	▪ Reporte de tareas de fase de pruebas. ▪ Procedimientos de pruebas Aceptación. ▪ Reporte de anomalías. ▪ Informe de Inspección	▪ Plan de Pruebas actualizado ▪ Reporte tareas de O&M ▪ Reportes de anomalías

Plan de Actividades

Id	Nombre de tarea	Duración
1	**Estudio Preliminar**	**3.5 días**
2	Identifica los Objetivos	4 horas
3	Diseño del Plan de ACS	16 horas
4	Cronograma de actividade	8 horas
5	Organización del Equipo d	4 horas
6	**PLAN DE CALIDAD**	**36.5 días**
7	**Revisión de la Organizac**	**6.25 días**
8	Supervisión y Control	2 horas
9	Recopila información	16 horas
10	Identificación de los o	4 horas
11	Revisa la Información	16 horas
12	Aplica lista de compro	8 horas
13	Evalúa y mide los ries	8 horas
14	Informa sobre la revis	8 horas
15	Registra los resultado	2 horas
16	**Revisión de la Planificac**	**3.75 días**
17	Supervisión y Control	2 horas
18	Identifica los Objetivo	4 horas
19	Recopila información	8 horas
20	Revisa la Información	8 horas

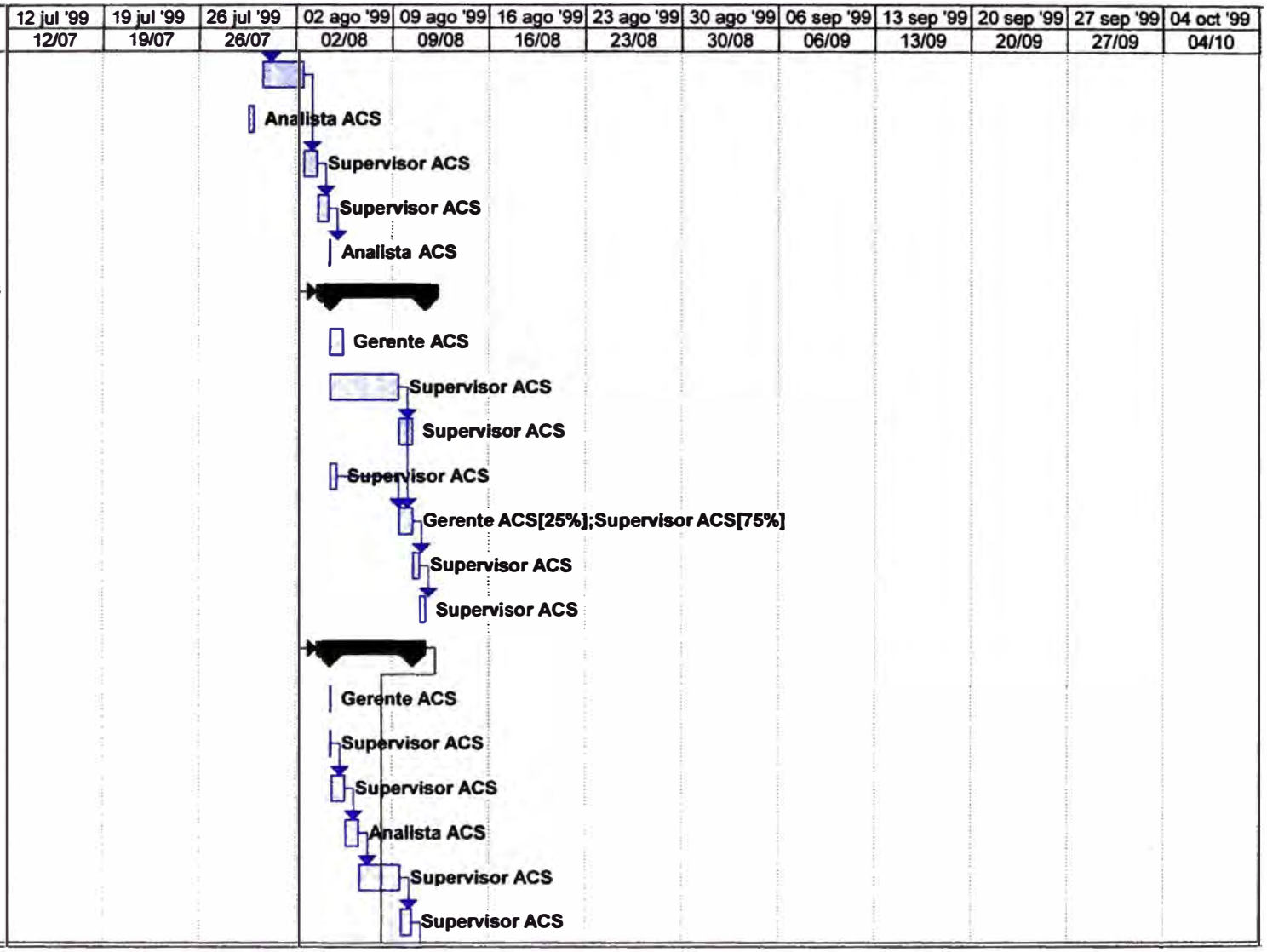


Proyecto: ProySQA
Fecha: lu 05/07/99

Tarea
División
Progreso
Hito
Resumen
Tarea resumida
División resumida
Hito resumido
Progreso resumido
Tareas externas
Resumen del proyecto

Plan de Actividades

Id	Nombre de tarea	Duración
21	Verificar los Criterios	8 horas
22	Aplica lista de compro	2 horas
23	Evalúa y mide los rie	8 horas
24	Informa sobre la revis	4 horas
25	Registra los resultado	2 horas
26	**Revisión del Contrato**	**4.75 dias**
27	Supervisión y Control	8 horas
28	Recopila información	24 horas
29	Revisa y verifica el cc	8 horas
30	Aplica lista de compro	4 horas
31	Evalúa y mide los rie	8 horas
32	Informa sobre la revis	4 horas
33	Registra los resultado	2 horas
34	**Revisión de Estándares**	**4 dias**
35	Supervisión y Control	2 horas
36	Verifica la existencia	2 horas
37	Revisa y evalúa los E	8 horas
38	Aplica lista de compro	8 horas
39	Evalúa y mide los rie	8 horas
40	Informa sobre la revis	4 horas

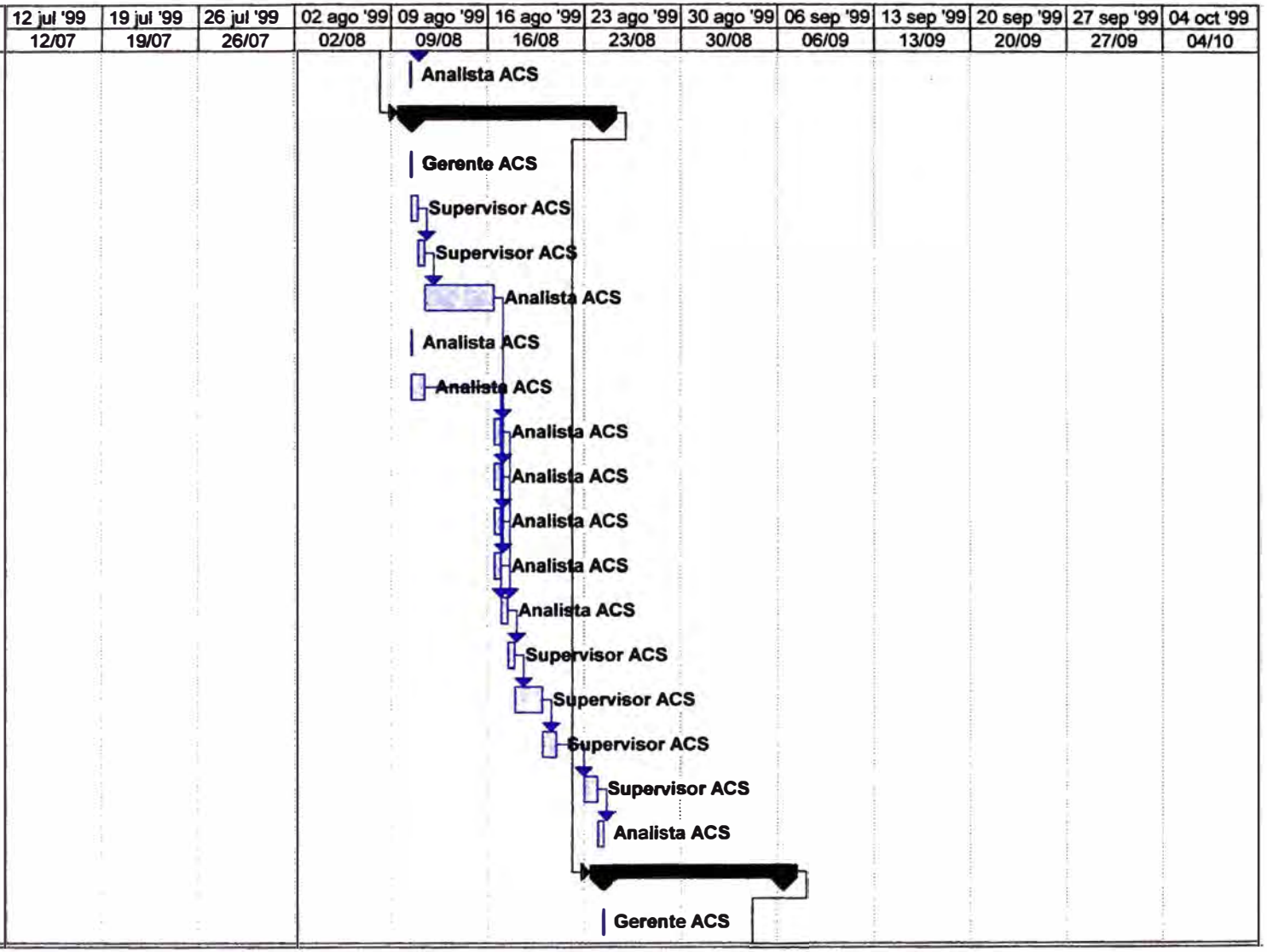


Proyecto: ProySQA
Fecha: lu 05/07/99

Tarea
División
Progreso
Hito
Resumen
Tarea resumida
División resumida
Hito resumido
Progreso resumido
Tareas externas
Resumen del proyecto

Plan de Actividades

Id	Nombre de tarea	Duración
41	Registra los resultado	2 horas
42	**Análisis de Requerimient**	**9.75 días**
43	Supervisión y Control	2 horas
44	Verifica las fuentes de	4 horas
45	Verifica la naturaleza	4 horas
46	Verifica las caracterís	24 horas
47	Verifica como fue prej	2 horas
48	Verifica el uso de Pro	8 horas
49	Verifica los Diagrama	4 horas
50	Verifica Diccionario d	4 horas
51	Verifica el Diagrama c	4 horas
52	Verifica el análisis de	4 horas
53	Verifica si el Diseño e	4 horas
54	Verifica si los requerir	4 horas
55	Determina los Criterio	16 horas
56	Evalúa y mide los rie	8 horas
57	Informa sobre la revis	8 horas
58	Registra los resultado	2 horas
59	**Revisión del Diseño**	**9.25 días**
60	Supervisión y Control	2 horas

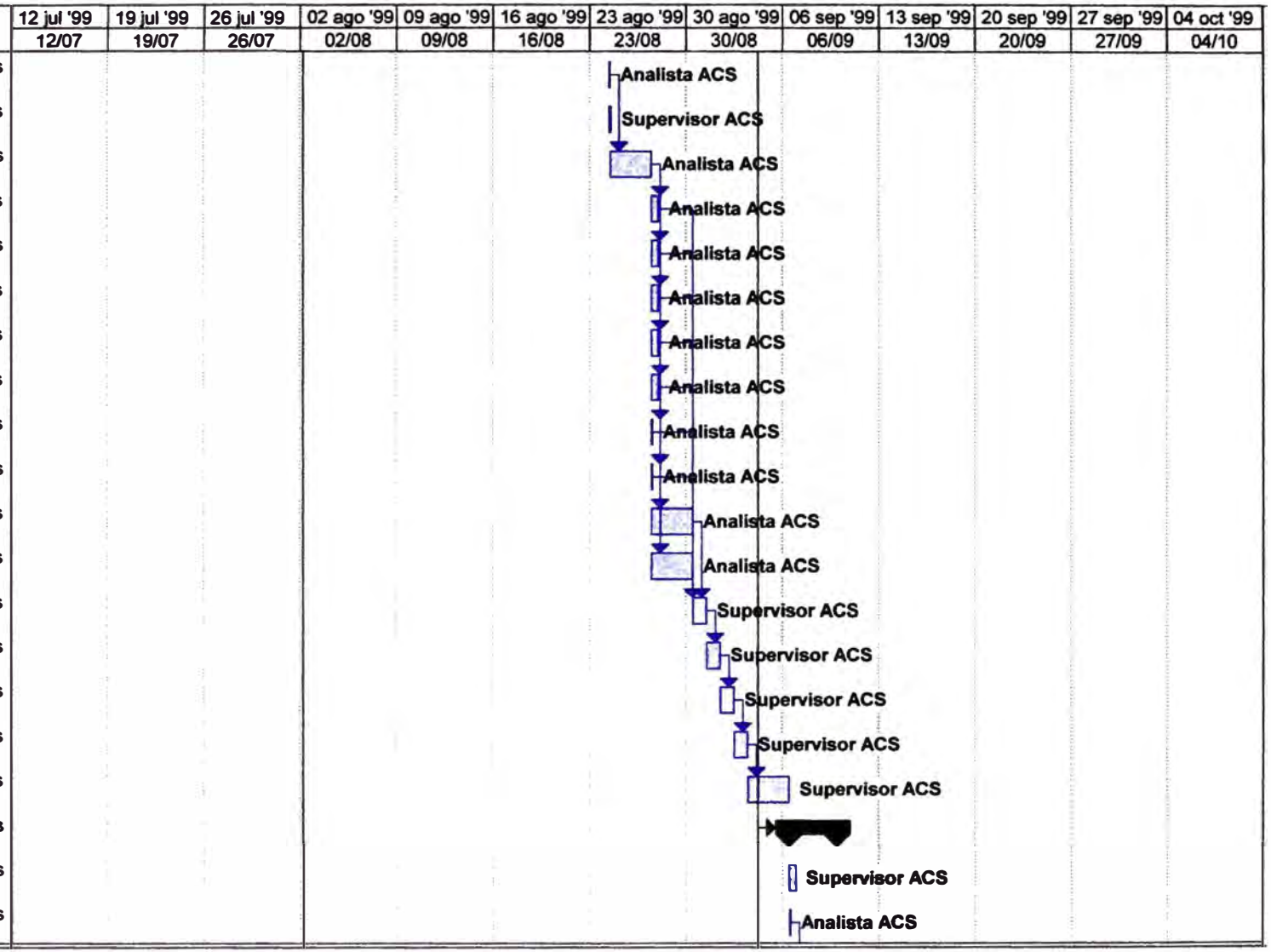


Proyecto: ProySQA
Fecha: lu 05/07/99

Tarea
División
Progreso
Hito
Resumen
Tarea resumida
División resumida
Hito resumido
Progreso resumido
Tareas externas
Resumen del proyecto

Plan de Actividades

Id	Nombre de tarea	Duración
61	Verifica el Método de	2 horas
62	Los Requerimientos ir	4 horas
63	Evalúa las caracterís	24 horas
64	Verifica el Diagrama c	4 horas
65	Verifica el Diagrama c	4 horas
66	Verifica el Diagrama c	4 horas
67	Verifica el Diagrama c	4 horas
68	Verifica el Diseño de l	4 horas
69	Verifica el Diseño de '	2 horas
70	Verifica el Diseño de l	2 horas
71	Evalúa la Configuraci	8 horas
72	Verifica la Documenta	8 horas
73	Verifica los resultados	8 horas
74	Revisión Final	8 horas
75	Reevalua el Diseño	8 horas
76	Revisión de Control	8 horas
77	Determina los Criterio	8 horas
78	**Inspección del Código**	**3.5 días**
79	Identifica los element	4 horas
80	Evalúa el uso de Está	2 horas

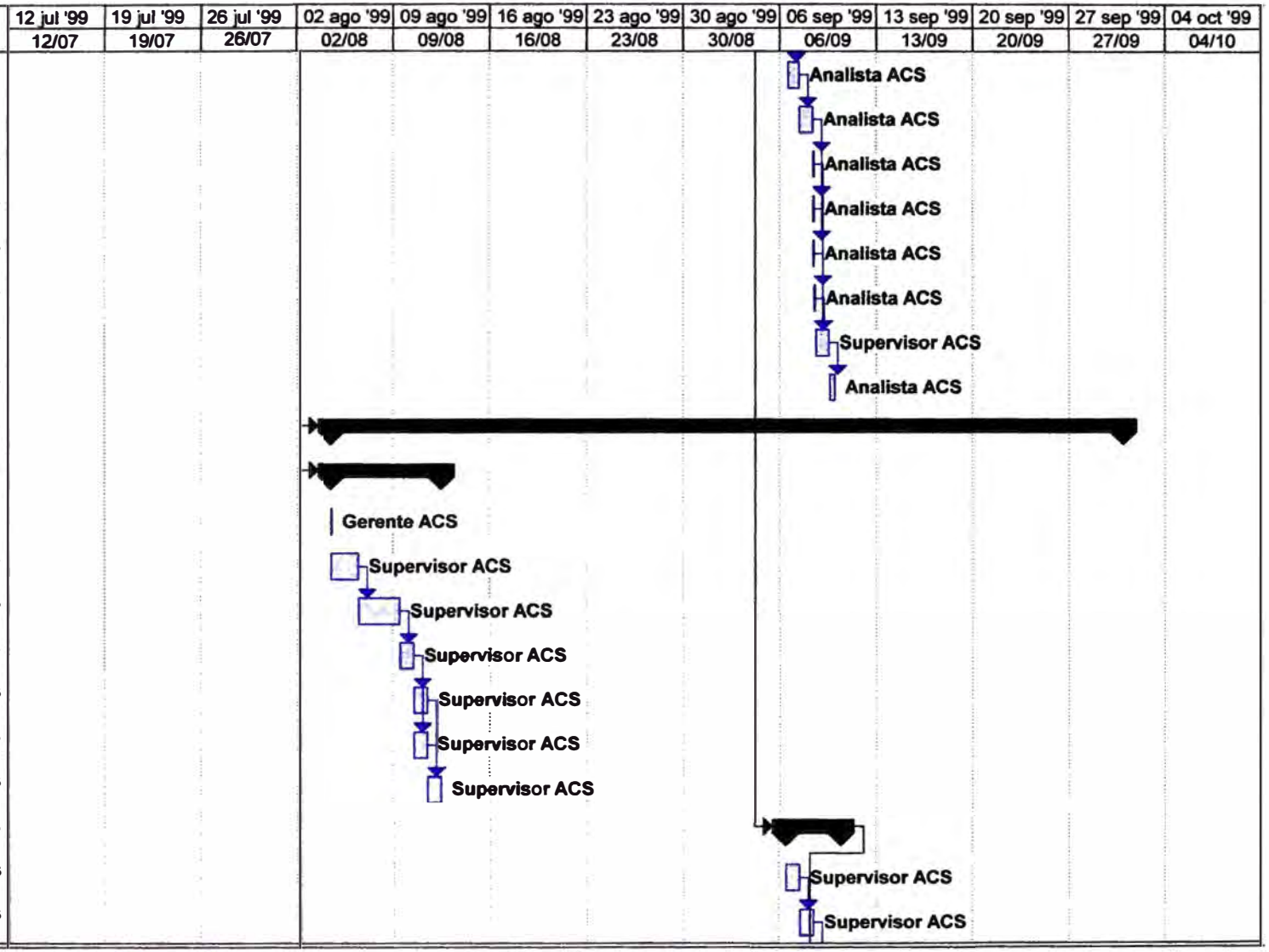


Proyecto: ProySQA
Fecha: lu 05/07/99

Tarea
División
Progreso
Hito
Resumen
Tarea resumida
División resumida
Hito resumido
Progreso resumido
Tareas externas
Resumen del proyecto

Plan de Actividades

Id	Nombre de tarea	Duración
81	Verifica que el Códigc	4 horas
82	Requerimientos del D	8 horas
83	Análisis Estático	2 horas
84	Análisis la Estructura	2 horas
85	Análisis de Interfaces	2 horas
86	Documentación del P	2 horas
87	Informa sobre la Insp	8 horas
88	Registra los resultado	2 horas
89	**PLAN DE PRUEBAS**	**41.5 días**
90	**Administración de Prueb**	**6 días**
91	Supervisión y Control	2 horas
92	Diseña las Pruebas d	16 horas
93	Prepara el Ambiente	8 horas
94	Reporte y Resolucior	8 horas
95	Política de Cambio de	8 horas
96	Política de Desviacior	8 horas
97	Registro y Seguridad	8 horas
98	**Casos de Prueba**	**4 días**
99	Diseña casos de Prue	8 horas
100	Diseña casos de Prue	8 horas

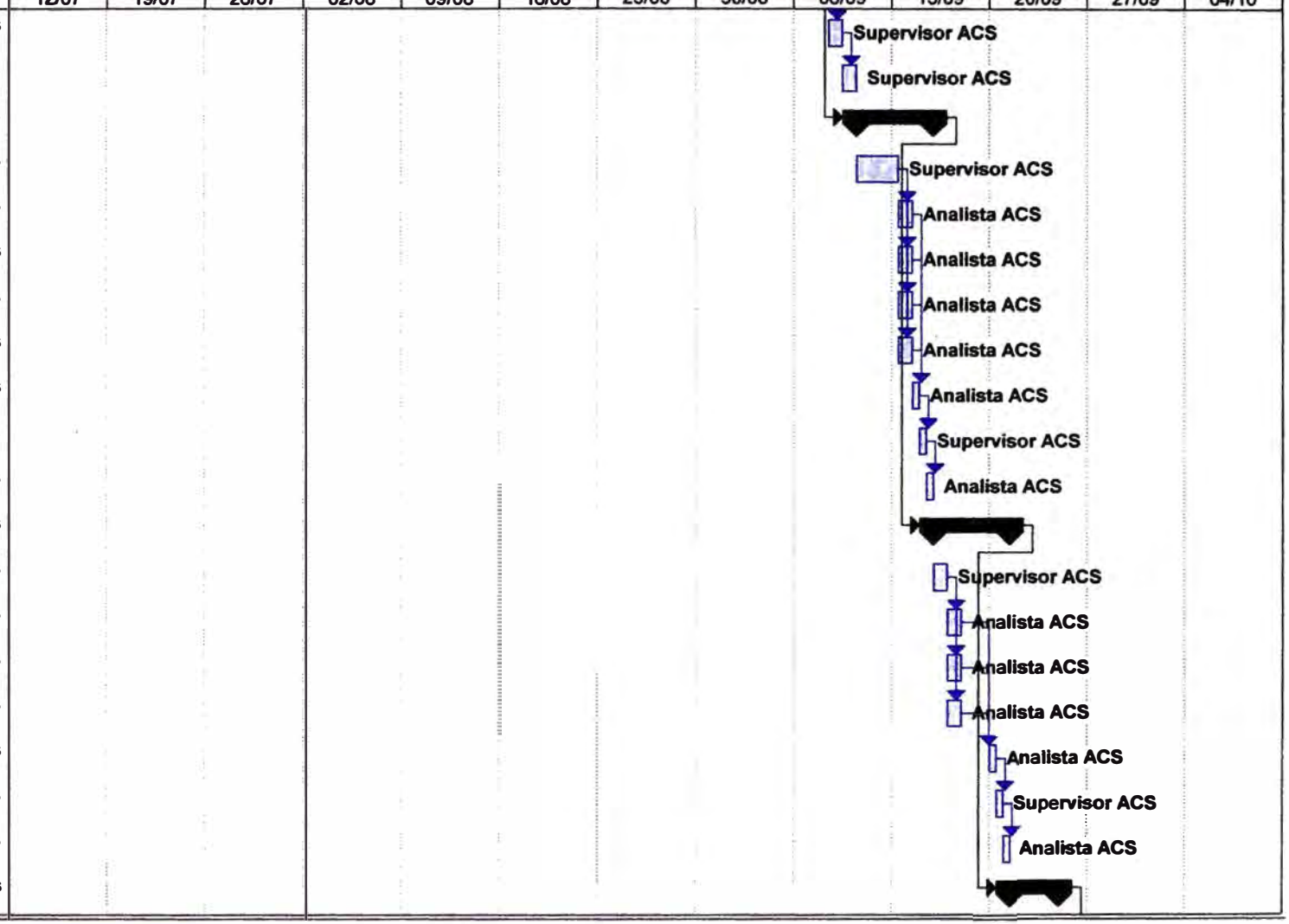


Proyecto: ProySQA
Fecha: lu 05/07/99

Tarea	Resumen	Progreso resumido	
División	Tarea resumida	Tareas externas	
Progreso	División resumida	Resumen del proyecto	
Hito	Hito resumido		

Plan de Actividades

Id	Nombre de tarea	Duración
101	Diseña casos de Prue	8 horas
102	Diseña casos de Prue	8 horas
103	**Pruebas Unitarias**	**3.5 días**
104	Planifica las Pruebas	8 horas
105	Prueba de Interfaz	8 horas
106	Prueba Estructura de	8 horas
107	Prueba de Condicione	8 horas
108	Prueba de Caminos ir	8 horas
109	Reporte de Pruebas	4 horas
110	Informa sobre las pru	4 horas
111	Registra los resultado	4 horas
112	**Pruebas de Integración**	**3.5 días**
113	Planifica las Pruebas	8 horas
114	Prueba de Integración	8 horas
115	Prueba de Integración	8 horas
116	Prueba de Regresión.	8 horas
117	Reporte de Pruebas	4 horas
118	Informa sobre las pru	4 horas
119	Registra los resultado	4 horas
120	**Pruebas de Sistemas**	**3.5 días**

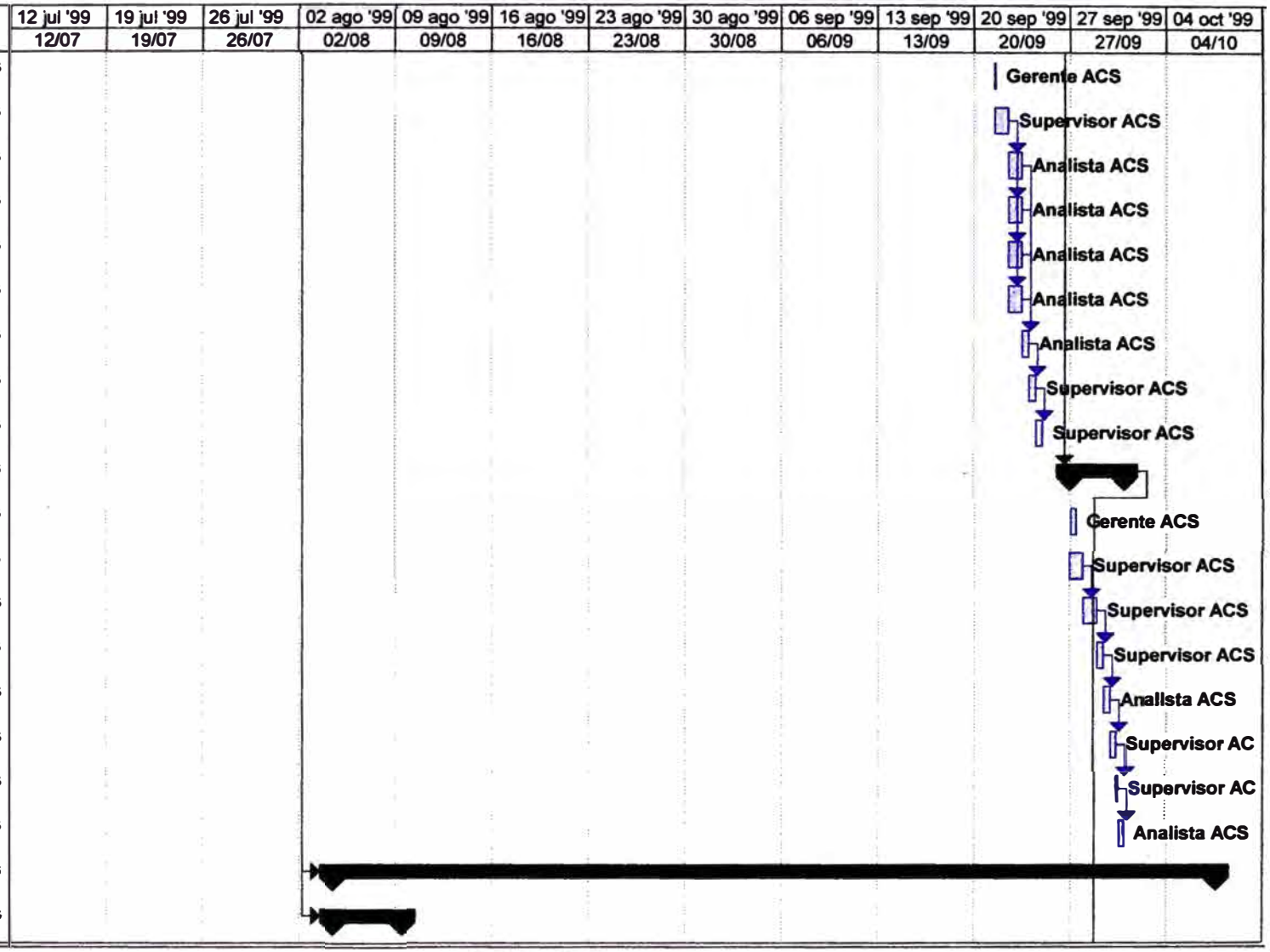


Proyecto: ProySQA
Fecha: lu 05/07/99

Tarea
División
Progreso
Hito
Resumen
Tarea resumida
División resumida
Hito resumido
Progreso resumido
Tareas externas
Resumen del proyecto

Plan de Actividades

Id	Nombre de tarea	Duración
121	Supervisión y Control	2 horas
122	Planifica las Pruebas	8 horas
123	Prueba de Recuperac	8 horas
124	Prueba de Seguridad	8 horas
125	Prueba de Resistenci	8 horas
126	Prueba de Rendimien	8 horas
127	Reporte de Pruebas	4 horas
128	Analiza los resultados	4 horas
129	Informa sobre las pru	4 horas
130	**Pruebas de Aceptación**	**4 días**
131	Supervisión y Control	2 horas
132	Criterios de Aceptació	8 horas
133	Planificación de las P	8 horas
134	Analiza y Documenta	4 horas
135	Reporte de Pruebas	4 horas
136	Logra la Aceptación d	2 horas
137	Informa sobre las pru	4 horas
138	Registra los resultado	2 horas
139	**PLAN DE CONFIGURACION**	**46.38 dias**
140	**Revisión de la Configura**	**3 días**

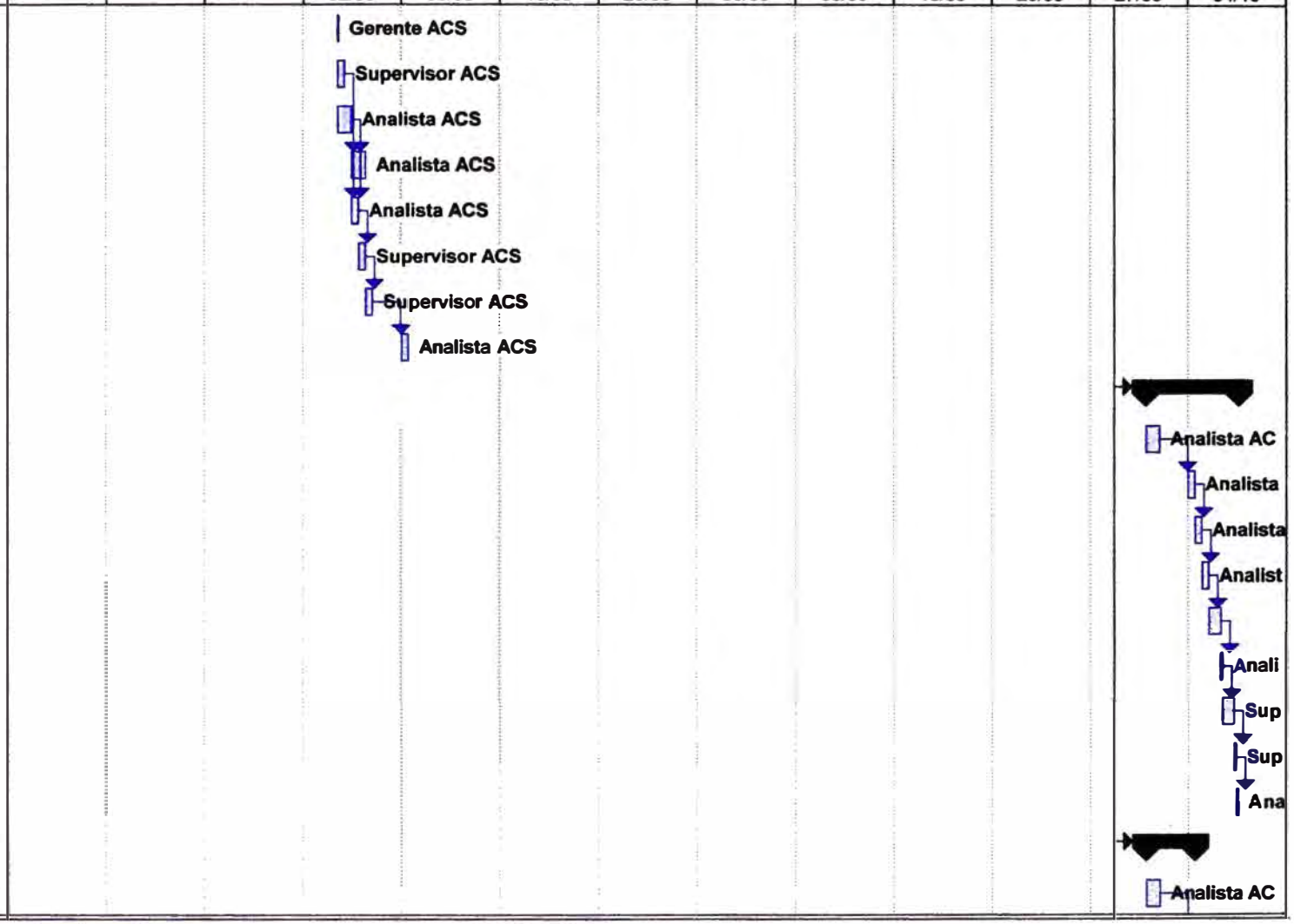


Proyecto: ProySQA
Fecha: lu 05/07/99

Tarea	Resumen	Progreso resumido	
División	Tarea resumida	Tareas externas	
Progreso	División resumida	Resumen del proyecto	
Hito	Hito resumido		

Plan de Actividades

Id	Nombre de tarea	Duración
141	Supervisión y Control	2 horas
142	Identificación de los e	4 horas
143	Instalación del Ambie	8 horas
144	Verifica la Herramient	8 horas
145	Aplica lista de compro	4 horas
146	Evalúa y mide los rie	4 horas
147	Informa sobre la revis	4 horas
148	Registra los resultado	4 horas
149	**Revisión del Control de (**	**4.88 días**
150	Revisa y analiza la so	8 horas
151	Identifica y verifica los	4 horas
152	Garantiza los Cambio	4 horas
153	Audita los Cambios	4 horas
154	Controla el Pase a Pr	5 horas
155	Aplica lista de compro	4 horas
156	Evalúa y mide los rie	4 horas
157	Informa sobre la revis	4 horas
158	Registra los resultado	2 horas
159	**Control de Versiones**	**1.5 días**
160	Verifica las facilidades	8 horas

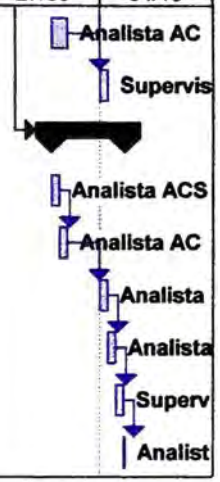


Proyecto: ProySQA
Fecha: lu 05/07/99

Tarea
División
Progreso
Hito
Resumen
Tarea resumida
División resumida
Hito resumido
Progreso resumido
Tareas externas
Resumen del proyecto

Plan de Actividades

Id	Nombre de tarea	Duración
161	Verifica el uso de la H	8 horas
162	Informa sobre la revis	4 horas
163	**Control de Release**	**2.75 días**
164	Verifica la Naturaleza	4 horas
165	Analiza el Cambio	4 horas
166	Controla el cumplimie	4 horas
167	Verifica los elementos	4 horas
168	Informa sobre la revis	4 horas
169	Registra los resultado	2 horas



Proyecto: ProySQA
Fecha: lu 05/07/99

Tarea
División
Progreso
Hito
Resumen
Tarea resumida
División resumida
Hito resumido
Progreso resumido
Tareas externas
Resumen del proyecto

Capitulo 4

PLAN DE CALIDAD DEL SOFTWARE

4.1 Propósito

Hacer cumplir los requerimientos del usuario o cliente y asegurar la calidad del producto mediante revisiones formales desde la fase de Concepción hasta la fase de Codificación, permitiendo que el producto obtenido este preparado para el Plan de Pruebas; además, registrar los resultados de las evaluaciones efectuadas para futuras comparaciones.

4.2 Alcance

Este plan se aplica a todo Software desarrollado internamente o por terceros.

4.3 Actividades

Las actividades propuesta en nuestro plan tiene como meta:

Mejorar la planificación del proyecto.

Disminuir el número de defectos en el Software.

Aumentar la fiabilidad del software.

Disminuir la densidad de defectos en el software.

Reducir los costos de las no conformidades.

Aumentar la productividad en el Desarrollo de Software.

Cada una de nuestras actividades que mencionaremos debe ser concluida por el Equipo de ACS para verificar si el Equipo de Desarrollo puede continuar con su labor, esto permitirá que no se acarreen errores que después no puedan remediarse; de esta manera, llegar a disminuir la cantidad de defectos. Para esto el Equipo de ACS deberá participar activamente en las actividades de Desarrollo del Software para no prolongar el cronograma de actividades del Equipo de Desarrollo; además, deberá anticipar su labor diseñando el Plan de Pruebas y el Diseño de sus casos de pruebas.

A continuación detallamos las actividades a efectuarse en nuestro plan:

- **Revisión de la Planificación**

  La Revisión de la Planificación es una actividad de suma importancia, debido a que en este punto se logrará identificar claramente el ambiente de calidad y los requerimientos existentes en el negocio. Cabe mencionar que una falta de calidad en el ámbito del negocio, puede causar serias dificultades para desarrollar software con calidad.

- **Revisión del Análisis de Requerimientos**

  La Revisión del Análisis de Requerimientos juega un papel importante en todo proceso de desarrollo, ya que una mala conceptualización del sistema genera retrasos, sobrecostos y alto nivel en el mantenimiento del software. Como indicaremos, los requerimientos del negocio deberán ser especificados en forma correcta y completa en la especificación de requerimientos.

- **Revisión del Diseño**

  En la Revisión del Diseño se debe tener muy en cuenta las interfaces internas y externas; pues, esto generaría serios problemas al momento de poner en marcha un sistema generando retrasos, sobrecostos. Como indicaremos, la especificación de requerimientos deberá ser contrastada con la especificación del diseño.

- **Revisión de la Construcción**

  La Revisión de la Construcción, es importante ya que en esta etapa del desarrollo, se le proporcionará al software las características de calidad necesarias para el negocio.

4.4 Revisión de la Planificación

La revisión de la planificación consiste en cuatro actividades, siendo estas:

- **Revisión de la Organización**

  Es necesario efectuar una revisión de la organización; pues, nos permite conocer el ambiente de calidad y control que existe en los procesos de la empresa.

- **Revisión de la Planificación del Proyecto**

  Necesitamos conocer como han sido conceptualizados y planificados los requerimientos del negocio.

- **Revisión del Contrato**

  Es una revisión importante debido a los problemas que se pueden generará por una mala conceptualización de los requerimientos por parte de nuestro proveedor. Cabe mencionar que esto generará no solo perdidas económicas, sino tiempo y recursos invertidos y sobre todo el no haber generado un buen negocio el cual no permita tener mayor utilidad.

- **Revisión de Estándares**

  El uso de estándares es necesario para tener una forma ordenada y controlada de efectuar procesos, documentos, mediciones, entre otros. Los estándares nos permiten tener una forma de no incurrir en costos innecesarios, retrasos en los procesos, toma de decisiones de manera empírica y otros.

Procedimiento

4.4.1 Revisión de la Organización

Esta es una revisión importante, la cual permitirá verificar el ambiente de calidad y control existente en el negocio y encontrar posibles riesgos que lleve a mal fin la ejecución de cualquier proyecto. En cuanto al ambiente de calidad, conocer la existencia de una filosofía de calidad cuyos miembros aseguren un mejoramiento continuo y satisfacción al cliente interno y externo. En cuanto al ambiente de control, la efectividad de los controles, procedimientos, técnicas y segregación de funciones que aseguren el buen control y cumplimiento de los planes y proyectos. En este formato se deberá registrar información de las personas que aprobaron el procedimiento, cambios que se han efectuado y los departamentos en los cuales debe ser aplicado.

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Cambios

Revisión	Fecha	Persona Responsable	Descripción del Cambio
A		Nombre	Revisión Inicial

Lista de Distribución

(lista de los departamentos que reciben copia del procedimiento)

1. Propósito

- Aplicar los conocimientos más recientes de la organización para analizar y realizar evaluaciones de riesgos sobre las diferentes partes del negocio.
- Conocer si se aplica el concepto de calidad en la organización.
- Establecer los procedimientos para examinar y revisar los componentes del negocio.
- Informar los riesgos potenciales encontrados durante la revisión y recomendar las acciones correctivas necesarias.
- Mantener un registro de la revisión de la organización y documentar las actividades de la revisión.

2. Alcance

Este procedimiento se aplica a cualquier organización que se encuentre o vaya a desarrollar software.

3. Responsabilidades

<Gerente General o Director Principal> Proporciona los objetivos y misión del negocio.

<Areas del Negocio> Proporcionaran información relevante sobre las actividades que desarrollan, planes, proyectos, procedimientos, entre otros.

<Gerente de ACS> Solicita información del proyecto, evalúa los resultados obtenidos en la revisión, informa recomendaciones y controla su implantación.

<Supervisor de ACS> Revisa información del negocio para encontrar el nivel de madurez en la calidad que tiene la organización. Elabora informe de la revisión de la organización indicando sus recomendaciones.

4. Procedimiento

4.1 General

<Gerente o Supervisor de ACS> Revisa nuevas formas de gestión.

4.2 Recopila información y documentos relacionados al Negocio

<Gerente o Supervisor de ACS> solicita información y/o documentación relacionada al negocio mediante un *<documento>*.

<Gerente de Area o Departamento> recepciona el *<Documento>* por el cual el *<Gerente o Supervisor de ACS>* solicita la información y/o documentación relacionados al negocio. Responde la solicitud haciendo entrega de la información, documentación y planes solicitados.

<Gerente o Supervisor de ACS> revisa que la información y/o documentación solicitada este completa. En caso contrario indagar el(los) motivo(s) por el que no se ha proporcionado la información o documentación requerida. Elabora una lista de la información y/o documentos faltantes e informa al *<Gerente General o Director Principal>*.

4.3 Identificación de los objetivos y misión del negocio

<Gerente o Supervisor de ACS> revisa el Plan Estratégico del Negocio para identificar los objetivos y misión del negocio y conocer si están basados en la Calidad. En caso de no obtener dicho documento, entablar una reunión con la alta gerencia para establecer e identificar los objetivos y misión del negocio.

4.4 Revisa la Información y documentos obtenidos

<Gerente o Supervisor de ACS> revisa la información y/o documentación obtenidos acerca del negocio para tener un conocimiento cabal, sobre la organización.

4.5 Aplica lista de comprobación

<Analista o Supervisor de ACS> solicita una reunión con el *<Gerente de Area, Dpto>* con el fin de indagar o tener mayor conocimiento de los componentes del negocio.

<Analista o Supervisor de ACS> efectúa una serie de preguntas al *<Gerente o Jefe del Proyecto>* anotando cada una de sus respuestas.

4.6 Evalúa y mide los Riesgos

<Gerente o Supervisor de ACS> Verifica y evalúa que el Plan Estratégico de Sistemas este de acuerdo a los objetivos formulados en el Plan Estratégico del Negocio.

<Gerente o Supervisor de ACS> Verifica y evalúa las políticas de calidad, su aplicación y cumplimiento por parte de la organización. Verifica si existe como procedimiento un control estadístico de los procesos del negocio. Verifica si existen círculos calidad para la solución de problemas, y evalúa su comportamiento y resultados.

<Gerente o Supervisor de ACS> verifica que existan los recursos (humanos, equipos, tecnología, etc.) adecuados e idóneos en la organización para el logro de los objetivos.

<Gerente o Supervisor de ACS> efectúa una evaluación de los resultados obtenidos; luego, desarrolla un análisis de los riesgos en que se pueda estar incurriendo.

<Gerente o Supervisor de ACS> define la prioridad de los riesgos del negocio. Indica las recomendaciones necesarias para atenuar o eliminar los riesgos.

4.7 Informa sobre la revisión

<Gerente o Supervisor de ACS> informa al *<Gerente General o Director Principal>* los riesgos, su prioridad, su impacto, técnicas de control y recomendaciones a implantar en los procesos del negocio.

<Gerente General o Director Principal> lee las observaciones efectuadas por el *<Gerente o Supervisor de ACS>* y apoya la implantación de las recomendaciones.

<Gerente o Supervisor de ACS> controla que las recomendaciones hayan sido implantadas.

4.8 Registra los resultados obtenidos

<Gerente o Supervisor de ACS> registra los resultados obtenidos en *<formatos>* o *<archivos>* los cuales almacena en *<almacén>*. En caso de tener un mecanismo automatizado registrar los datos e información pertinente. Actualiza el Plan de Calidad del Negocio.

5. Documentación relacionada

<Entradas>

1. Plan Estratégico del Negocio.
2. Plan Estratégico de Sistemas.
3. Plan de Calidad del Negocio.
4. Plan de Contingencias.
5. Manual de Organización y Funciones
6. Lista de Comprobación

<Entregables>

1. Análisis y evaluación de riesgos.
2. Informe de la revisión.

6. Lista de Comprobación

Responda a cada una de las preguntas y anote el puntaje correspondiente de acuerdo a lo observado y verificado. Luego, obtenga el Puntaje Total sumando los puntajes asignados, dividir el Puntaje Total entre el número de preguntas, redondee el valor obtenido. Este valor compárelo con la tabla de valores del puntaje y conocerá la calificación de la fase revisada o inspeccionada.

Valores aplicados al Puntaje:

1= No tiene, no existe 2= Malo 3= Regular

4= Bueno 5= Excelente, Si tiene

Actividad (Pregunta)	**Puntaje** (1-5)
1. ¿Existe definido un Plan Estratégico basado en la calidad?	
2. ¿Está la misión, las metas y objetivos de la organización plenamente establecidos?	
3. ¿Están identificados los Factores Críticos del Negocio?	
4. ¿Existe cultura de Calidad en la organización?	
5. ¿Existe un Plan de Aseguramiento de la Calidad?	
6. ¿Existe una función de calidad en el Proceso de Software?	
7. ¿Está la función de calidad verdaderamente implicada en todas las funciones y asociada a las grandes decisiones de la empresa, o está relegada a una simple función de control sin poder de decisión?	
8. ¿Existe la función de Calidad incorporada en algún proceso de negocios de la empresa?	
9. ¿Se utilizan normas y estándares basadas en la Calidad?	
10. ¿Existe definido un Plan de Sistemas?	

7. Informe de Revisión

INFORME DE REVISION
<Destinatario>
<Fecha>

CONTENIDO

1. TERMINOS DE REFERENCIA
< > Información general o referencias sobre las actividades del negocio.
2. PUNTOS PARA ATENCION
< > Observaciones encontradas en la revisión de la organización de acuerdo a la lista de comprobación y riesgos detectados en los que se puede incurrir.
3. RECOMENDACIONES
< > Recomendaciones para atenuar o eliminar los riesgos detectados.
4. PERSONAL PARTICIPANTE
< > Personal que ha participado para llevar a cabo la revisión.
5. RECURSOS EMPLEADOS
< > Registra los recursos humanos (horas/hombre), materiales, servicios y gastos empleados.

Procedimiento
4.4.2 Revisión de la Planificación del Proyecto

Esta revisión es importante ya que permitirá conocer si la conceptualización del sistema es la correcta. Establecer si el Líder del Proyecto estableció correctamente los recursos a utilizar, la estimación de esfuerzos, cronograma de actividades, criterios de aceptación y riesgos de tipo técnico / negocio / normas gubernamentales propios de un proyecto.
En este formato se deberá registrar información de las personas que aprobaron el procedimiento, cambios que se han efectuado y los departamentos en los cuales debe ser aplicado.

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Cambios

Revisión	Fecha	Persona Responsable	Descripción del Cambio
A		Nombre	Revisión Inicial

Lista de Distribución

(lista de los departamentos que reciben copia del procedimiento)

1. Propósito

- Evaluar si el proyecto propuesto satisface los objetivos y requerimientos del usuario.
- Evaluar e identificar en el proyecto los riesgos de tipo técnico, de negocio, de políticas, entre otros.
- Verificar los Criterios de Aceptación del Software o Producto Final.
- Informar los riesgos potenciales encontrados durante la revisión y recomendar las acciones correctivas necesarias.
- Establecer los procedimientos para examinar, y revisar la información y documentación obtenida para asegurar la calidad del proyecto.
- Mantener un registro de la revisión de la Planificación del Proyecto y documentar las actividades de la revisión.

2. Alcance

Este procedimiento se aplica a los proyectos del negocio que se encuentran en pleno desarrollo, para conocer el ambiente de control y riesgos que puedan existir durante la etapa de ejecución.

3. Responsabilidades

<*Usuario, Gerente de Area, Dpto*> Proporciona los objetivos, metas y requerimientos funcionales.

<*Gerente de Proyecto, de Sistemas*> Proporciona información y documentos solicitados por <*Gerente o Supervisor de ACS*>.

<*Gerente de ACS*> Solicita información del proyecto, evalúa los resultados obtenidos en la revisión, informa recomendaciones y controla su implantación.

<*Supervisor de ACS*> Solicita, revisa y evalúa información del proyecto, identifica riesgos y elabora recomendaciones para una mejora en la gestión del

proyecto. Asimismo registra la información obtenida en la revisión y los resultados obtenidos su evaluación.

4. Procedimiento

4.1 General

<Gerente o Supervisor de ACS> Revisa nuevas formas de gestión y planificación de proyectos.

4.2 Identifica los Objetivos del Proyecto

<Supervisor de ACS> cuantifica, determina y verifica los objetivos a cumplir por el sistema, estén basados en los requerimientos funcionales de los *<Usuarios, Gerentes de Area, Dpto>*. Identifica los límites del proyecto, con respecto a los *<departamento1, departamento2,...* > implicados y operaciones implicadas.

<Supervisor de ACS> verifica si se han identificado todos los riesgos del proyecto ya sean de tipo: técnico, de negocio, de políticas, legales, entre otros, que impidan el normal desarrollo del proyecto.

<Supervisor de ACS> determina si se han establecido las metas y ventajas competitivas que se pueden obtener con el cumplimiento del proyecto.

4.3 Recopila información y documentos relacionados al proyecto

<Gerente o Supervisor de ACS> solicita información y documentación relacionada al proyecto mediante un *<documento>*.

<Gerente del Proyecto> recepciona el *<documento>* por el cual el *<Gerente o Supervisor de ACS>* solicita la información o documentos relacionados al proyecto. Responde la solicitud haciendo entrega de la información y/o documentación solicitada.

<Gerente o Supervisor de ACS> revisa que la información y/o documentación solicitada este completa. En caso contrario indagar el(los) motivo(s) por el que no se ha proporcionado la información o documentación requerida. Elabora una lista de la información y/o documentos faltantes e informar al *<Gerente de Sistemas, General>*.

4.4 Evalúa la Configuración

<Analista o Supervisor de ACS> evalúa y verifica que el hardware seleccionado corresponda a las necesidades del sistema a desarrollar, teniendo especial énfasis en características especiales necesarias para el sistema; tales como: longitud de la palabra, tamaño del número real, velocidad del procesador, tamaño del almacenamiento, sistema operativo, entre otros.

<Analista o Supervisor de ACS> evalúa y verifica que el lenguaje seleccionado para el desarrollo del software sea el adecuado; así como, el software accesorio necesario para el normal funcionamiento del sistema.

4.5 Revisa la Información y documentos obtenidos

<Gerente o Supervisor de ACS> revisa la información y/o documentación obtenida del proyecto para obtener un conocimiento cabal, sobre la planificación del proyecto, equipo que conforma el proyecto, estimación de esfuerzos, estimación de recursos, posibles riesgos, cronograma de actividades entre otros.

4.6 Verifica los Criterios de Aceptación

<Gerente o Supervisor de ACS> verifica o determina los criterios de aceptación del software o producto final. Verifica que se encuentre acordes a los objetivos y requisitos del proyecto y/o negocio.

4.7 Aplica lista de comprobación

<Analista o Supervisor de ACS> solicita una reunión con el *<Gerente o Jefe del Proyecto>* con el fin de indagar o tener mayor conocimiento del proyecto.

<Analista o Supervisor de ACS> efectúa una serie de preguntas al *<Gerente o Jefe del Proyecto>* anotando cada una de sus respuestas.

4.8 Evalúa y mide los Riesgos

<Gerente o Supervisor de ACS> analiza los datos obtenidos, basándote en el conocimiento obtenido acerca de cómo se ha elaborado o viene elaborando el proyecto.

<Gerente o Supervisor de ACS> Verifica y evalúa que el plan del proyecto este de acuerdo a los objetivos formulados. Verifica y evalúa que las estimaciones propuestas de tiempo, esfuerzo, recursos, entre otros hallan sido calculadas basadas en modelos o técnicas actuales y correctamente evaluados. Evalúa y mide la magnitud de los riesgos del proyecto y su manejo.

<Gerente o Supervisor de ACS> efectúa una evaluación de los resultados obtenidos; luego, desarrolla un análisis de los riesgos en que se pueda estar incurriendo en el proyecto.

<Gerente o Supervisor de ACS> define la prioridad de los riesgos del proyecto. Indica las recomendaciones necesarias para atenuar o eliminar los riesgos del proyecto.

4.9 Informa sobre la revisión

<Gerente o Supervisor de ACS> informa al *<Gerente de Proyecto, de Sistemas>* los riesgos, su prioridad, su impacto, técnicas de control y recomendaciones a implantar en el proyecto.

<Gerente o Jefe del Proyecto> lee las observaciones efectuadas por el *<Gerente o Supervisor de ACS>*, levanta las observaciones implantando las recomendaciones.

<Gerente o Supervisor de ACS> controla que las recomendaciones hayan sido implantadas.

4.10 Registra los resultados obtenidos

<Gerente o Supervisor de ACS> registra los resultados obtenidos en *<formatos>* o *<archivos>* los cuales almacena en *<almacén>*. En caso de tener un mecanismo automatizado registrar los datos e información pertinente. Actualiza el Plan de Aseguramiento de la Calidad para el proyecto.

5. Documentación relacionada

<Entradas>

1. Requerimientos del Negocio
2. Plan de Sistemas
3. Plan del Proyecto
4. Análisis de la estimación de esfuerzos, recursos.
5. Análisis de Riesgos
6. Cronograma de Actividades
7. Lista de Comprobación

<Entregables>

1. Análisis y evaluación de riesgos
2. Informe de la revisión

6. Lista de Comprobación

Responda a cada una de las preguntas y anote el puntaje correspondiente de acuerdo a lo observado y verificado. Luego, obtenga el Puntaje Total sumando los puntajes asignados, dividir el Puntaje Total entre el número de preguntas,

redondee el valor obtenido. Este valor compárelo con la tabla de valores del puntaje y conocerá la calificación de la etapa revisada o inspeccionada.

Valores aplicados al Puntaje:

1= No tiene, no existe 2= Malo 3= Regular

4= Bueno 5= Excelente, Si tiene

Actividad (Pregunta)	**Puntaje** (1-5)
Planificación 1. ¿Está el objetivo del proyecto plenamente establecidos? 2. ¿Son consistentes los objetivos y fines establecidos para el software con los objetivos y fines del Sistema? 3. ¿Están las funciones principales y el alcance del sistema están definidos sin ambigüedades? 4. ¿Están identificados los Factores Críticos de Exito del Proyecto? 5. ¿Están claramente establecidas las Restricciones del Proyecto? 6. ¿Es la solución Técnicamente Factible(Análisis Costo - Beneficio)? 7. ¿Se ha elegido la mejor alternativa? 8. ¿Existe un Análisis de Riesgos para el Proyecto? 9. ¿Existe un Plan de Aseguramiento de Calidad para Proyecto de Software? 10. ¿Existe un Plan de Administración de Configuración para el Proyecto? 11. ¿Existe un Plan de Pruebas de Validación y Verificación del Software? 12. ¿Se ha elaborado los Criterios de Aceptación del Producto Final? 13. ¿Los Criterios de Aceptación son conformes de acuerdo a los requisitos del negocio y objetivos del proyecto? **14.** ¿Existe la plataforma hardware y software adecuado para nuestro proyecto? **15.** ¿Se han planteado los aspectos de rendimiento, fiabilidad y facilidad de mantenimiento?	
Organización 1. ¿Se ha definido la organización del equipo de proyecto? 2. ¿Según la forma de organización, permite la comunicación entre todos los integrantes del proyecto y los miembros de las áreas operativas del negocio? 3. ¿El personal es calificado? 4. ¿El personal esta capacitado en las herramientas que se utilizaran en el proyecto? **5.** ¿Se dispone de suficiente personal? ¿Se ha previsto la rotación de personal?	

Estimación 1. ¿ Están definidos los Recursos para el Proyecto? 2. ¿Se ha efectuado una estimación de los recursos humanos, hardware y software a emplear en el proyecto? 3. ¿Se han estimado los costos ha incurrir en el proyecto? **4.** ¿Se tiene herramientas automáticas de estimación?	
Evaluación y Control 1. ¿Se ha identificado definido y estandarizado los puntos de control en cada proceso? 2. ¿Se ha definido la forma de medición en los puntos de control identificados? 3. ¿Existe el Cronograma para el Desarrollo del Proyecto? **4.** ¿Corresponde la complejidad funcional del sistema con el riesgo de desarrollo, costos y cronograma de actividades?	
Equipo 1. ¿Qué hardware se adapta mejor a las funciones especificadas? 2. ¿Qué hardware se puede comprar? ¿Cuáles son los proveedores, la disponibilidad y el costo? 3. ¿Qué interfaces se requieren?	
Estándares 1. ¿ Se va a trabajar con una metodología? **2.** ¿ Se tienen estándares para las diferentes etapas del Desarrollo del Proyecto?	
Capacitación 1. ¿Existe un Plan de Capacitación? 2. ¿Existe un suficiente personal entrenado? 3. ¿Tiene el suficiente nivel técnico el Equipo de Proyecto? 4. ¿Tienen el jefe de proyecto experiencia en la Gestión de Proyectos? 5. ¿Tienen experiencia en las herramientas de desarrollo los programadores?	
Documentación 1. ¿La documentación del proyecto es entendible? 2. ¿La documentación del proyecto es fácil de usar? 3. ¿La documentación del proyecto se puede mantener o actualizar fácilmente? 4. ¿El tiempo para entender la documentación es elevado?	

7. Informe de Revisión

INFORME DE REVISION
<Destinatario>
<Fecha>

CONTENIDO

1. TERMINOS DE REFERENCIA
< > Información general sobre el proyecto.
2. PUNTOS PARA ATENCION
< > Observaciones encontradas en la revisión de la planificación del proyecto de acuerdo a la lista de comprobación y riesgos detectados.
3. RECOMENDACIONES
< > Recomendaciones para atenuar o eliminar los riesgos detectados.
4. PERSONAL PARTICIPANTE
< > Personal que ha participado para llevar a cabo la revisión.
5. RECURSOS EMPLEADOS
< > Registra los recursos humanos (horas/hombre), materiales, servicios y gastos empleados.

8. Formatos

<Plan del Proyecto> Es una plantilla que servirá para verificar el contenido del documento entregado por el *<Equipo de Desarrollo>*. Se mencionan los puntos que deberá tener dicho documento para verificar que este se encuentre completo.

Q Z **Plan del Proyecto** Num: ____Pag.:1/n

1. Introducción
   - 1.1 Propósito
   - 1.2 Alcance
   - 1.3 Definiciones, abreviaturas
   - 1.4 Principales Funciones
   - 1.4 Restricciones Técnicas y de Gestión
   - 1.5 Referencias
2. Organización del Proyecto
   - 2.1 Modelo del Proceso
   - 2.2 Estructura Organizacional
   - 2.3 Responsabilidades
   - 2.4 Equipo de Trabajo (Gerente, Jefes, Analistas, Programadores y otros)
   - 2.5 Areas involucradas (Gerente, Jefe, personal operativo)
3. Administración del Proyecto
   - 3.1 Propósito
   - 3.2 Consideraciones y Prioridades
   - 3.3 Estimación de Recursos
   - 3.4 Gestión de Riesgos
   - 3.5 Mecanismos de Control y Monitoreo
   - 3.6 Cronograma de Actividades
4. Recursos Técnicos
   - 4.1 Metodologias, Métodos y Técnicas
   - 4.2 Herramientas de Software (Sistema Operativo, Lenguaje Visual, de Pruebas, de Gestión de Configuración, entre otros).
   - 4.3 Equipos / Hardware (Estaciones de Trabajo, Servidores, Líneas de comunicación, modem's, interfaces, entre otros).
   - 4.4 Soporte
5. Consideraciones para el Análisis
6. Criterios de Aceptación

QZ

Procedimiento
4.3.3 Revisión del Contrato

Es una revisión importante ya que se pone en juego aspectos; tales como: dinero, tiempo, recursos, perdida de nuevos negocios. Se deberá observar con atención el contenido del contrato ya que deberá contener en forma completa aspectos de tipo técnico, económico, recursos, tiempos, responsabilidades y sobre todo Criterios de Aceptación y Validación para llegar a la conformidad con el Cliente o Usuario.
En este formato se deberá registrar información de las personas que aprobaron el procedimiento, cambios que se han efectuado y los departamentos en los cuales debe ser aplicado.

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Cambios

Revisión	Fecha	Persona Responsable	Descripción del Cambio
A		Nombre	Revisión Inicial

Lista de Distribución

(lista de los departamentos que reciben copia del procedimiento)

1. Propósito

- Asegurar que los requerimientos en cada contrato estén definidos y documentados.

- Asegurar que las especificaciones de los requerimientos, términos y condiciones estén adecuadamente definidos.

- Evaluar el manejo de los cambios del cliente durante el desarrollo o implantación del software. Verificar su inclusión en los términos del contrato.

- Evaluar los criterios de aceptación y verificar su inclusión en los términos del contrato.

- Evaluar el manejo de los problemas detectados después de la aceptación, reclamos y quejas relacionados a la calidad. Verificar su inclusión en los términos del contrato.

- Verificar el rol del Usuario en la especificación de requerimientos, instalación y aceptación. Verificar si se incluye en los términos del contrato.

- Verificar las facilidades, herramientas y software a ser proporcionado por el Comprador. Verificar que su inclusión en los términos del contrato.

- Verificar los estándares, procedimientos y metodología a ser utilizados por el Proveedor. Verificar su inclusión en los términos del contrato.

- Establecer los procedimientos para examinar y revisar los requerimientos del contrato para asegurar que estén adecuadamente definidos.

- Mantener un registro de la revisión del contrato, proveer la autorización del contrato, y documentar las actividades de la revisión.

2. Alcance

Este procedimiento se aplica al software desarrollado por terceros o comprado a terceros. El software que se encuentra en catálogos de venta técnicos. Requerimientos especiales, que no se encuentran en los catálogos, siendo definidos por las especificaciones del Cliente y revisado por el Area de Sistemas y el Equipo de ACS.

3. Responsabilidades

<*Usuario*> Elabora los requerimientos funcionales y sus definiciones.

<*Area de Sistemas*> Recibe los requerimientos de los Usuarios. Elabora los requerimientos técnicos de acuerdo a los requerimientos funcionales del usuario. Proporciona los estándares, metodología de desarrollo, configuraciones de equipos, herramientas de trabajo entre otros.

<*Proveedor*> Desarrolla el software de acuerdo a los requerimientos del usuario. Proporciona información relativa al contrato y su plan de actividades.

<*Legal*> Proporciona el modelo de contrato y emite su opinión en términos legales del contrato elaborado.

<*Gerente de ACS*> Solicita información del proyecto, evalúa los resultados obtenidos en la revisión, informa recomendaciones y verifica se incluyan en el contrato.

<*Supervisor de ACS*> Revisa información del contrato para encontrar desviaciones con respecto a los requerimientos y estándares de la organización. Elabora informe de la revisión del contrato.

4. Procedimiento

4.1 General

Revisa catálogos y literatura del software ha adquirir.

<Supervisor de ACS> revisa los contratos.

4.2 Recopila información y documentos relacionados al Contrato

<Gerente o Supervisor de ACS> solicita información y documentación relacionada al contrato mediante un *<documento>*.

<Gerente del Proyecto> recepciona el *<documento>* por el cual el *<Gerente o Supervisor de ACS>* solicita la información o documentos relacionados al proyecto. Responde la solicitud haciendo entrega de la información y/o documentación solicitada.

<Gerente o Supervisor de ACS> revisa que la información y/o documentación solicitada este completa. En caso contrario indagar el(los) motivo(s) por el que no se ha proporcionado la información o documentación requerida. Elabora una lista de la información y/o documentos faltantes e informar al *<Gerente de Sistemas, General>*.

4.3 Revisa y verifica el contenido del Contrato

<Gerente o Supervisor de ACS> revisa los términos de referencia y anexos del contrato; así como la documentación del proyecto relacionada al contrato. Analiza de acuerdo a una lista de comprobación, anotando los resultados obtenidos.

<Gerente o Supervisor de ACS> verifica que el contrato contenga anexos o puntos en el cual se definan de manera completa los requerimientos, y especificaciones técnicas del software.

<Gerente o Supervisor de ACS> verifica que el contrato contenga anexos o puntos en que se indiquen los estándares y metodología que aplicará el *<Proveedor>* en el desarrollo o implementación del software.

<*Gerente o Supervisor de ACS*> verifica que el contrato contenga anexos o puntos en que se indiquen los procedimientos que utilizará el <*Proveedor*> para las pruebas de aceptación, manejo de cambios durante y de después la culminación del proyecto, manejo de problemas detectados antes y después de la aceptación.

<*Gerente o Supervisor de ACS*> verifica que el contrato contenga anexos o puntos en que se indique el rol del usuario en las diferentes fases del proyecto.

<*Gerente o Supervisor de ACS*> verifica que el contrato contenga anexos o puntos en que se indique las facilidades, herramientas y recursos que requiere el <*Proveedor*> para desarrollar o implementar el software.

<*Gerente o Supervisor de ACS*> verifica que el contrato contenga un cronograma de actividades en el cual se indiquen fechas de inicio y fin, y entregables.

<*Gerente o Supervisor de ACS*> verifica que el contrato contenga la obligaciones del <*Proveedor*> y <*Comprador*>.

<*Gerente o Supervisor de ACS*> verifica que existan criterios de evaluación y selección del <*Proveedor*> de acuerdo a: experiencia, capacidad, prácticas de calidad, fortaleza financiera, asistencia técnica; entre otros.

<*Gerente o Supervisor de ACS*> verifica que existan Criterios de Aceptación, de Evaluación y selección del <*Software*> de acuerdo a: referencias del producto o servicio, garantía del producto, entre otros.

4.4 Aplica lista de comprobación

<Analista o Supervisor de ACS> solicita una reunión con el *<Gerente o Jefe del Proyecto>* con el fin de indagar o tener mayor conocimiento sobre el contrato.

<Analista o Supervisor de ACS> efectúa una serie de preguntas al *<Gerente o Jefe del Proyecto>* anotando cada una de sus respuestas.

4.5 Evalúa y mide los Riesgos

<Gerente o Supervisor de ACS> analiza los datos obtenidos, basándote en el conocimiento obtenido acerca de cómo se ha elaborado o viene elaborando el contrato con respecto a los requerimientos del proyecto relacionado.

<Gerente o Supervisor de ACS> Verifica y evalúa que el contrato este de acuerdo a los objetivos formulados en el proyecto relacionado. Evalúa y mide la magnitud de los riesgos detectados en el contrato.

<Gerente o Supervisor de ACS> efectúa una evaluación de los resultados obtenidos; luego, desarrolla un análisis de los riesgos en que se pueda estar incurriendo en el contrato.

<Gerente o Supervisor de ACS> Indica las recomendaciones necesarias para atenuar o eliminar los riesgos del contrato.

4.6 Informa sobre la revisión

<Gerente o Supervisor de ACS> informa al *<Gerente General o Director Principal>* los riesgos y recomendaciones a incorporar en el contrato.

<Gerente General o Director Principal> lee las observaciones efectuadas por el *<Gerente o Supervisor de ACS>* y apoya la incorporación de las recomendaciones.

<Gerente o Supervisor de ACS> controla que las recomendaciones hayan sido incorporadas.

4.7 Registra los resultados obtenidos

<Gerente o Supervisor de ACS> registra los resultados obtenidos en *<formatos>* o *<archivos>* los cuales almacena en *<almacén>*. En caso de tener un mecanismo automatizado registrar los datos e información pertinente. Actualiza el Plan de Aseguramiento de la Calidad para el proyecto.

5. Documentación relacionada

<Entradas>

1. Plan del Proyecto
2. Análisis de la estimación de esfuerzos, recursos.
3. Cronograma de Actividades
4. Lista de Comprobación

<Entregables>

1. Análisis y evaluación de riesgos
2. Informe de la revisión

6. Lista de Comprobación

Responda a cada una de las preguntas y anote el puntaje correspondiente de acuerdo a lo observado y verificado. Luego, obtenga el Puntaje Total sumando los puntajes asignados, dividir el Puntaje Total entre el número de preguntas, redondee el valor obtenido. Este valor compárelo con la tabla de valores del puntaje y conocerá la calificación de la fase revisada o inspeccionada.

Valores aplicados al Puntaje:

1= No tiene, no existe	2= Malo	3= Regular
4= Bueno	5= Excelente, Si tiene	

Actividad (Preguntas)	**Puntaje** (1-5)
Financiero 1. ¿Puede obtenerse el estado financiero actual del Proveedor para ser examinado? 2. ¿Los estados financieros del proveedor son confiables, se han auditado en forma independiente? **3.** ¿Tiene el Proveedor o cualquiera de sus asociados problemas de quiebra o litigación?	
Experiencia y Capacidades 1. ¿Solicita por función de trabajo el número de personal y su experiencia? 2. ¿Solicita los nombres de los vendedores, representantes técnicos y personal de soporte? ¿Pueden ser entrevistados? 3. ¿Solicita los productos de software que el proveedor vende y el número de instalaciones de cada uno de ellos. **4.** ¿Solicita una lista de usuarios del software?	
Desarrollo y Control de Procesos 1. ¿Se usaran prácticas y estándares en el desarrollo o instalación del software? 2. ¿Las prácticas y estándares son los adecuados en el desarrollo o instalación del software? ¿Son corrientemente usados en sus procesos? 3. ¿Se podrá disponer de la documentación? 4. ¿Se ha planeado la conversión o modificación del software? 5. ¿Cómo se probará lo cumplido, existen criterios para la aceptación? 6. ¿Cómo se efectuaran los cambios en los requerimientos durante el desarrollo? 7. ¿Cuáles son las acciones a tomar ante problemas detectados después de aceptación, relacionados a la calidad o quejas efectuadas? 8. ¿Cómo se efectuará la verificación de la exactitud e integridad de copias del software? 9. ¿Se incluye un calendario de actividades, incluyendo horas extras, fines de semana y feriados? 10. ¿Se proporcionan fechas y costos por cada entregable? 11. ¿Se Incluye en el contrato la forma de aprobación de cada etapa de la instalación y/o desarrollo del software? **12.** ¿Se considera la complejidad del proyecto y el riesgo en alcanzar los requerimientos del contrato?	
Ayuda Técnica 1. ¿Qué asistencia se provee al efectuar la instalación? 2. ¿Se indica donde se realizara el entrenamiento al Usuario del software? 3. ¿El software y su documentación podrán ser modificados de acuerdo a los requerimientos del Usuario? 4. ¿Quién hará los cambios al software y la documentación?	

5. ¿Las modificaciones invalidan la garantía? 6. ¿Los nuevos releases (software y documentación) son planeados o en proceso? 7. ¿Los futuros releases serán proporcionados por el Proveedor durante el periodo de garantía? ¿Serán sin costo adicional?	
Prácticas de Calidad 1. ¿Cómo se efectuará el seguimiento del desarrollo y control de procesos? 2. ¿Se efectuara una revisión de los requerimientos, diseño, y codificación? 3. ¿Las revisiones a los requerimientos, diseño y codificación son utilizadas y efectivas? 4. ¿El proceso de acción correctiva establece el manejo y solución de errores? 5. ¿Se entregarán los procedimientos instalación y customización del software?	
Servicio de Mantenimiento 1. ¿Existe una garantía por escrito acerca del nivel y calidad del servicio de mantenimiento a proveer? ¿Quién proveerá el soporte al software? 2. ¿Se suministrará actualizaciones y correcciones de errores con su respectiva documentación? 3. ¿Será proporcionada la documentación necesaria para el mantenimiento del software por parte del Comprador? 4. ¿Quién y cómo llevará a cabo las actualizaciones y correcciones de errores? 5. ¿Qué tiempo se esperará para la atención en la corrección de errores? 6. ¿El soporte técnico estará disponible las 24 horas del día? 7. ¿Qué se incluye en el soporte técnico? a. correcciones de errores b. modificaciones c. nuevas versiones del software d. actualización de la documentación del usuario e. asistencia en la instalación f. entrenamiento	
Producto 1. ¿Esta estipulado una demostración del funcionamiento del software? 2. ¿Las especificaciones técnicas del software satisfacen los requerimientos del Comprador? 3. ¿Se proporcionara la documentación técnica para ser examinada? 4. ¿Se indica la versión, release o genlevel del software? ¿Es la última versión? 5. ¿Se indica el número de copias, instalaciones o licencias del software? 6. ¿Se indica el tipo de medio a entregar cada elemento del software, incluyendo: formato aceptable, versión?	

7. ¿Se indica el idioma que maneja el software y en el que esta escrita la documentación? 8. ¿Tiene una certificación de poder trabajar el año en un formato de cuatro dígitos? 9. ¿Se mencionan el copyright del autor y/o licencias acordadas? 10. ¿Se indica la documentación requerida, tales como: manuales de diseño y manuales de usuario, entre otros?. 11. ¿Se facilitarán los accesos (claves de instalación, contraseñas) del software? 12. ¿El código fuente será entregado por el proveedor para que se puedan hacer modificaciones? 13. ¿El software requerirá cambios cuando se cambie el sistema operativo? ¿Si es así, cómo serán cumplidos los cambios? **14.** ¿El desarrollo del software tendrá continuidad? ¿Podrá discontinuarse en el futuro?	
Garantía del producto 1. ¿El período de la garantía es incondicional? 2. ¿Si no, existe una garantía? 3. ¿Después de hacer una implementación exitosa de acuerdo a las pruebas de aceptación se inicia el período de la garantía incondicional? 4. ¿El período de garantía incondicional provee un nivel especificó de ejecución de los productos del software por un período dado según las premisas donde se instala? **5.** ¿Cuál es el tiempo de la garantía incondicional?	
Costos 1. ¿Cuál es el precio al que esta disponible? 2. ¿Cuales son los términos de la licencia y renovaciones? 3. ¿Qué se incluye en el precio de adquisición o licencias? 4. ¿Cuál es el costo, si cualquier, se asocia por un período de garantía incondicional? 5. ¿Cuál es el costo de mantenimiento después del período de garantía? 6. ¿Cuál es el costo de las modificaciones? 7. ¿Cuál es el costo de los perfeccionamientos? **8.** ¿Las actualizaciones y correcciones de error se proveen sin ningún costo?	
Formulación 1. ¿Se usó un contrato standard? 2. ¿Puede obtener el contrato previamente para su examinación? 3. ¿Los términos del contrato son negociables? 4. ¿Están incluidos los Criterios de Aceptación del Software o Producto Final? **5.** ¿Está incluido el royalty?	
Forma de Pago 1. ¿Se indican los gastos de impresión asociados con la publicación de la documentación del usuario?	

2. ¿Se indican los gastos por viajes y viáticos ha incurrir en el desarrollo o instalación? 3. ¿Cuál es la frecuencia y cantidad de pagos al proveedor? ¿Esta coincide con los resultados de las pruebas o realizaciones? 4. ¿Existe un método de determinar los pagos del incentivo asociados con resultados significantes, realizaciones, costos, o cronogramas? 5. ¿Se limitan los pagos a las copias de los productos del software y entregables realmente proveídos por el Proveedor y no a una cantidad de licencias o volumen de dinero? 6. ¿Se ha especificado la forma de detener o recuperar el pago por trabajos incompletos o inaceptables? 7. ¿Se ha especificado la reducción del pago si no se encuentran los requerimientos o cualquier entregable establecido en el contrato?	
Obligaciones del Comprador y del Proveedor 1. Definición de Plan del Desarrollo del Software. a. ¿Fueron cumplidos los pasos del desarrollo identificados por el proveedor? b. ¿Existe un producto (entregable) incluido al final de cada paso para demostrar que se ha completado el paso satisfactoriamente, ejm: estudios, estudio de factibilidad, general y detalle de diseño, datos de prueba y prueba del plan, programas actuales, manuales del usuario, publicaciones de soporte y resultados de las pruebas de integración/ aceptación? c. ¿Se debe satisfacer los pasos previos antes de continuar con el próximo paso identificado en el desarrollo del proyecto? d. ¿Las obligaciones del Comprador y del Proveedor se incluyeron en algún documento? 2. Quién es responsable por: a. ¿Publicación y gastos de la documentación del usuario? b. ¿Publicación de las versiones? c. ¿Distribución del software a los usuarios finales? d. ¿Avisos e informes, si se especificó? e. ¿Nuevo software que reemplaza al antiguo? f. Existe un representante por: Proveedor Comprador 3. ¿Se ha especificado el rol del usuario para la especificación de requerimientos, instalación y aceptación?. 4. ¿El Comprador proporcionará facilidades, herramientas e items de hardware o software al Proveedor? ¿Se ha especificado en algún documento? 5. ¿El Proveedor efectuará la custodia de copias backup de los programas fuentes, documentos, librerías y otros elementos del software entregados, incluyendo un plan de contingencias? 6. ¿El Proveedor dispone de personal con experiencia en el software y/o productos a utilizar y/o instalar? 7. ¿El entrenamiento al usuario será proveído por el Proveedor? 8. ¿El Proveedor proporcionara el soporte adecuado durante	

instalaciones iniciales del software? **9.** ¿El Proveedor se responsabiliza de proveer soporte técnico oportunamente?	

7. Informe de Revisión

INFORME DE REVISION
<Destinatario>
<Fecha>

CONTENIDO

1. TERMINOS DE REFERENCIA
< > Información general del contrato, referencias del proveedor, del proyecto, del producto.
2. PUNTOS PARA ATENCION
< > Observaciones encontradas en la revisión del contrato de acuerdo a la lista de comprobación, riesgos en los que se puede incurrir.
3. RECOMENDACIONES
< > Recomendaciones para que se puedan cumplir normas técnicas y legales vigentes. Así como el cumplimiento de los requerimientos del Usuario.
4. PERSONAL PARTICIPANTE
< > Personal que ha participado para llevar a cabo la revisión.
5. RECURSOS EMPLEADOS
< > Registra los recursos humanos (horas/hombre), materiales, servicios y gastos empleados.

Procedimiento

4.4.4 Revisión de Estándares

Su revisión es de suma importancia ya que los Estándares deben reflejar la mejor, fácil y segura forma de realizar un proceso, permitiendo medir el desempeño y niveles de calidad, evitando de esta manera la recurrencia de errores. Se deberá verificar su adecuada documentación para preservar el conocimiento y la experiencia, y sirvan de base para el entrenamiento del personal.
En este formato se deberá registrar información de las personas que aprobaron el procedimiento, cambios que se han efectuado y los departamentos en los cuales debe ser aplicado.

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Cambios

Revisión	Fecha	Persona Responsable	Descripción del Cambio
A		Nombre	Revisión Inicial

Lista de Distribución

(lista de los departamentos que reciben copia del procedimiento)

1. Propósito

- Solicitar la actualización y mejora de los estándares utilizados por la organización.
- Establecer los procedimientos para examinar y revisar que los estándares estén adecuadamente definidos y documentados.
- Verificar que los Estándares permitan los apropiados controles durante el proceso de desarrollo del Software y adecuados rastros de auditoria.
- Verificar el cumplimiento de los estándares durante el desarrollo y/o ejecución de los Proyectos.
- Informar los riesgos potenciales encontrados durante la revisión y recomendar las acciones correctivas necesarias.
- Mantener un registro de la revisión de estándares y documentar las actividades de la revisión.

2. Alcance

Este procedimiento se aplica a los estándares utilizados en el desarrollo o implementación de software. Entiéndase por estándar a todo tipo de norma, formato, documento, símbolo, procedimiento, uso de metodología; entre otros, utilizados en el desarrollo o implementación de software.

3. Responsabilidades

<Area de Sistemas> Aplicará los estándares definidos por la organización en el desarrollo o implementación de software.

<Gerente de ACS> Solicita información del proyecto, evalúa los resultados obtenidos en la revisión, informa recomendaciones y controla su implantación.

<Supervisor de ACS> Revisa los estándares para encontrar defectos recomendando su mejora o actualización de acuerdo a los cambios tecnológicos u organizativos. Elabora informe de la revisión del contrato.

4. Procedimiento

4.1 General

<Gerente o Supervisor de ACS> revisa y evalúa nuevos estándares con el fin de adaptarlos o incorporarlos en la organización.

4.2 Verifica la existencia de Estándares

<Gerente o Supervisor de ACS> verifica la existencia de estándares y su aplicación en las actividades relacionadas al proceso de desarrollo e implementación de software. Se debe verificar que no solo existan estándares de tipo tecnológico, sino también de tipo administrativo y operativos.

<Gerente o Supervisor de ACS> verifica que los estándares se encuentren almacenados en *<almacén>*.

4.3 Revisa y evalúa los Estándares

<Gerente o Supervisor de ACS> Verifica si en algún proceso no se aplican estándares. Evalúa si cumplen la finalidad por la cual han sido implementados en el proceso de desarrollo de software.

<Gerente o Supervisor de ACS> Evalúa si los estándares representan la mejor, más fácil y más segura forma de realizar un trabajo. Verifica si ofrecen la mejor manera de preservar el conocimiento y la experiencia. Verifica si suministran una manera de medir el desempeño.

<Gerente o Supervisor de ACS> Evalúa si los estándares proporcionan una base para el mejoramiento, auditoria, diagnóstico y detección de errores.

4.4 Aplica lista de comprobación

<Analista o Supervisor de ACS> solicita una reunión con el *<Gerente o Jefe del Proyecto>* con el fin de indagar o tener mayor conocimiento sobre los estándares.

<Analista o Supervisor de ACS> efectúa una serie de preguntas al *<Gerente o Jefe del Proyecto>* anotando cada una de sus respuestas.

4.5 Evalúa y mide los Riesgos

<Gerente o Supervisor de ACS> evalúa el estándar aplicando una lista de comprobación basada en el conocimiento y la experiencia.

<Gerente o Supervisor de ACS> mide la magnitud de los riesgos detectados en los estándares. Efectúa una evaluación de los resultados obtenidos; luego, desarrolla un análisis de los riesgos detectados en los estándares.

<Gerente o Supervisor de ACS> indica las recomendaciones necesarias para eliminar los riesgos en el uso de los estándares actualmente utilizados, o implementar estándares en los procesos en que no se utilicen, o cambiarlos por otros que cumplan con mejorar los procesos.

4.6 Registra los resultados obtenidos

<Gerente o Supervisor de ACS> informa al *<Gerente General o Director Principal>* los riesgos y recomendaciones a incorporar en los estándares.

<Gerente General o Director Principal> lee las observaciones efectuadas por el *<Gerente o Supervisor de ACS>* y apoya la incorporación de las recomendaciones.

<Gerente o Supervisor de ACS> controla que las recomendaciones hayan sido incorporadas.

4.7 Informa sobre la revisión

<Gerente o Supervisor ACS> registra los resultados obtenidos en *<formatos>* o *<archivos>* los cuales almacena en *<almacén>*. En caso de tener un mecanismo automatizado registrar los datos e información pertinente. Actualiza y mejora los estándares.

5. Documentación relacionada

<Entradas>

1. Estándares de la Organización para el Area de Sistemas
2. Lista de Comprobación

<Entregables>

1. Análisis y evaluación de riesgos
2. Informe de la revisión

6. Lista de Comprobación

Responda a cada una de las preguntas y anote el puntaje correspondiente de acuerdo a lo observado y verificado. Luego, obtenga el Puntaje Total sumando los puntajes asignados, dividir el Puntaje Total entre el número de preguntas, redondee el valor obtenido. Este valor compárelo con la tabla de valores del puntaje y conocerá la calificación de la fase revisada o inspeccionada.

Valores aplicados al Puntaje:

1= No tiene, no existe 2= Malo 3= Regular

4= Bueno 5= Excelente, Si tiene

Actividades (Preguntas)	**Puntaje** (1-5)
1. ¿Existen Estándares definidos? 2. ¿Se aplican los estándares existentes? 3. ¿Los estándares utilizados son los adecuados para la elaboración de proyectos?	

4. ¿Los estándares actuales permiten medir y evaluar el desempeño? 5. ¿Se ha medido el impacto de no aplicar los estándares? 6. ¿Suministra una base para el mantenimiento y mejoramientos? 7. ¿La documentación de los estándares es inadecuada? 8. ¿Personal del Proyecto tiene conocimiento y entrenamiento de los estándares? 9. ¿Existe una metodología para el desarrollo e implementación del sistema?	

7. Análisis de Medición

Aplica las siguientes métricas, de acuerdo a las indicaciones dadas en el Anexo B:

- Indice de Defectos.
- Densidad de Defectos
- Estadística de Defectos por Fase.
- Horas Hombre por principales Defectos detectados.

8. Informe de Revisión

INFORME DE REVISION
<Destinatario>
<Fecha>

CONTENIDO

1. TERMINOS DE REFERENCIA
< > Información general de los estándares utilizados en la organización concernientes con el desarrollo / mantenimiento / implementación / explotación de software.
2. PUNTOS PARA ATENCION
< > Observaciones encontradas en la revisión de Estándares de acuerdo a la lista de comprobación, riesgos en los que se puede incurrir.
3. RECOMENDACIONES
< > Recomendaciones para que se puedan cumplir normas técnicas actuales.
4. PERSONAL PARTICIPANTE
< > Personal que ha participado para llevar a cabo la revisión.
5. RECURSOS EMPLEADOS
< > Registra los recursos humanos (horas/hombre), materiales, servicios y gastos empleados.

QZ

Procedimiento
4.5 Revisión del Análisis de Requerimientos

Es primordial verificar la especificación de los requerimientos y los usos del usuario para el sistema ya que será la base para la creación de un software correcto, confiable y completo.
En este formato se deberá registrar información de las personas que aprobaron el procedimiento, cambios que se han efectuado y los departamentos en los cuales debe ser aplicado.

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Cambios

Revisión	Fecha	Persona Responsable	Descripción del Cambio
A		Nombre	Revisión Inicial

Lista de Distribución

(lista de los departamentos que reciben copia del procedimiento)

1. Propósito

- Manejar los cambios del cliente durante el ciclo de desarrollo.
- Mantener un registro de la revisión de la especificación requerimientos y documentar las actividades de la revisión.
- Establecer los procedimientos para examinar, revisar y asegurar que la Especificación de los Requerimientos estén adecuadamente definidos.
- Seguimiento y verificación que los requerimientos establecidos en el proyecto hayan sido considerados en la Especificación de los Requerimientos.
- Analizar y verificar que la especificación de requerimientos este correcta, consistente, completa y exacta.

2. Alcance

Este procedimiento se aplica al software desarrollado por terceros o comprado a terceros. Es software que se encuentra en catálogos de venta técnicos. Requerimientos especiales, que no se encuentran en los catálogos, siendo definidos por las especificaciones del Cliente y revisado por el Area de Sistemas y el equipo de ACS.

3. Responsabilidades

<*Usuario*> Elabora los requerimientos funcionales y sus definiciones. Comunica los requerimientos al <*Equipo del Proyecto*> por medio de <*documento*>.

<*Jefe de Area o Dpto*> aprueba la especificación del software.

<*Area de Sistemas*> Elabora los requerimientos técnicos de acuerdo a los requerimientos funcionales del usuario. Proporciona los estándares,

metodología de desarrollo, configuraciones de equipos, herramientas de trabajo entre otros.

<Analista o Supervisor de ACS> Revisa información del contrato para encontrar desviaciones con respecto a los requerimientos y estándares de la organización. Elabora informe de la revisión del contrato.

4. Procedimiento

4.1 General

<Analista o Supervisor de ACS> verifica el uso de una metodología y estándares acordes al desarrollo de software o herramientas utilizadas.

4.2 Verifica las fuentes del Requerimiento

<Analista o Supervisor de ACS> verifica el estado de cada uno de los requerimientos asignados, propuestos, aprobados e incorporados dentro del sistema.

<Analista o Supervisor de ACS> verifica que las fuentes que dan origen a los requerimientos puedan ser identificadas, estas fuentes pueden venir de un refinamiento de requerimientos más abstracto o de una reunión con el usuario.

<Analista o Supervisor de ACS> verifica que la existencia de condiciones extras para el requerimiento original del usuario y pueda comprobarse su origen o fuente.

<Analista o Supervisor de ACS> establece relaciones entre la documentación y la jerarquía de los documentos.

4.3 Verifica la naturaleza del Requerimiento

<Analista o Supervisor de ACS> verifica si la especificación del requerimiento es particular para un producto de software, programa o

conjuntos de programas que trabajan en un ambiente especifico. Considera los siguientes temas básicos:

Funcionalidad: ¿Qué se supone hace el software?

Interfaces Externas: ¿Cómo interactua el software con las personas?

Performance: ¿Cuál es la velocidad, disponibilidad, tiempo de respuesta, tiempo de recuperación, etc.?

Atributos: ¿Cuáles son las consideraciones de portabilidad, corrección, mantenibilidad, seguridad?

Restricciones: Requerimientos estándares, lenguaje ha utilizar, políticas para la integridad de la base de datos, formato de los datos, recursos limitados, etc.

4.4 Verifica el ambiente del Requerimiento

<Analista o Supervisor de ACS> verifica que el papel que cumple el requerimiento en el proceso de desarrollo del software, debe verificarse que no vaya más allá de los límites de ese papel. Lo cual significa que:

El requerimiento debe estar correctamente definido, pueden existir caso en que se defina de acuerdo a la naturaleza de la tarea o características especiales del proyecto.

No debe describir características de diseño, ni detalles de implementación. Debe ser descrito en la fase de Diseño.

No debe imponerse condiciones adicionales en el software, estas deberán especificarse en otro documento.

4.5 Verifica las características del Requerimiento

<Analista o Supervisor de ACS> verifica que el requerimiento tenga las siguientes características:

- Correcto: No existe herramienta o procedimiento que asegure que este correcto. Para lo cual habrá que compararlo con alguna especificación superior, con otra documentación, con otros estándares. Alternativamente el usuario podrá decir si la especificación del requerimiento refleja correctamente las necesidades actuales.

- Sin Ambiguedad: Verifica que la redacción de los textos o descriptivas de la especificación del requerimiento tienen una sola interpretación.

- Completo: Verifica que la especificación del requerimiento incluya:

  - Todos los requerimientos significativos, relacionados a la funcionalidad, performance, base de datos, consideraciones en el diseño, atributos o interfaces externas.

  - Definición de las respuestas del software a toda clase de datos de entrada y situaciones generadas por el usuario.

  - Todas las figuras, tablas, diagramas, definiciones de términos y unidades de medida deben tener una descripción y referencias.

- Consistente: Verifica en la especificación de requerimiento, que los requerimientos individuales descriptos no tengan conflicto. Existen tres tipos de conflictos:

  - Algunas características especificadas del mundo - real tengan conflicto. Ejm: El color de una ventana se especifique como verde y en otra parte de la especificación como azul.

  - Dos acciones especificas estén en conflicto lógica o temporalmente. Ejm. Se describe procesar A y luego B, en otra parte de la especificación se describe que A y B se procesan simultáneamente.

- Dos o más requerimientos describen el mismo objeto del mundo - real usando diferentes términos para el objeto.

- Clasifica por importancia y/o estabilidad: Identifica la importancia de los requerimientos si son: esenciales, condicionales y opcionales. Identifica la estabilidad del requerimiento basándote en el número esperado de cambios que llegue a tener un requerimiento.

- Verificable: Verifica que por algún mecanismo hombre o máquina se pueda verificar que el software satisface los requisitos. En general cualquiera requisito ambiguo no es verificable. En caso que un requerimiento no sea verificable, recomienda su revisión o eliminación.

- Modificable: Verifica si la especificación del requerimiento al efectuar un cambio no altera su estructura y estilo. Pudiéndose efectuar el cambio en forma fácil, completa y consistentemente. La redundancia de un requerimiento puede causar un error al actualizar la especificación del requerimiento.

- Rastreable: Verifica que cada requerimiento haga referencia explícita a los documentos fuente y que tenga un nombre único o número de referencia.

4.6 Verifica el análisis de interfaces

<Analista o Supervisor de ACS> verifica que todas las interfaces externas e interfaces internas dentro de las funciones del software se encuentren completas, correctas y exactamente especificadas en la Especificación de Requerimientos. Asegúrate que se encuentren interfaces; tales como: repositorio de datos, protocolos de transferencia de datos, compatibilidad de interfaces, entre otros.

4.7 Verifica como fue preparado el Requerimiento

<Analista o Supervisor de ACS> verifica que la Especificación del requerimiento halla sido preparada por el *<Desarrollador>* y el *<Usuario>*. Debido a que usualmente el <Desarrollador> desconoce los problemas del *<Usuario>*, y esté desconoce del proceso de desarrollo de software.

4.8 Verifica la evolución del Requerimiento

<Analista o Supervisor de ACS> verifica si la especificación del requerimiento ha sido actualizada sin quedar incompleto. Identifica si existe un proceso de cambio, el cual permita una auditoría exacta y completa del cambio.

4.9 Verifica el uso de Prototipos

<Analista o Supervisor de ACS> verifica si el uso de prototipos ha servido para observar anticipadamente aspectos importantes del sistema y tener una menor cantidad de cambios durante el desarrollo.

4.10 Verifica los Diagramas de Flujo de Datos

<Analista o Supervisor de ACS> verifica si los procesos son consistentes, que no tengan la característica de un almacén de datos, que no generen datos únicamente.

<Analista o Supervisor de ACS> verifica la existencia de almacenes de datos que solo sirven para grabación y no de lectura.

<Analista o Supervisor de ACS> verifica la existencia de flujos de datos y sin nombre y/o numeración. Verifica que los diagramas hayan sido explicados en hojas de papel de dimensiones manejables y se puedan identificar un máximo de diez procesos.

<Analista o Supervisor de ACS> verifica la existencia de interfaces complejas y que los procesos estén particionados hasta que puedan ser especificados completamente y consistentemente.

<Analista o Supervisor de ACS> verifica que los diagramas sean entendibles.

4.11 Verifica Diccionario de Datos

<Analista o Supervisor de ACS> verifica que no existan elementos o entidades redundantes o con sinónimos; ya que no podrían facilitar su actualización.

4.12 Verifica el Diagrama de Proceso

<Analista o Supervisor de ACS> verifica que los procesos estén correctamente balanceados, lo que significa que tenga sus entradas y salidas correspondientes, indicando las interfaces necesarias.

<Analista o Supervisor de ACS> verifica el uso de símbolos o gráficos inválidos, que no correspondan a una entrada o salida respectivamente.

<Analista o Supervisor de ACS> verifica el uso de descriptivas, las cuales deben proporcionar con claridad (sin ambigüedad) una descripción de cómo se realiza el proceso.

4.13 Verifica si el Diseño esta incluido en el Análisis de Requerimiento

<Analista o Supervisor de ACS> verifica que la especificación del requerimiento no incluya: módulos, funciones, descripción del flujo de información, control modular, estructura de datos.

4.14 Verifica si los Requerimientos del Negocio están incluidos en la Especificación del Requerimiento

<Analista o Supervisor de ACS> verifica que la especificación del requerimiento no incluya: costos, calendario de entregas, reporte de procedimiento, métodos de desarrollo, criterios de validación y verificación, procedimiento de aceptación.

<Analista o Supervisor de ACS> determinar y verifica que lo especificado satisfaga completamente a los requerimientos del negocio y que todas las partes del software han sido especificadas.

4.15 Determina los Criterios de Validación

<Analista o Supervisor de ACS> determina los criterios de validación para la fase de aceptación y pruebas funcionales.

4.16 Evalúa y mide los riesgos

<Analista o Supervisor de ACS> evalúa el requerimiento aplicando una lista de comprobación basada en el conocimiento y la experiencia.

<Analista o Supervisor de ACS> Mide la magnitud de los riesgos detectados en los requerimientos. Efectúa una evaluación de los resultados obtenidos; luego, desarrolla un análisis de los riesgos detectados en los requerimientos.

<Analista o Supervisor de ACS> Indica las recomendaciones necesarias para eliminar los riesgos en los requerimientos.

4.17 Registra los resultados obtenidos

<Analista o Supervisor de ACS> informa al *<Jefe de Proyecto>* los riesgos y recomendaciones a incorporar en los requerimientos.

<Jefe de Proyecto> lee e incorpora las recomendaciones para mejorar la especificación de los requerimientos.

<Analista o Supervisor de ACS> controla que las recomendaciones hayan sido incorporadas.

4.18 Informa sobre la revisión

<Analista o Supervisor de ACS> registra los resultados obtenidos en *<formatos>* o *<archivos>* los cuales almacena en *<almacén>*. En caso de tener un mecanismo automatizado registrar los datos e información pertinente.

5. Documentación relacionada

<Entradas>

1. Requerimientos del Negocio
2. Plan del Proyecto.
3. Estándares.
4. Cronograma de Actividades.
5. Especificación de Requerimientos.
6. Requerimientos Funcionales y no Funcionales, Interfaces.
7. Análisis de riesgos.
8. Documentación del Usuario.
9. Plan de Pruebas.

<Entregables>

1. Análisis y evaluación de riesgos.
2. Informe de la revisión.
3. Reporte de Anomalías o Discrepancias.
4. Plan de pruebas Actualizado.

6. Lista de Comprobación

Responda a cada una de las preguntas y anote el puntaje correspondiente de acuerdo a lo observado y verificado. Luego, obtenga el Puntaje Total sumando los puntajes asignados, dividir el Puntaje Total entre el número de preguntas,

redondee el valor obtenido. Este valor compárelo con la tabla de valores del puntaje y conocerá la calificación de la fase revisada o inspeccionada.

Valores aplicados al Puntaje:

1= No tiene, no existe 2= Malo 3= Regular

4= Bueno 5= Excelente, Si tiene

Actividad (Preguntas)	**Puntaje** (1-5)
1. ¿La Especificación de Requerimientos describe los alcances del sistema? 2. ¿Están definidos los ambientes de Software o condiciones impuestas al software? 3. ¿Están definidos y documentados los Requerimientos Funcionales? 4. ¿Están definidas las funciones del software, que es lo que el software hace y como el software deberá responder a su ambiente? 5. ¿Están definidas las restricciones del software? ¿Estas son realistas? 6. ¿Están definidos los aspectos de rendimiento, fiabilidad, facilidad de uso entre otros? 7. ¿Se ha establecido el número de requerimientos del proyecto? 8. ¿Se ha establecido el número de requerimientos susceptibles a cambios? 9. ¿Se ha definido el contenido y estructura de la información de forma adecuada? 10. ¿Están definidas adecuadamente las interfaces importantes (internas y externas) del sistema? 11. ¿Los diagramas elaborados son claros? ¿Pueden utilizarse sin texto complementario? 12. ¿Se ha realizado un Prototipo para el usuario? 13. ¿El Cliente o Usuario ha revisado el prototipo? 14. ¿Existen inconsistencias, omisiones o redundancias? 15. ¿Se ha especificado los Requerimientos de Rendimiento? 16. ¿Se han especificado las Restricciones de Diseño? 17. ¿Se han especificado las pruebas para los requerimientos? 18. ¿Se han establecido con detalle los criterios de validación? 19. ¿Están completos los criterios de validación? 20. ¿Se han establecido los beneficios que se obtendrán con la solución? 21. ¿Se ha estado en permanente contacto con el Usuario o Cliente?	
Documentación 1. ¿La documentación de la Especificación del Requerimiento es entendible?	

2. ¿La documentación de la Especificación del Requerimiento es fácil de usar? 3. ¿La documentación de la Especificación del Requerimiento se puede mantener o actualizar fácilmente? 4. ¿El tiempo para entender la documentación es elevado?	

7. Análisis de Medición

Aplica las siguientes métricas, de acuerdo a las indicaciones dadas en el Anexo B:

- Indice de Defectos.
- Densidad de Defectos
- Estadística de Defectos por Fase.
- Distribución del Error
- Horas Hombre por principales Defectos detectados.
- Cumplimiento del Requerimiento.
- Métrica de la Calidad de la Especificación.
- Número de Requerimientos en Conflicto

8. Informe de Revisión

INFORME DE REVISION
<Destinatario>
<Fecha>

CONTENIDO

1. TERMINOS DE REFERENCIA
< > Información general sobre los requerimientos del negocio, identificación de los requerimientos funcionales y no funcionales.
2. PROCEDIMIENTO APLICADO
< > Procedimiento aplicado al efectuar la revisión
3. RESULTADOS OBTENIDOS
< > Observaciones encontradas en la revisión de los requerimientos de acuerdo a la lista de comprobación, riesgos en los que se puede incurrir y defectos encontrados.
4. RECOMENDACIONES
< > Recomendaciones para el cumplimiento de los requerimientos del Negocio, Clientes o Usuario.
5. PERSONAL PARTICIPANTE
< > Personal que ha participado para llevar a cabo la revisión.
6. RECURSOS EMPLEADOS
< > Registra los recursos humanos (horas/hombre), materiales, servicios y gastos empleados.

9. Formatos

<Especificación de Requerimientos> Es una plantilla que servirá para verificar el contenido del documento entregado por el *<Equipo de Desarrollo>*. Se mencionan los puntos que deberá tener dicho documento para verificar que este se encuentre completo.

Q Z **Especificación de Requerimientos** Num: _Pag.:1/n

1. Introducción
   1.1 Propósito
   1.2 Alcance
   1.3 Definiciones, abreviaturas
   1.4 Referencias
2. Descripción
   2.1 Visión General
   2.2 Funciones del Software
   2.3 Características del Uso
   2.4 Consideraciones
3. Especificaciones
   3.1 Requerimiento Funcional
      3.1.1 Flujo de Información
         3.1.1.1 Diagrama de Flujo de Datos 1
            3.1.1.1.1 Entidades
            3.1.1.1.2 Procesos
            3.1.1.1.3 Topología
      .
      .
      3.1.2 Descripción del Proceso
         3.1.2.1 Proceso 1
            3.1.2.1.1 Datos de entrada
            3.1.2.1.2 Algoritmos o formulas
            3.1.2.1.3 Entidades afectadas
            3.1.2.1.4 Servicios / Métodos / Herencias
      .
      .
      3.1.3 Diccionario de Datos
         3.1.3.1 Elemento de Dato 1
            3.1.3.1.1 Nombre / Descripción
            3.1.3.1.2 Unidad / Formato / Tipo de Dato
            3.1.3.1.3 Precisión / Exactitud / Rango
      .
      .
   3.2 Requerimientos de la Interfaces externas
      3.2.1 Interfaces del Usuario
      3.2.2 Interfaces de Hardware
      3.2.3 Interfaces de Software
      3.2.4 Interfaces de Comunicación
   3.3 Atributos del Software (Performance, Calidad, entre otros)
   3.4 Consideraciones para el diseño
   3.5 Criterios de Aceptación

4.6 Revisión del Diseño

Las actividades a realizar son las siguientes:

- **Revisión del Diseño Lógico**

  Un mal diseño lógico del sistema causará graves problemas en la construcción del software y por consiguiente perdidas de tiempo, recursos y económicas.

- **Revisión del Diseño Físico**

  Un mal diseño físico del sistema traería como consecuencia graves problemas en la calidad del software, dificultando las operaciones diarias que se realicen.

Procedimiento
4.6.1 Revisión del Diseño Lógico

El Diseño Lógico deberá ser revisado para determinar si las relaciones lógicas entre entidades y las relaciones entre procesos cumplen con la especificación de requerimientos, evitando la posibilidad de realizar cambios que vayan en desmedro de la integridad del diseño.
En este formato se deberá registrar información de las personas que aprobaron el procedimiento, cambios que se han efectuado y los departamentos en los cuales debe ser aplicado.

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Cambios

Revisión	Fecha	Persona Responsable	Descripción del Cambio
A		Nombre	Revisión Inicial

Lista de Distribución

(lista de los departamentos que reciben copia del procedimiento)

1. Propósito

- Verificar conceptualmente cómo trabajará el sistema, indiferente de la implementación física en una plataforma de hardware y sistema operativo.
- Definir los controles aplicados a las actividades de diseño para controlar el diseño del nuevo software y sus elementos.
- Asegurar y verificar que los requerimientos establecidos en la Especificación de Requerimientos hayan sido considerados en todos los niveles del Diseño.
- Asegurar y verificar que no existen partes del diseño generadas sin su respectivo requerimiento.
- Analizar y verificar que el Diseño este correcto, completo, exacto y consecuentemente satisfecho.
- Verificar que la documentación del diseño este conforme con los estándares aplicados, prácticas y convenciones.
- Controlar y verificar las responsabilidades de los diseñadores para que el software no sea afectado adversamente en su calidad, seguridad, y eficacia.

2. Alcance

Este procedimiento es aplicado para el desarrollo de nuevo software o modificaciones al mismo.

3. Responsabilidades

<Area o Dpto. Usuario> Provee requerimientos iniciales del software, valores base de performance. Aprueba el diseño final y preliminar.

<Equipo de Proyecto> Desarrolla los diagramas del diseño.

<Jefe de Proyecto> Aprueba los diagramas del diseño. Asegura la disponibilidad de software, hardware y servicios requeridos para el desarrollo y prueba del software.

<Analista o Supervisor de ACS> es responsable por la comprobación del nuevo diseño.

<Analista o Supervisor de ACS> Verifica si todas las autorizaciones de cambios en el diseño se han completado y registrado. Revisa las pruebas necesarias y requerimientos de la inspección.

<Analista o Supervisor ACS> Revisa los niveles de inspección y cambios a los diagramas, gráficos, esquemas, la colección y registro de datos de prueba, y cualquier modificación a la especificación del software.

4. Procedimiento

4.1 General

Ver Contrato.

<Analista o Supervisor de ACS> Revisa las especificaciones y asegura si es producible, verificable y controlable.

4.2 Verifica el Método de Diseño

<Analista o Supervisor de ACS> identifica el método ha utilizar en el diseño del software, pudiendo ser utilizado de acuerdo al tipo de proyecto o naturaleza del requerimiento. Entre estos métodos mencionaremos a los siguientes: Orientado a la Función, Orientado a los Datos, Orientado al Control de Tiempo Real, Orientado a Objetos, Orientado a un Lenguaje Formal.

4.3 Los Requerimientos integrados al Diseño

Vea Especificación de Requerimientos.

<Analista o Supervisor de ACS> verifica que todos los requerimientos del *<Usuario o Cliente>* hayan sido incorporados en el diseño. El diseño del software incluye la aplicación de estatutos y requisitos regulatorios inherentes al negocio.

<Analista o Supervisor de ACS> verifica que los requerimientos designados a más de un elemento de software diseñado estén completamente y consistentemente satisfechos.

4.4 Evalúa las características del Diseño

<Analista o Supervisor de ACS> verifica las restricciones del diseño y requisitos de portabilidad para el empaquetamiento del software.

4.5 Verifica las características de las Entidades

<Analista o Supervisor de ACS> revisa si todas las entidades y sus atributos se encuentran especificados en el Diseño. Identifica si existen entidades con el mismo nombre.

<Analista o Supervisor de ACS> verifica la naturaleza de las entidades ya que pueden existir algunas entidades no válidas.

<Analista o Supervisor de ACS> verifica si ha identificado para cada entidad los recursos externos necesarios; tales como: dispositivos físicos, librerías de servicio y recursos de procesamiento.

4.5 Verifica el Diagrama de Flujo de Datos

<Analista o Supervisor de ACS> verifica si los procesos son consistentes, que no tengan la característica de un almacén de datos, que no generen datos únicamente.

<Analista o Supervisor de ACS> verifica la existencia de almacenes de datos que solo sirven para grabación y no de lectura.

<Analista o Supervisor de ACS> verifica la existencia de flujos de datos y sin nombre y/o numeración. Verifica que los diagramas hayan sido explicados en hojas de papel de dimensiones manejables y se puedan identificar un máximo de diez procesos.

<Analista o Supervisor de ACS> verifica la existencia de interfaces complejas y que los procesos estén particionados hasta que puedan ser especificados completamente y consistentemente.

<Analista o Supervisor de ACS> verifica que los diagramas sean entendibles.

4.6 Verifica el Diagrama de Transición de Estados

<Analista o Supervisor de ACS> verifica que el diseño tenga un bajo nivel de acoplamiento.

<Analista o Supervisor de ACS> verifica que los diagramas sean entendibles.

4.7 Verifica el Diagrama de Estructuras

<Analista o Supervisor de ACS> verifica las estructuras con un alto grado de salidas de control. Asimismo, verifica que el efecto de los controles en el módulo padre estén dados únicamente por los módulos hijos o subordinados.

<Analista o Supervisor de ACS> verifica los módulos restrictivos y el uso de módulos del tipo caja negra.

<Analista o Supervisor de ACS> verifica que el diseño tenga un alto nivel de cohesión.

<Analista o Supervisor de ACS> verifica que los diagramas sean entendibles.

4.8 Verifica el Diagrama de Entidad - Relación

<Analista o Supervisor de ACS> verifica que todas las entidades se encuentren relacionada. Debe identificarse aquellas entidades que no tienen ninguna relación o en el que no se indica el tipo de relación (uno - uno, uno - muchos, muchos - muchos).

<Analista o Supervisor de ACS> identifica las relaciones erradas entre entidades que no van de acuerdo a la operatividad funcional del sistema.

<Analista o Supervisor de ACS> verifica que los diagramas sean entendibles.

4.9 Verifica el Diseño de Vistas

<Analista o Supervisor de ACS> verifica si el diseño de la vista es el adecuado y contempla la información y las funciones necesarias.

<Analista o Supervisor de ACS> verifica si se ha indicado los rangos de los valores de entrada, salidas, tipos de formato, manejo de errores, colores, entre otros.

4.10 Verifica el Diseño de Interfaces

<Analista o Supervisor ACS> evalúa las interfaces de los módulos para reducir la complejidad y la redundancia y mejorar la consistencia.

<Analista o Supervisor ACS> revisa si se ha identificado las entidades necesarias para las interfaces.

4.11 Verifica la Documentación del Diseño

<Analista o Supervisor de ACS> revisa que la documentación del diseño contemple las siguientes partes: Descripción General, Descripción de la Descomposición, Descripción de las Dependencias, Descripción de Interfaces y Detalle del Diseño por Módulo y Entidad.

4.12 Verifica los resultados de la revisión

<Analista o Supervisor de ACS> revisa los resultados y los diagramas del diseño.

<Analista o Supervisor de ACS> registra los resultados en *<formato>*. El *<formato>* se mantiene en el *<almacén>* por *<tiempo>*.

4.13 Revisión Final

La revisión final determina si el diseño elaborado por el *<Equipo de Proyecto>* contempla la especificación de requerimientos.

Si se Sub-contrato, *<Equipo de Prueba>* especializado calificado para efectuar pruebas especializadas.

El *<formato>* registra las versiones del diseño a desarrollar. *<Gerente de ACS>* aprueba el *<formato>*. El *<formato>* se mantiene por el *<Equipo de ACS>* por *<tiempo>*.

4.14 Reevaluar el Diseño

Periódicamente el producto es reevaluado para asegurar que el diseño todavía es válido con respecto a todos los requerimientos especificados. La revisión incluye: *<criterios>*.

Se mantiene un archivo de la calificación y mantenimiento en el *<formato>*. *<Gerente de ACS>* aprueba el *<formato>*. El *<formato>* se mantiene por *<Equipo de Proyecto>* por *<tiempo>*.

4.15 Revisión de Control

Los cambios en el Diseño se revisan de acuerdo con el diseño inicial. *<Analista de ACS>* revisa los cambios y modificaciones al diseño del software. *<Analista de ACS>* revisa la documentación.

4.16 Determina los Criterios de Validación

<Analista o Supervisor de ACS> determina los criterios de validación para la fase de pruebas unitarias y de integración.

5. Documentación relacionada

<Entradas>

1. Plan del Proyecto
2. Análisis de la estimación de esfuerzos, recursos.
3. Análisis de Riesgos.
4. Cronograma de Actividades.
5. Especificación del Requerimiento Funcionales y no Funcionales.
6. Especificación de Interfaces.
7. Documentación del Diseño.
8. Documentación del Usuario.
9. Plan de Revisión de ACS.
10. Lista de Comprobación

<Entregables>

1. Análisis y evaluación de riesgos.
2. Informe de la revisión.
3. Reporte de anomalías o discrepancias.
4. Plan de Revisión de ACS (Actualizado).

6. Lista de Comprobación

Responda a cada una de las preguntas y anote el puntaje correspondiente de acuerdo a lo observado y verificado. Luego, obtenga el Puntaje Total sumando los puntajes asignados, dividir el Puntaje Total entre el número de preguntas, redondee el valor obtenido. Este valor compárelo con la tabla de valores del puntaje y conocerá la calificación de la fase revisada o inspeccionada.

Valores aplicados al Puntaje:

1= No tiene, no existe 2= Malo 3= Regular

4= Bueno 5= Excelente, Si tiene

Actividades (Preguntas)	Puntaje (1-5)
1. ¿Están reflejados los requerimientos del software en su arquitectura? 2. ¿Están identificados los flujos de proceso del Cliente? 3. ¿Se han revisado los procedimientos operativos del Cliente? 4. ¿Están definidas adecuadamente las interfaces internas y externas? 5. ¿Los recursos del sistema están definidos para ser utilizados apropiadamente en la transferencia de datos? 6. ¿Están definidos el Modelo de Datos (Atributos, Diagrama Entidad Relación)? 7. ¿Es consistente la estructura de datos con los requerimientos del software? 8. ¿Están definidos el Diagrama de Flujo de Datos (Contexto y Nivel 1), 9. ¿Están definidos el Diagrama de Descomposición Funcional? 10. ¿Se ha conseguido una modularidad efectiva? 11. ¿Es adecuado el nivel de detalle del Diseño? 12. ¿Se está elaborando el Diccionario de Datos? 13. ¿Se han especificado las Restricciones de Diseño? 14. ¿Se ha especificado el manejo de seguridad por el software?	
Documentación 1. ¿La documentación de la Especificación del Diseño es entendible? 2. ¿La documentación de la Especificación del Diseño es fácil de usar? 3. ¿La documentación de la Especificación del Diseño se puede mantener o actualizar fácilmente? 4. ¿El tiempo para entender la documentación es elevado?	

7. Análisis de Medición

Aplica las siguientes métricas, de acuerdo a las indicaciones dadas en el Anexo B:

- Indice de Defectos
- Densidad de Defectos
- Estadística de Defectos por Fase
- Distribución del Error

- Horas Hombre por principales Defectos detectados
- Estructura del Diseño
- Integridad

8. Informe de Revisión

INFORME DE REVISION
<Destinatario>
<Fecha>

CONTENIDO

1. TERMINOS DE REFERENCIA
< > Información general sobre el método de diseño utilizado, interfaces necesarias, esquema general del diseño del software.
2. PROCEDIMIENTO APLICADO
< > Procedimientos aplicados al efectuar la revisión
3. RESULTADOS OBTENIDOS
< > Observaciones encontradas en la revisión del diseño lógico de acuerdo a la lista de comprobación u otros procedimientos aplicados, riesgos en los que se puede incurrir y defectos encontrados.
4. RECOMENDACIONES
< > Recomendaciones para el cumplimiento de los requerimientos del Negocio, Cliente o Usuario.
5. PERSONAL PARTICIPANTE
< > Personal que ha participado para llevar a cabo la revisión.
6. RECURSOS EMPLEADOS
< > Registra los recursos humanos (horas/hombre), materiales, servicios y gastos empleados.

9. Formatos

<Especificación del Diseño> Es una plantilla que servirá para verificar el contenido del documento entregado por el *<Equipo de Desarrollo>*. Se mencionan los puntos que deberá tener dicho documento para verificar que este se encuentre completo.

QZ **Especificación del Diseño** Num: ____Pag.:1/n

1. Introducción
   - 1.1 Propósito
   - 1.2 Alcance
   - 1.3 Definiciones, abreviaturas
   - 1.4 Referencias
2. Descripción
   - 2.1 Visión General
   - 2.3 Características Especiales
   - 2.4 Consideraciones
3. Especificaciones
   - 3.1 Módulos
     - 3.1.1 Modulo 1
       - 3.1.1.1 Propósito
       - 3.1.1.2 Descomposición Modular
       - 3.1.1.3 Dependencias
       - 3.1.1.4 Interfaces
       - 3.1.1.5 Descripción detallada del Diseño

     .
     .
   - 3.2 Procesos
     - 3.2.1 Proceso 1
       - 3.2.1.1 Propósito
       - 3.2.1.2 Descomposición del Proceso
       - 3.2.1.3 Dependencias
       - 3.2.1.4 Interfaces
       - 3.2.1.5 Descripción detallada del Proceso

     .
     .
   - 3.3 Entidades
     - 3.3.1 Entidad 1
       - 3.3.1.1 Propósito
       - 3.3.1.2 Descomposición de Datos
       - 3.3.1.3 Dependencias
       - 3.3.1.4 Descripción detalla de la Entidad

     .
     .
   - 3.4 Atributos del Diseño (Performance, Calidad, entre otros)
   - 3.5 Consideraciones para la Codificación

QZ

Procedimiento
4.6.2 Revisión del Diseño Físico

Es importante verificar la correcta especificación de los requerimientos de hardware y software base (sistema operativo, lenguaje de programación, etc.) para evitar costos innecesarios y contratiempos que perjudiquen las actividades del proyecto.
En este formato se deberá registrar información de las personas que aprobaron el procedimiento, cambios que se han efectuado y los departamentos en los cuales debe ser aplicado.

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Cambios

Revisión	Fecha	Persona Responsable	Descripción del Cambio
A		Nombre	Revisión Inicial

Lista de Distribución

(lista de los departamentos que reciben copia del procedimiento)

1. Propósito

- Verificar como trabajará el sistema de acuerdo a sus requerimientos físicos, según una plataforma de hardware, sistema operativo, administrador de base de datos, elementos de comunicación.

- Controlar y verificar que el hardware, sistema operativo, interfaces técnicas y otros elementos necesarios para la puesta en marcha del software, no afecten adversamente en su calidad, eficacia, eficiencia, performance y seguridad.

- Asegurar y verificar que los requerimientos establecidos en la Especificación de Requerimientos se consideren en el Diseño.

2. Alcance

Este procedimiento es aplicado para el desarrollo de nuevo software o modificaciones al mismo.

3. Responsabilidades

<Area o Dpto. Usuario> Provee requerimientos iniciales del software, valores base de performance y su utilidad en el mercado. Aprueba los diseños final y preliminar.

<Equipo de Proyecto> Desarrolla diagramas preliminares. *<Jefe de Proyecto>* aprueba los diagramas preliminares.

<Jefe de Proyecto> Asegura la disponibilidad de software, hardware y servicios requeridos para el desarrollo y prueba del software.

<Analista o Jefe de Proyecto> Traduce las necesidades del cliente en especificaciones técnicas de hardware y/o software (sistema operativo, administrador de base de datos, utilitarios y otros).

<Analista o Supervisor de ACS> Verifica si todas las autorizaciones de cambios en el diseño se han completado y registrado. Revisa las pruebas necesarias y requerimientos de la inspección. Revisa los niveles de inspección y cambios a los diagramas preliminares, gráficos, esquemas, la colección y registro de datos de prueba y cualquier modificación a la especificación del software.

4. Procedimiento

4.1 General

Ver Contrato.

Ver Especificación de Requerimientos.

<Analista o Supervisor de ACS> Evalúa el uso de un software que permita la evaluación de la configuración física del sistema por medio de simulaciones (tal como: SCERT) para determinar los volúmenes de archivos, tablas; tiempos de ejecución para actualizaciones, cálculos, operaciones con matrices.

4.2 Evalúa la Configuración

<Analista o Supervisor de ACS> revisa las especificaciones y asegura si es producible, verificable y controlable el ambiente del software.

<Analista o Supervisor de ACS> evalúa y verifica que el hardware corresponda a las necesidades del sistema a desarrollar, teniendo especial énfasis en características especiales necesarias para el sistema; tales como: longitud de la palabra, tamaño del número real, velocidad del procesador, tamaño del almacenamiento, sistema operativo, entre otros.

<Analista o Supervisor de ACS> inspecciona que el lenguaje seleccionado para el desarrollo del software sea el adecuado.

<Analista o Supervisor de ACS> evalúa e inspecciona las capacidades de memoria a ser utilizadas por el sistema, su tiempo de acceso.

<Analista o Supervisor de ACS> evalúa e inspecciona los métodos de entrada y salida; tales como: teclados en línea, discos ópticos, impresoras, representaciones visuales y de voz.

<Analista o Supervisor de ACS> evalúa e inspecciona la comunicación de datos para satisfacer la necesidad de información, estableciendo el tipo y número de canales controladores necesarios para una buena comunicación.

<Analista o Supervisor de ACS> evalúa e inspecciona el software accesorio necesario para el normal funcionamiento del sistema.

4.3 Diseño de Interfaces

<Analista o Supervisor de ACS> evalúa si cada una de las interfaces diseñadas son necesarias. Evalúa si han sido diseñadas para una administración efectiva de la configuración.

4.4 Diseño de Formatos

<Analista o Supervisor de ACS> evalúa y verifica si los formatos de salida diseñados son los adecuados, y contiene todos los datos y campos necesarios.

5. Documentación relacionada

<Entradas>

1. Especificación del Requerimiento
2. Especificación de Interfaces
3. Estimación del Esfuerzo
4. Plan de Revisión de ACS
5. Lista de Comprobación

<Entregables>

1. Análisis y evaluación de riesgos
2. Informe de la revisión
3. Reporte de Anomalías
4. Plan de Revisión de ACS (Actualizado)

6. Lista de Comprobación

Responda a cada una de las preguntas y anote el puntaje correspondiente de acuerdo a lo observado y verificado. Luego, obtenga el Puntaje Total sumando los puntajes asignados, dividir el Puntaje Total entre el número de preguntas, redondee el valor obtenido. Este valor compárelo con la tabla de valores del puntaje y conocerá la calificación de la fase revisada o inspeccionada.

Valores aplicados al Puntaje:

1= No tiene, no existe 2= Malo 3= Regular

4= Bueno 5= Excelente, Si tiene

Actividades (Preguntas)	**Puntaje** (1-5)
1. ¿ Están identificados los flujos de proceso del Cliente? 2. ¿ Se han revisado los procedimientos operativos del Cliente? 3. ¿ Están definidas adecuadamente las interfaces internas y externas? 4. ¿Los requerimientos de performance y límites de las interfaces están bien definidos? 5. ¿Los recursos del sistema están definidos para ser utilizados apropiadamente en la transferencia de datos? 6. ¿Las interfaces utilizan estándares apropiados? 7. ¿ Se han especificado las Restricciones de Diseño?	
Documentación 1. ¿La documentación de la Especificación del Diseño es entendible? 2. ¿La documentación de la Especificación del Diseño es fácil de usar? 3. ¿La documentación de la Especificación del Diseño se puede mantener o actualizar fácilmente? 4. ¿El tiempo para entender la documentación es elevado?	

7. Análisis de Medición

Aplica las siguientes métricas, de acuerdo a las indicaciones dadas en el Anexo B:

- Indice de Defectos
- Densidad de Defectos
- Estadística de Errores por Fase
- Horas Hombre por principales Defectos detectados

8. Informe de Revisión

INFORME DE REVISION
<Destinatario>
<Fecha>

CONTENIDO

1. TERMINOS DE REFERENCIA
< > Información general sobre el diseño, equipos, dispositivos o elementos necesarios para la correcta instalación del software.
2. PROCEDIMIENTO APLICADO
< > Procedimientos aplicados al efectuar la revisión
3. RESULTADOS OBTENIDOS
< > Observaciones encontradas en la revisión del diseño físico de acuerdo a la lista de comprobación u otros procedimientos aplicados, riesgos en los que se puede incurrir y defectos encontrados.
4. RECOMENDACIONES
< > Recomendaciones para el cumplimiento de los requerimientos del Negocio, Cliente o Usuario. Así como, compra de equipos o dispositivos necesarios para la correcta implementación del software.
5. PERSONAL PARTICIPANTE
< > Personal que ha participado para llevar a cabo la revisión.
6. RECURSOS EMPLEADOS
< > Registra los recursos humanos (horas/hombre), materiales, servicios y gastos empleados. Los objetivos de esta fase son traducir entidades del plan en una descripción física unitaria del sistema.

Procedimiento
4.7 Inspección de la Construcción

Es necesario verificar la correcta implementación de especificación del diseño; además, se hace necesario el uso de herramientas automáticas que permitan un análisis estático debido al tiempo que implica realizar este proceso. Esto permitirá encontrar errores y poder obtener un código limpio, flexible y eficiente.
En este formato se deberá registrar información de las personas que aprobaron el procedimiento, cambios que se han efectuado y los departamentos en los cuales debe ser aplicado.

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Cambios

Revisión	Fecha	Persona Responsable	Descripción del Cambio
A		Nombre	Revisión Inicial

Lista de Distribución

(lista de los departamentos que reciben copia del procedimiento)

1. Propósito

- Asegurar y verificar que las Especificaciones del Diseño han sido incorporados en los correspondientes códigos de los programas o elementos.
- Asegurar que todas las propiedades de calidad del Código Fuente son satisfechas.
- Detectar errores de lógica y control del flujo.
- Verificar el uso de estándares y procedimientos en la codificación de los programas o elementos.

2. Alcance

Este procedimiento se aplica al software desarrollado internamente o por terceros.

3. Responsabilidades

<Proveedor> Desarrolla el software de acuerdo a los requerimientos del usuario. Proporciona información relativa al contrato y su plan de actividades.

<Analista o Supervisor de ACS> Inspecciona el código del programa fuente identificando el uso de variables, estructuras y documentación.

<Area de Sistemas> Elabora los programas especificados en el Diseño del Software.

4. Procedimiento

4.1 General

<Analista o Supervisor de ACS> revisa el uso del lenguaje de programación que se utiliza en la organización. Además el requerimiento de hardware y software por parte del programa.

<Analista o Supervisor de ACS> analiza el uso de una herramienta automática para la Inspección de la codificación de los programas. Por Ejm: Herramienta de Análisis Estático como SQA - Suite.

<Analista o Supervisor de ACS> analiza las técnicas necesarias ha ser usadas en la evaluación del código, dependiendo del nivel de riesgo o criticidad deseado (por ejemplo una revisión, inspección, una herramienta, etc.).

4.2 Identifica los elementos del Software

<Analista o Supervisor de ACS> identifica los elementos del software y los datos. A continuación, se mencionan algunos de ellos:

Elementos	Datos
▪ Subsistemas ▪ Procedimientos ▪ Programas ▪ Rutinas ▪ Switches ▪ Triggers ▪ Store Procedure	▪ Archivos y tablas ▪ Registros lógicos ▪ Campos

<Analista de ACS> analiza y selecciona los elementos de software a evaluar.

4.3 Evalúa el Uso de Estándares

<Analista de ACS> verifica que el código fuente cumpla con los estándares establecidos, practicas y convenciones.

<Analista de ACS> asegura que los estándares de codificación sean entendibles y disponibles al Equipo de ACS antes de que se inicie la fase de codificación.

4.4 Verifica que el Código Fuente corresponda y cumpla con los Requerimientos del Diseño

<*Analista de ACS*> verifica que el código fuente corresponda con la especificación del diseño y la especificación del diseño corresponde con el código fuente. Identifica posibles inconsistencias entre el diseño y el código, identificando anomalías; tales como: código fuente sin componentes de diseño, componentes de diseño que no están claramente asociados con el código fuente o cualquier otra anormalidad que emerja del proceso.

<*Analista de ACS*> verifica que ninguna especificación del diseño sea omitida o codificada más de una vez; asimismo, identifica la existencia de código extraño.

<*Analista de ACS*> verifica que la especificación del diseño este contemplado en forma completa y consistente en cada unos de los elementos del código fuente.

4.5 Determina los atributos de Calidad del Código Fuente

<*Analista o Supervisor de ACS*> determina los atributos de calidad que el código fuente debe cumplir para obtener un código fuente eficiente. Estos atributos deben estar en función a los factores y sub- factores de calidad, los cuales deberán ser relacionados con cada uno de los elementos del Software.

4.6 Estructura del Programa (If, Loop, Do y otros)

<*Analista o Supervisor de ACS*> inspecciona que la codificación de los programas se halla efectuado de manera modular.

<*Analista o Supervisor de ACS*> utiliza una herramienta automática para el análisis de estructuras del programa, como por ejemplo: SQA SUITE, el cual podrá realizar un análisis estático completo del software.

4.7 Semántica del Programa

<Analista o Supervisor de ACS> verifica errores de sintaxis y semánticos que no necesariamente son identificados por los compiladores o interpretes.

4.8 Errores de lógica

<Analista o Supervisor de ACS> inspecciona la lógica del programa. Verifica la existencia de puntos de ruptura y cálculos en que exista un divisor entre cero.

4.9 Analiza la Estructura de Control

<Analista o Supervisor de ACS> inspecciona las condiciones lógicas contenidas en los programas, pudiendo estar en: operadores lógicos incorrectos y sobrantes, variables lógicas, paréntesis lógicos, operadores relacionales, expresiones aritméticas.

<Analista o Supervisor de ACS> inspecciona la construcción de los bucles y verifica que la existencia de bucles no estructurados.

4.10 Uso de variables y parámetros

<Analista o Supervisor de ACS> inspecciona el uso de nombre de variables o parámetros inapropiados, simbólicos o abreviados. Por ejm: Rosa, I, Veces y otros.

<Analista o Supervisor de ACS> Inspecciona que los nombre de las variables no tengan una longitud excesiva o innecesaria que confunda su uso.

<Analista o Supervisor de ACS> Inspecciona que el número de parámetros no sea excesivo, considerar como tope máximo un número de 6 parámetros.

<Analista o Supervisor de ACS> Verifica el uso de estándares para la creación y uso de variables y/o parámetros.

4.11 Análisis de Interfaces

<Analista o Supervisor de ACS> evalúa el código fuente con el hardware, operador, y documentación de diseño del software para la corrección, consistencia, completitud, exactitud y análisis de los datos manejados por cada interface.

<Analista o Supervisor de ACS> evalúa el código fuente y los elementos de software que están mal referenciados o definidos y aquellos que están referenciados pero no definidos.

4.12 Evalúa la documentación del Código Fuente

<Analista o Supervisor de ACS> inspecciona que los programas, rutinas, funciones o código elaborado tengan un comentario de la labor que realizan, fechas de actualización, autor, entre otros datos de importancia.

<Analista o Supervisor de ACS> revisa que los programas se encuentren documentados y estos se encuentren actualizados, ya que esto facilita el mantenimiento y los cambios al sistema. La documentación deberá contener:

- Características del Programa: debe indicar el objeto o finalidad del programa, software utilizado para su compilación, conexiones y componentes de hardware utilizados, tiempo de proceso estimado, fecha de creación, autor entre otros.
- Salidas: debe indicar los tipos de salida y archivos que se obtienen
- Entradas: debe indicar los tipos de entrada y archivos utilizados.

- Cambios del Programa: debe indicar los cambios realizados al programa desde su creación, incorporándosele fecha, autor del cambio, motivo y otros.

- Diseño del Programa: debe mostrar el diseño estructural del programa en forma gráfica y narrada. Debe indicar las partes modulares mas importantes, entradas, salidas, constantes, tablas y otros.

- Pseudo código: debe permitir entender la lógica del programa de manera entendible (lenguaje natural).

4.13 Registra los resultados obtenidos

<Analista o Supervisor de ACS> informa al *<Jefe de Proyecto>* los defectos que contiene el código y recomendaciones a incorporar en el código.

<Jefe de Proyecto> lee e incorpora las recomendaciones para mejorar el código.

<Analista o Supervisor de ACS> controla que las recomendaciones hayan sido incorporadas.

4.14 Informa sobre la Inspección

<Analista o Supervisor de ACS> registra los resultados obtenidos en *<formatos>* o *<archivos>* los cuales almacena en *<almacén>*. En caso de tener un mecanismo automatizado registrar los datos e información pertinente.

5. Documentación relacionada

<Entradas>

1. Especificación del Diseño Funcional
2. Especificación del Diseño Físico
3. Especificación de Interfaces

4. Especificación del código fuente
5. Especificación de algoritmos utilizados
6. Estimación del Esfuerzo
7. Modelo de Datos
8. Diagrama de Flujo de Datos
9. Plan de Pruebas
10. Plan de Revisión de ACS
11. Lista de Comprobación
12. Lista de la Configuración: librerías, programas, triggers.
13. Código Fuente

<Entregables>

1. Análisis y evaluación de riesgos
2. Informe de la revisión
3. Reporte de anomalías.
4. Plan de Revisión de ACS (Actualizado)

6. Lista de Comprobación

Responda a cada una de las preguntas y anote el puntaje correspondiente de acuerdo a lo observado y verificado. Luego, obtenga el Puntaje Total sumando los puntajes asignados, dividir el Puntaje Total entre el número de preguntas, redondee el valor obtenido. Este valor compárelo con la tabla de valores del puntaje y conocerá la calificación de la fase revisada o inspeccionada.

Valores aplicados al Puntaje:

1= No tiene, no existe 2= Malo 3= Regular
4= Bueno 5= Excelente, Si tiene

Actividades (Preguntas)	**Puntaje** (1-5)
1. ¿Se ha traducido adecuadamente el diseño al código? 2. ¿Existen los programas para cada especificación? 3. ¿Los programas se han desarrollado basados en los estándares existentes?	

4. ¿Han pasado por el área de pruebas? 5. ¿Se ha realizado análisis estático del código fuente? 6. ¿Se han medido el grado de complejidad del software? 7. ¿La estructura del programa es modular? 8. ¿Los programas hacen lo que las especificaciones dice deben hacer? 9. ¿Han avanzado esta fase según cronograma? 10. ¿Están definidas las reglas de programación? 11. ¿Están definidos los estándares de programación? 12. ¿Existen comentarios incorrectos o ambiguos? 13. ¿Se ha hecho uso adecuado de los estándares? 14. ¿Son correctas las constantes físicas?	
Documentación 1. ¿El programa esta adecuadamente documentado? 2. ¿La documentación del programa es fácil de entender? 3. ¿El tiempo para entender la documentación es elevado?	
Mantenimiento 1. ¿El programa facilita su mantenimiento? 2. ¿El programa tiene una estructura o lógica entendible? 3. ¿El tiempo de mantenimiento al programa es elevado?	

7. Análisis de Medición

Aplica las siguientes métricas, de acuerdo a las indicaciones dadas en el Anexo B:

- Indice de Defectos
- Densidad de Defectos
- Estadística de Errores por Fase
- Distribución del Error
- Horas Hombre por principales Defectos detectados
- Medida de la Complejidad de McCabe
- La Ciencia del Software de Halstead

8. Informe de Revisión

INFORME DE INSPECCION
<Destinatario>
<Fecha>

CONTENIDO

1. TERMINOS DE REFERENCIA
< > Información general sobre la codificación efectuada, lineamientos seguidos para conseguir que el código corresponda a las necesidades del diseño.
2. PROCEDIMIENTO APLICADO
< > Procedimientos aplicados al efectuar la inspección.
3. RESULTADOS OBTENIDOS
< > Observaciones encontradas en la inspección del código de acuerdo a la lista de comprobación u otros procedimientos aplicados y defectos encontrados.
4. RECOMENDACIONES
< > Recomendaciones para obtener un código más puro y tener la mínima cantidad de defectos.
5. PERSONAL PARTICIPANTE
< > Personal que ha participado para llevar a cabo la revisión.
6. RECURSOS EMPLEADOS
< > Registra los recursos humanos (horas/hombre), materiales, servicios y gastos empleados.

9. Formatos

<Especificación de la Codificación> Es una plantilla que servirá para verificar el contenido del documento entregado por el *<Equipo de Desarrollo>*. Se mencionan los puntos que deberá tener dicho documento para verificar que este se encuentre completo.

Q Z **Especificación del Código Fuente** Num:__Pag.:1/n

1. Introducción
   - 1.1 Propósito
   - 1.2 Alcance
   - 1.3 Definiciones, abreviaturas
   - 1.4 Referencias
2. Descripción
   - 2.1 Visión General
   - 2.2 Características Especiales
   - 2.3 Estándares
   - 2.4 Elementos del Software
   - 2.5 Consideraciones
3. Especificaciones
   - 3.1 Componente / Elemento / Programa
     - 3.1.1 Código Fuente 1
       - 3.1.1.1 Propósito
       - 3.1.1.2 Autor(es) / Fecha Creación / Modificaciones / Fecha de Cambio
       - 3.1.1.3 Herramientas utilizadas (Lenguaje, Compiladores, entre otros)
       - 3.1.1.4 Interfaces
       - 3.1.1.5 Archivos relacionados
       - 3.1.1.6 Reportes o Formatos relacionados
       - 3.1.1.6 Diagrama de Proceso
       - 3.1.1.7 Diagrama de Flujo / Pseudo Código
       - 3.1.1.8 Listado del Código Fuente

     .
     .
   - 3.2 Atributos del Código (Performance, Calidad, entre otros)
   - 3.3 Consideraciones para las Pruebas
   - 3.4 Criterios de Validación

Capitulo 5

PLAN DE PRUEBAS DEL SOFTWARE

5.1 Propósito

El propósito es definir un esquema general para definir, organizar, administrar y controlar las pruebas de Verificación y Validación de todo el Software desarrollado y obtener la aceptación del Cliente o Usuario respecto al producto final.

5.2 Alcance

El alcance de este Plan es verificar y validar todo el software desarrollado y todos los componentes involucrados para conformación de los requerimientos. El Plan se aplica a los productos de software en la fase de pruebas.

5.3 Actividades

- **Administración de Pruebas**

  Definir procedimientos operativos de administración de las pruebas para definir conflictos y solución de anomalías y definir políticas.

- **Casos de Prueba**

Es importante debido a que es una forma de verificar la validez de la funcionalidad de los módulos y el software.

- **Pruebas de Unitarias**

  Esta prueba comprobará la calidad del componente con respecto a la especificación del diseño.

- **Pruebas de Integración**

  Esta prueba comprobará la calidad de los componentes interactuando entre sí; con respecto a la especificación de los requerimientos.

- **Pruebas de Sistema**

  Esta prueba comprobará la calidad del software con respecto a los requerimientos funcionales del negocio.

- **Pruebas de Aceptación**

  Son los criterios o niveles de calidad a tener en cuenta para la aceptación del software.

Actividades	Requerimientos	Diseño	Construcción	Pruebas
Planificación de Pruebas	• Sistema • Aceptación	• Unitarias • Integración		
Generación de Diseño de Pruebas		• Unitarias • Integración • Sistema • Aceptación		
Generación de casos de Pruebas			• Unitarias • Integración • Sistema • Aceptación	
Generación de procedimientos de Prueba			• Unitarias • Integración • Sistema	• Aceptación
Ejecución de Pruebas				• Unitarias • Integración • Sistema • Aceptación

Procedimiento
5.4 Administración de Pruebas

Se debe preparar anticipadamente el plan de pruebas; de tal manera, que durante su ejecución se puedan haber previsto la gran variedad de casos de prueba posibles y los criterios de Evaluación pertinentes. Esto será soportado por un buen ambiente de pruebas, en el cual deberá ser previsto el uso de una herramienta de pruebas, librerías de soporte entre otros. Asimismo, elabora los procedimientos adecuados de como proceder con las anomalías y desviaciones encontradas, siendo esto último un punto de suma importancia.
En este formato se deberá registrar información de las personas que aprobaron el procedimiento, cambios que se han efectuado y los departamentos en los cuales debe ser aplicado.

Aprobado: _____
Cargo Nombre

Aprobado: _____
Cargo Nombre

Aprobado: _____
Cargo Nombre

Aprobado: _____
Cargo Nombre

Aprobado: _____
Cargo Nombre

Cambios

Revisión	Fecha	Persona Responsable	Descripción del Cambio
		Nombre	Revisión Inicial

Lista de Distribución

(lista de los departamentos que reciben copia del procedimiento)

1. Propósito

- Diseñar pruebas que verifiquen la ejecución apropiada del software
- Instalar y documentar el ambiente de prueba previo a que la solución y las pruebas de aceptación sean ejecutadas.
- Asegurar que el ambiente de prueba este listo para la ejecución de la prueba y que los equipos, programas, herramientas y el ambiente de soporte estén disponibles.
- Describe el método de resolución de anomalías y emisión de reportes.
- Determinar el criterio para la cual una tarea del Plan debe ser sometida cuando las entradas son cambiadas.
- Describe los procedimientos y formatos usados para la aprobación de desviaciones al Plan de Pruebas.
- Describir como los productos y resultados de Software deberán ser configurados, protegidos y almacenados.

2. Alcance

El procedimiento es utilizado por el <*Equipo de ACS*> para la Gestión del Plan de Pruebas.

3. Responsabilidades

<*Jefe Proyecto*> Proporciona las facilidades para que el <*Equipo de ACS*> realice las pruebas.

<*Gerente de ACS*> Encargado de Aprobar el Plan de Pruebas.

<*Supervisor de ACS*> Encargado de las coordinaciones con el <*Jefe de Proyecto*> en la planificación de las pruebas, reporta al <*Gerente de ACS*>.

<Analista de ACS> Encargado de definir los procedimientos de pruebas, casos de pruebas, ejecutaran las pruebas, registraran los resultados y reportara los resultados a *<Supervisor de ACS>*.

<Soporte Técnico> Proporcionara Apoyo operativo y logístico a los *<Analista de ACS>*.

4. Procedimiento

4.1 General

<Supervisor de ACS> verifica la adquisición de Herramientas de Prueba que faciliten la labor de validación y prueba. Verifica la experiencia del personal en el manejo de dichas herramientas, en caso de no haber experiencia incluir en el Plan de Entrenamiento la capacitación necesaria al *<Equipo de ACS>*.

4.2 Diseña las Pruebas de Evaluación

<Supervisor de ACS> diseña el Plan de Pruebas desde la fase de Análisis de Requerimientos, teniendo en cuenta todas las consideraciones necesarias para la aceptación del *<Cliente o Usuario>*.

<Supervisor de ACS> define los criterios de pruebas de acuerdo a cada fase de pruebas.

<Analista o Supervisor de ACS> diseña los procedimientos de ejecución de las pruebas y las métricas a utilizar.

<Supervisor de ACS> diseña los casos de pruebas necesarios de acuerdo al diseño de las pruebas, enfatizando la reutilización de materiales de prueba existentes, modificados y complementados como sea necesario. Donde sea posible, ajustar y reutilizar los casos de prueba, grupos y datos existentes.

<Supervisor de ACS> documenta las condiciones a ser probadas, la naturaleza de la prueba y los resultados esperados a ser demostrados. Selecciona los datos de prueba adecuada.

<Supervisor de ACS> selecciona las técnicas de pruebas apropiadas; así como, las herramientas apropiadas para soportar las técnicas seleccionadas, basadas en la tecnología utilizada y el ambiente de prueba.

<Supervisor de ACS> solicita reuniones de coordinación con *<Soporte Técnico>* para la comprobación de la instalación y operatividad de las métricas a usar.

<Supervisor de ACS> documenta los resultados de las pruebas.

4.3 Prepara el Ambiente de Pruebas

<Analista de ACS> establece una configuración única que se adapte a las pruebas funcionales que se planearon. Utiliza un ambiente en el que puedas simular las condiciones del ambiente de producción.

<Analista de ACS> instala las herramientas de prueba seleccionadas, verifica su operatividad y disposición al *<Equipo de ACS>*. Utiliza estas herramientas para generar datos de prueba.

<Analista de ACS> define las librerías de prueba necesarias para almacenar los archivos, datos y programas de prueba.

4.4 Reporte y Resolución de anomalías.

<Supervisor de ACS> determina el criterio para reportar una anomalía, determinar una lista de distribución de reportes de anomalías, la autoridad y las líneas de tiempo para resolver las anomalías.

<Supervisor de ACS> define los niveles de anomalías. Por cada anomalía detectada deberá a definir el procedimiento de resolución satisfactoria.

<Gerente de ACS> evalúa los efectos de costo, horario y los efectos en la calidad.

4.5 Política de Cambio de tareas

<Gerente de ACS> evalúa el costo del cambio de las entradas en las tareas del Plan y sus efectos en los costos, cronogramas y calidad.

4.6 Política de Desviaciones

<Gerente de ACS> delega al *<Supervisor de ACS>* la responsabilidad de la aprobación de desviaciones del Plan de Pruebas de ACS.

<Supervisor de ACS> identifica las tareas a realizarse en caso de desviación del Plan.

<Supervisor de ACS> determina las desviaciones razonables del Plan de Pruebas, informando cualquier anomalía al *<Gerente de ACS>*.

<Analista de ACS> determina el impacto en la calidad del Software de las nuevas tareas o pruebas a realizarse.

4.7 Registro y Seguridad

<Analista de ACS> determina las previsiones de seguridad necesarias para proteger la información de las pruebas de alteraciones accidentales o deliberadas.

<Analista de ACS> verifica los medios donde se almacenen y registren los resultados de las pruebas.

5. Documentación relacionada

<Entradas>

1. Cronograma de Actividades.
2. Plan del Proyecto
3. Especificación de los requerimientos funcionales y no funcionales
4. Especificación de las interfaces internas y externas.

<Entregables>

1. Plan de Pruebas.

6. Lista de Comprobación

Responda a cada una de las preguntas y anote el puntaje correspondiente de acuerdo a lo observado y verificado. Luego, obtenga el Puntaje Total sumando los puntajes asignados, dividir el Puntaje Total entre el número de preguntas, redondee el valor obtenido. Este valor compárelo con la tabla de valores del puntaje y conocerá la calificación de la fase revisada o inspeccionada.

Valores aplicados al Puntaje:

1= No tiene, no existe	2= Malo	3= Regular
4= Bueno	5= Excelente, Si tiene	

Actividades (Preguntas)	**Puntaje** (1-5)
Plan de Pruebas 1. ¿Existe un Plan de Pruebas definido? 2. ¿El Plan de Pruebas es consistente con el Plan del Proyecto? 3. ¿Existe un Plan para los componentes, pruebas de integración, prueba del sistema y aceptación de las pruebas? 4. ¿Existe un ambiente de Pruebas? 5. ¿Existe Herramientas de pruebas? ¿Esta correctamente implementada? 6. ¿Se han identificado los tipos de pruebas funcionales, pruebas limites, pruebas de performance, pruebas de uso, pruebas de resistencia? 7. ¿Se han identificado los Casos de Prueba, Datos de Prueba y Resultados esperados? 8. ¿Se conoce cuantos errores (de tipo concreto) se esperan cuando comiencen las pruebas? 9. ¿Se ha establecido el número de módulos propensos a error? 10. ¿Están definidos los criterios en que se juzgará la realización de las Pruebas?. 11. ¿Existe documentación de las pruebas de documentación del Usuario?. 12. ¿Quién es el responsable para generar los diferentes niveles de diseño de pruebas, casos y procedimientos? 13. ¿Quién es el responsable para ejecutar los diferentes niveles de pruebas? 14. ¿Quién es el responsable para construir y mantenimiento de las pruebas? 15. ¿Quién es el responsable para el manejo de configuraciones? 16. ¿Cuales son los criterios para detener el esfuerzo de prueba? 17. ¿Cuales son los criterios para reiniciar el esfuerzo de la prueba? 18. ¿Cuándo el código fuente será ubicado bajo control de cambios? 19. ¿Cuales diseño de pruebas, casos, y procedimientos serán ubicados bajo el control de cambios? 20. ¿Para que niveles de pruebas se escriben informes de anomalías? **21.** ¿Esta definida una metodología de implementación?	

Procedimiento 1. ¿Se han probado todos los caminos lógicos independientes? 2. ¿Se han identificado y listado los casos de prueba junto con los resultados? 3. ¿Se ha probado el manejo de errores? 4. ¿Se ha probado los valores límites? 5. ¿Se ha probado el rendimiento y limitación temporales? 6. ¿Se ha especificado la variación aceptable respecto a los resultados esperados?	
Facilidad de Prueba 1. ¿ El sistema tiene pocos errores y se hace fácil las pruebas? 2. ¿ Ningún error bloquea la ejecución de las pruebas? 3. ¿ Se genera una salida distinta para cada entrada? 4. ¿ Un resultado incorrecto se puede identificar fácilmente? 5. ¿ Todos los resultados posibles se pueden generar a través de alguna combinación de entrada? 6. ¿ Se puede controlar directamente los estados y las variables de hardware y software? 7. ¿ El código fuente es accesible? 8. ¿ Los formatos de entrada y resultados son consistentes y estructurados? 9. ¿ Las pruebas pueden especificarse, automatizarse y reproducirse convenientemente? 10. ¿ Los módulos del software se pueden probar independientemente? : 11. ¿ Existe simplicidad funcional, mínimas características que cumplir los requisitos? 12. ¿ Existe simplicidad estructural, arquitectura modularizada? 13. ¿ Existe simplicidad en el código, se adoptan estándares? 14. ¿ Los cambios del software son infrecuentes? 15. ¿ Los cambios del software están controlados? 16. ¿ Los cambios del software no invalidan las pruebas existentes? 17. ¿ El software se recupera de las fallas? 18. ¿ El diseño se ha entendido perfectamente? 19. ¿ Las dependencias entre los componentes internos, externos y compartidos se han entendido perfectamente? 20. ¿ La documentación técnica es instantáneamente accesible? **21.** ¿ La documentación técnica esta bien organizada, es especifica, detallada y exacta?	

Procedimiento
5.5 Casos de Prueba

Prepara anticipadamente todos casos de prueba posibles especificando los valores de entrada, resultados esperados y condiciones límites; de tal manera, que se pueda evaluar de manera completa todas las características del software.
En este formato se deberá registrar información de las personas que aprobaron el procedimiento, cambios que se han efectuado y los departamentos en los cuales debe ser aplicado.

Aprobado: _____
Cargo Nombre

Aprobado: _____
Cargo Nombre

Aprobado: _____
Cargo Nombre

Aprobado: _____
Cargo Nombre

Aprobado: _____
Cargo Nombre

Cambios

Revisión	Fecha	Persona Responsable	Descripción del Cambio
		Nombre	Revisión Inicial

Lista de Distribución

(lista de los departamentos que reciben copia del procedimiento)

1. Propósito

- Especificar el mínimo número de casos que proveen un adecuado nivel de confiabilidad en las pruebas, para cada nivel establecido.
- Diseñar pruebas que verifiquen la ejecución apropiada del software en su totalidad, su integración, interfaces con otros sistemas y operación en un ambiente semejante al de producción.
- Diseñar los componentes de la prueba que permitan probar el software sin integrarse con los otros componentes y sistemas externos, incluyendo la comunicación entre módulos.
- Diseñar las pruebas unitarias para los módulos nuevos y/o modificados.
- Especificar la funcionalidad de nuevas herramientas de pruebas que han de ser desarrolladas o adquiridas.

2. Alcance

Los casos de pruebas serán utilizados desde un ambiente de Pruebas para las Pruebas Unitarias, Pruebas de Integración, Pruebas de Sistemas y Pruebas de Aceptación.

3. Responsabilidades

<Cliente o Usuario> Solicita el cumplimiento de sus requerimientos.

<Supervisor de ACS> Encargado de la planificación de las pruebas Unitarias del Software. Coordina con el *<Jefe Proyecto>* en el calendario de pruebas.

4. Procedimiento

4.1 General

<Supervisor de ACS> verifica que cada uno de los casos de prueba especifique los valores de entrada y los resultados esperados.

4.2 Diseña casos de Pruebas Unitarias

<*Supervisor de ACS*> diseña los casos de prueba que verifiquen la funcionalidad de los módulos, validen todos los caminos lógicos, aseguren el cubrimiento de todas las condiciones (normales, excepciones, errores), y recomendar las técnicas y herramientas de pruebas.

<*Supervisor de ACS*> crea una lista de toda la funcionalidad contenida en las unidades a ser probadas.

<*Supervisor de ACS*> crea matrices que reflejen los caminos lógicos en el código. La prueba debe cubrir a cada instrucción, salto o decisión y cualquier condición (incluyendo múltiples condiciones).

<*Supervisor de ACS*> diseña los casos de prueba de unidad que provean la cobertura de toda la funcionalidad y caminos lógicos identificados, considerando todas las condiciones normales y no esperadas, de errores de entrada/salida.

4.3 Diseña casos de Pruebas de Integración

<*Supervisor de ACS*> diseña los casos de prueba que verifiquen la funcionalidad inter - módulos y las interfaces inter - módulos. Donde sea posible, utilizar pruebas de "Caja Negra" para Prueba de Integración de Módulos.

<*Supervisor de ACS*> determina la secuencia y alcance de la integración de los módulos del software.

4.4 Diseña casos de Prueba del Sistema

<*Supervisor de ACS*> diseña y documenta los casos de prueba requeridos para ejecutar exitosamente la prueba del sistema, asegurando la cobertura de todos los aspectos funcionales, de tiempo de ejecución, de ambiente y de operación del software.

<Supervisor de ACS> documenta los casos en suficiente detalle para permitir un desarrollo e implantación fáciles.

La prueba del sistema debe enfocarse en validar transacciones completas del negocio.

4.5 Diseña casos de Prueba de Aceptación

<Supervisor de ACS> diseña los casos de prueba de aceptación del *<Cliente o Usuario>* de acuerdo a los Criterios de Aceptación considerados durante la Planificación del Proyecto.

5. Documentación relacionada

<Entradas>

1. Cronograma de Actividades.
2. Plan de Pruebas
3. Diseño de Pruebas
4. Especificación de Requerimientos
5. Especificación del Diseño
6. Especificación de Interfaces
4. Requerimientos de las Interfaces Externas.

<Entregables>

1. Plan de Pruebas.
2. Casos de Prueba Unitaria.
3. Casos de Prueba de Integración.
4. Casos de Prueba de Sistemas.
5. Casos de Prueba de Aceptación.
6. Especificación de Programas Compartidos.
7. Especificación de Datos Compartidos.

6. Lista de Comprobación

Responda a cada una de las preguntas y anote el puntaje correspondiente de acuerdo a lo observado y verificado. Luego, obtenga el Puntaje Total sumando

los puntajes asignados, dividir el Puntaje Total entre el número de preguntas, redondee el valor obtenido. Este valor compárelo con la tabla de valores del puntaje y conocerá la calificación de la fase revisada o inspeccionada.

Valores aplicados al Puntaje:

1= No tiene, no existe 2= Malo 3= Regular

4= Bueno 5= Excelente, Si tiene

Actividades (Preguntas)	**Puntaje** (1-5)
Diseño de Pruebas 1. ¿ Están definidos las características del Diseño de Pruebas? 2. ¿ Están especificados los atributos de calidad? 3. ¿ Están definidos los límites de carga? 4. ¿ Están definidos las pruebas de fuerza? 5. ¿ Están definidos las configuraciones que el software puede necesitar soportar? 6. ¿ Están identificados la compatibilidad con existentes o planeados componentes de sistemas? 7. ¿ Se tiene en cuenta la seguridad del sistema? 8. ¿ Se considera los el almacenamiento limite del software y data relacionada? 9. ¿ Se tiene en cuenta la performance como tiempo de respuesta, exactitud? 10. ¿ Se verifica la instabilidad del software, principalmente en el ambiente del usuario? 11. ¿ Se verifica la confiabilidad y disponibilidad en las especificaciones? 12. ¿ Se verifica la recuperación de fallas de datos y software? 13. ¿ Se verifican la existencia de Guías de usuario? 14. ¿ Se consideran los factores humanos para usabilidad y aceptabilidad del usuario? 15. ¿ Se está considerando las interfaces con otros componentes de sistema? **16.** ¿ Se está considerando las interfaces de hardware?	
Casos de Pruebas 1. ¿Se han identificado todas las funciones del negocio involucradas? 2. ¿Se ha identificado toda la funcionalidad técnica, las características y requerimientos (funcionalidad estructural)? 3. ¿Se han identificado todos los requerimientos operativos? 4. ¿La funcionalidad cubre todas las transacciones del negocio requeridas? 5. ¿La funcionalidad cubre tanto las actividades manuales como las automatizadas? 6. ¿Están los clientes de acuerdo en que se satisfacen todos los requerimientos? 7. ¿Los diseñadores de la aplicación están de acuerdo que se están cubriendo todos los requerimientos? 8. ¿Existen criterios de evaluación para cada función? 9. ¿Se consideraron todas las condiciones de prueba? 10. ¿Se consideraron condiciones atípicas? 11. ¿Están disponibles todas las herramientas de prueba requeridas? 12. ¿Se han cubierto todos los aspectos de seguridad? 13. ¿El cronograma incluye todas las actividades de prueba? 14. ¿Los ambientes y facilidades de prueba podrán manejar todos los	

requerimientos de las actividades de prueba? 15. ¿Están definidos detalladamente los requerimientos funcionales y técnicos de tal forma que permiten el desarrollo de herramientas de prueba? 16. ¿ Se han identificado los datos de entrada y los resultados esperados? 17. ¿ Se han definidos escenarios de pruebas para descubrir errores, omisiones y resultados esperados?	
Análisis de interfaces 1. ¿ Están todos los objetivos de las interfaces técnicamente adecuados y bien entendidos 2. ¿ Son todos los elementos bien definidos 3. ¿ Son todas las restricciones y limites claramente definidos 4. ¿ Fueron todos los diferentes tipos de interfaces tomadas en cuenta y apropiadamente descrito? 5. ¿ Están todas las interfaces funcionales hardware a software especificados en términos cuantificables tales ¿cómo unidades, bits por segundo, formato de mensajes, reglas de prioridad, longitud de palabra, protocolos? 6. ¿ Están todas las interfaces funcionales software a software y todos los niveles de interfaces de datos especificados en términos cuantitativos tales como definición de datos, formatos de datos, reglas de prioridad, contenido de mensajes y protocolo de comunicaciones? 7. ¿ Están todas los requerimientos de performance bien definidos, y son los limites especificados? 8. ¿ Es la criticidad de las interfaces tomada en consideración? 9. ¿ Son todas las interfaces tomadas en cuenta? 10. ¿ Cuál es el impacto si las interfaces son degradadas? 11. ¿ Se pueden probar y mantener todas las interfaces? 12. ¿ Tienen los estándares apropiados bien identificados para las interfaces incluyendo aquellos de la actual aplicación en su ambiente?	

Q Z

Procedimiento
5.6 Pruebas Unitarias

Ejecuta las pruebas unitarias para conocer si los módulos del software trabajan de acuerdo a lo especificado en el diseño, cumpliendo con las condiciones y restricciones especificadas. En este formato se deberá registrar información de las personas que aprobaron el procedimiento, cambios que se han efectuado y los departamentos en los cuales debe ser aplicado.

Aprobado:		
Cargo	Nombre	
Aprobado:		
Cargo	Nombre	
Aprobado:		
Cargo	Nombre	
Aprobado:		
Cargo	Nombre	
Aprobado:		
Cargo	Nombre	

Cambios

Revisión	Fecha	Persona Responsable	Descripción del Cambio
		Nombre	Revisión Inicial

Lista de Distribución

(lista de los departamentos que reciben copia del procedimiento)

1. Propósito

- Asegurar que la lógica del módulo sea completa y correcta.
- Asegurar que los componentes trabajan como se ha diseñado.
- Verificar que el resultado de la prueba corresponda al Diseño detallado.
- Generar reportes y resolución de anomalías.

2. Alcance

Las pruebas Unitarias de Software deberán ser realizadas a las unidades mínima de software desde el ambiente de Pruebas.

3. Responsabilidades

<*Supervisor de ACS*> Encargado de la planificación y cumplimiento de las pruebas Unitarias del Software. Coordina con el <*Gerente del Proyecto*> en el calendario de las pruebas.

<*Analista de ACS*> Encargado de la ejecución de las Pruebas Unitarias.

<*Gerente del Proyecto*> Proporciona las facilidades para efectuar las pruebas.

4. Procedimiento

4.1 Planifica las Pruebas

<*Supervisor de ACS*> determina si los elementos de software (por ejemplo unidades y módulos) se están implementando correctamente de acuerdo a los requerimientos de los componentes.

<*Supervisor de ACS*> verifica el criterio de Cumplimiento con los requerimientos del diseño.

<*Supervisor de ACS*> evalúa el tiempo estimado, las dimensiones y la exactitud.

<*Supervisor de ACS*> determina la performance en los limites e interfaces bajo condiciones de esfuerzo y error.

<*Supervisor de ACS*> determina las medidas de cobertura de pruebas, fiabilidad, y mantenibilidad.

<*Supervisor de ACS*> coordina el Plan de Pruebas Unitarias con el <*Jefe de Proyectos*> para determinar observaciones.

<*Supervisor de ACS*> determina los tipos de pruebas a realizar y revisa los casos de prueba correspondientes.

<*Supervisor de ACS*> define el calendario de las pruebas Unitarias del software. Coordina con los <*Analista de ACS*> para la ejecución de los mismos.

<*Analista de ACS*> ejecuta las Pruebas Unitarias de acuerdo a los procedimientos de pruebas determinados según el calendario especificado previamente.

<*Analista o Supervisor de ACS*> analiza los resultados para determinar que el software correctamente implementa el diseño.

<*Analista o Supervisor de ACS*> informa las recomendaciones al <*Equipo del Proyecto*>.

4.2 Prueba de Interfaz

<*Analista de ACS*> verifica que la información fluya de forma adecuada hacia y desde el módulo que esta siendo probado.

4.3 Prueba Estructura de Datos

<*Analista de ACS*> verifica que los datos que se mantienen temporalmente conservan su integridad durante todos los pasos de ejecución del algoritmo.

4.4 Prueba de Condiciones Limites

<Analista de ACS> prueba las condiciones limites para asegurar que el módulo funciona correctamente en los límites establecidos como restricciones de procesamiento.

4.5 Prueba de Caminos independientes

<Analista de ACS> prueba los caminos independientes ejecutando todos los caminos independientes o caminos básicos de la estructura de control del programa con el fin de asegurar que todas las sentencia del módulo se ejecutan por lo menos una vez.

<Analista de ACS> selecciona otras pruebas como:

Prueba de camino de manejo de errores.

4.6 Reporte de Pruebas

<Analista de ACS> documenta y efectúa un seguimiento de los resultados como se requiere en el Plan de Pruebas Unitarias.

4.7 Registra los resultados obtenidos

<Analista o Supervisor de ACS> registra los resultados obtenidos en *<formatos>* o *<archivos>* los cuales almacena en *<almacén>*. En caso de tener un mecanismo automatizado registrar los datos e información pertinente.

4.8 Informa sobre las pruebas

<Analista o Supervisor de ACS> verifica y evalúa los resultados de la prueba y dirige el informe al *<Gerente del Proyecto>* con copia al *<Gerente de ACS>*

<Gerente Proyecto> lee las observaciones efectuadas por él *<Supervisor de ACS>* y apoya la implantación de las recomendaciones.

<*Gerente de ACS*> controla que las recomendaciones hayan sido implantadas.

5. Documentación relacionada

<*Entradas*>

1. Cronograma de Actividades.
2. Requerimientos del Negocio
3. Especificación de Requerimientos Funcionales y no Funcionales.
4. Especificación de Requerimientos de Interfaces.
5. Especificación de Requerimientos de Diseño.
6. Especificación del Código Fuente
7. Documentación del Usuario.
8. Plan de Pruebas - Unitarias.
9. Casos de Prueba Unitaria
10. Código Fuente y Ejecutable

<*Entregables*>

1. Plan de Pruebas - Unitaria (actualizado).
2. Casos de Prueba Unitaria (actualizado).
3. Informe de Prueba Unitaria
4. Reporte de Anomalías.

6. Lista de Comprobación

Responda a cada una de las preguntas y anote el puntaje correspondiente de acuerdo a lo observado y verificado. Luego, obtenga el Puntaje Total sumando los puntajes asignados, dividir el Puntaje Total entre el número de preguntas, redondee el valor obtenido. Este valor compárelo con la tabla de valores del puntaje y conocerá la calificación de la fase revisada o inspeccionada.

Valores aplicados al Puntaje:

1= No tiene, no existe	2= Malo	3= Regular
4= Bueno	5= Excelente, Si tiene	

Actividades (Preguntas)	**Puntaje** (1-5)
Pruebas de Estructura de Datos Locales 1. ¿ Inicialización o valores implícitos erróneos? 2. ¿ Existen nombres de variables incorrectos o mal escritos? 3. ¿ Existen tipos de datos inconsistentes? 4. ¿ Existen condiciones de excepciones de desbordamientos por arriba o por abajo, o de direccionamiento?	
Pruebas de Interfaces 1. ¿Es igual el número de parámetros de entrada igual al número de argumentos? 2. ¿Coinciden los atributos de los parámetros y los argumentos? 3. ¿Coinciden los sistemas de unidades de los parámetros y de los argumentos? 4. ¿Es igual el número de argumentos transmitidos a los módulos de llamada que el número de parámetros? 5. ¿Son iguales los atributos de los argumentos transmitidos a los módulos de llamada y los atributos de los parámetros? 6. ¿Son iguales los sistemas de unidades de los argumentos transmitidos a los módulos de llamada y de los parámetros? 7. ¿Son correctos el número de los atributos y el orden de los argumentos de las funciones incorporadas? 8. ¿Son consistentes las definiciones de variables globales entre los módulos? 9. ¿Existen referencias a parámetros que no estén asociados con el punto de entrada actual? 10. ¿Se pasan las restricciones como argumentos? 11. ¿Son correctos los atributos de los archivos? 12. ¿Son correctas las sentencias de apertura? 13. ¿Cuadran las especificaciones de formato con las sentencias de E/S? 14. ¿Cuadra el tamaño del buffer con el tamaño del registro? 15. ¿Se abren los archivos antes de usarlos? 16. ¿Se tienen en cuenta las condiciones de fin- de- archivo? 17. ¿Se manejan los errores de E/S? 18. ¿Hay algún error textual en la información de salida?	

7. Análisis de Medición

Aplica las siguientes métricas, de acuerdo a las indicaciones dadas en el Anexo B:

- Indice de Defectos
- Densidad de Defectos

- Estadística de Errores por Fase
- Distribución del Error
- Horas Hombre por principales Defectos detectados
- Indice de Cobertura de Pruebas Funcionales
- Cobertura de Pruebas
- Exactitud de la Prueba

8. Informe de Revisión

INFORME DE PRUEBAS
<Destinatario>
<Fecha>

CONTENIDO

1. TERMINOS DE REFERENCIA
< > Información general sobre el estado actual de las Pruebas Unitarias.
2. PROCEDIMIENTO APLICADO
< > Procedimientos aplicados en la ejecución del Plan de Pruebas Unitarias.
3. RESULTADOS OBTENIDOS
< > Resultados de las pruebas, observaciones y defectos encontrados.
4. RECOMENDACIONES
< > Recomendaciones para atenuar o eliminar los defectos encontrados en la prueba de los módulos.
5. PERSONAL PARTICIPANTE
< > Personal que ha participado para llevar a cabo las pruebas.
6. RECURSOS EMPLEADOS
< > Registra los recursos humanos (horas/hombre), materiales, servicios y gastos incurridos en las pruebas.

QZ

Procedimiento
5.7 Pruebas de Integración

Ejecuta las pruebas de integración para conocer si los módulos del software trabajan de manera coordinada, pudiendo interactuar en conjunto de acuerdo a lo especificado en el diseño, en estas pruebas es importante evaluar que todas las interfaces trabajen de manera coordinada con el software desarrollado.
En este formato se deberá registrar información de las personas que aprobaron el procedimiento, cambios que se han efectuado y los departamentos en los cuales debe ser aplicado.

Aprobado: _____
Cargo Nombre

Aprobado: _____
Cargo Nombre

Aprobado: _____
Cargo Nombre

Aprobado: _____
Cargo Nombre

Aprobado: _____
Cargo Nombre

Cambios

Revisión	Fecha	Persona Responsable	Descripción del Cambio
		Nombre	Revisión Inicial

Lista de Distribución

(lista de los departamentos que reciben copia del procedimiento)

1. Propósito

- Detectar defectos en el Software.
- Asegurar que los requerimientos especificados en el diseño sean cumplidos.
- Verificar que el resultado de la prueba corresponda al Diseño de alto nivel.
- Validar que el producto final cumple con los requerimientos establecidos para la integración de los elementos del software.
- Generar reportes y resolución de anomalías.

2. Alcance

Las pruebas de Integración serán será planificada para todo el Software desde el ambiente de Pruebas.

3. Responsabilidades

<*Supervisor de ACS*> Encargado de la planificación de las pruebas de Integración del Software. Coordina con el <*Gerente del Proyecto*> en el calendario de las pruebas.

<*Analista de ACS*> Encargado de la ejecución de las Pruebas de Integración.

<*Gerente del Proyecto*> Encargado de proporcionar las facilidades para que el <*Equipo de ACS*> pueda realizar sus pruebas.

4. Procedimiento

4.1 Planifica las Pruebas

<*Supervisor de ACS*> determina las características funcionales, de rendimiento, y de diseño especifica que van a probarse. Determina criterio de terminación de cada fase de la prueba y se documenta las restricciones de la planificación.

<*Supervisor de ACS*> determina si los elementos de software (por ejemplo sub - elementos, interfaces) se están implementando correctamente de acuerdo a los requerimientos del diseño.

<*Supervisor de ACS*> determina el criterio de cumplimiento con los requerimientos funcionales en cada etapa de la integración.

<*Supervisor de ACS*> evalúa el tiempo estimado, las dimensiones y la exactitud.

<*Supervisor de ACS*> determina la performance en los limites e interfaces bajo condiciones de estrés y error.

<*Supervisor de ACS*> determina las Medidas de cobertura de pruebas, fiabilidad, y mantenibilidad.

<*Supervisor de ACS*> coordina el Plan de Pruebas de Integración con <*Jefe Proyectos*> para determinar observaciones.

<*Supervisor de ACS*> determina los tipos de pruebas a realizar.

<*Supervisor de ACS*> define el calendario de las pruebas de Integración del software. Coordina con los <*Analista ACS*> para la ejecución de los mismos.

<*Analista de ACS*> ejecuta las pruebas de Integración de acuerdo a los procedimientos de pruebas determinados según el calendario especificado previamente.

<*Analista o Supervisor de ACS*> analiza los resultados para determinar que el software correctamente implementa el diseño.

<*Analista o Supervisor de ACS*> informa sobre tus recomendaciones al <*Equipo de Proyecto*>.

4.2 Prueba de Integración descendente

<*Analista de ACS*> selecciona el módulo principal como controlador de la prueba, disponiendo de resguardos para todos los módulos directamente subordinados al modulo de control principal.

<*Analista de ACS*> selecciona el enfoque de integración, primero - en - profundidad o primero - en - anchura, y se van sustituyendo los resguardos subordinados uno a uno por los módulos reales.

<*Analista de ACS*> lleva a cabo pruebas cada vez que se integra un nuevo modulo.

<*Analista de ACS*> tras terminar cada conjunto de pruebas, se reemplaza otro resguardo con él modulo real.

<*Analista de ACS*> ejecuta la prueba de regresión, para asegurarse que no se han introducido errores.

4.3 Prueba de Integración ascendente

<*Analista de ACS*> agrupa los módulos de bajo nivel y que realicen una sub- función específica del software.

<*Analista de ACS*> describe un controlador o un programa de control de prueba para coordinar la entrada y salida de los casos de prueba.

<*Analista de ACS*> ejecuta las pruebas del grupo de módulos.

<*Analista de ACS*> elimina los controladores y combina los grupos moviéndose hacia arriba por la estructura del programa.

4.4 Prueba de Regresión

<*Analista de ACS*> vuelve a ejecutar un sub-conjunto de pruebas que se han llevado a cabo anteriormente para asegurarse que los cambios no han propagado efectos colaterales no deseados.

4.5 Reporte de Pruebas

<Analista de ACS> documenta y realiza seguimiento a los resultados como se requiere en el Plan de Pruebas de Integración.

4.6 Registra los resultados obtenidos

<Analista o Supervisor de ACS> registra los resultados obtenidos en *<formatos>* o *<archivos>* los cuales almacena en *<almacén>*. En caso de tener un mecanismo automatizado registrar los datos e información pertinente.

4.7 Informa sobre las pruebas

<Analista o Supervisor de ACS> verifica y evalúa los resultados de la prueba y dirige el informe al *<Gerente del Proyecto>* con copia al *<Gerente de ACS>*

<Gerente de Proyecto> lee las observaciones efectuadas por *< Supervisor de ACS>* y apoya la implantación de las recomendaciones.

<Gerente de ACS> controla que las recomendaciones hayan sido implantadas.

5. Documentación relacionada

<Entradas>

1. Cronograma de Actividades.
2. Requerimientos del Negocio
3. Especificación de Requerimientos Funcionales y no Funcionales.
4. Especificación de Requerimientos de Interfaces.
5. Especificación de Requerimientos de Diseño.
6. Documentación del Usuario.
7. Plan de Pruebas - Integración.
8. Casos de Prueba de Integración.
9. Código Fuente y Ejecutable

<Entregables>

1. Plan de Pruebas - Integración (actualizado).
2. Casos de Prueba de Integración (actualizado).
3. Reporte de Anomalías.
4. Informe de Prueba de Integración.

6. Análisis de Medición

Aplica las siguientes métricas, de acuerdo a las indicaciones dadas en el Anexo B:

- Indice de Defectos
- Densidad de Defectos
- Estadística de Errores por Fase
- Distribución del Error
- Horas Hombre por principales Defectos detectados
- Cobertura de Pruebas
- Suficiencia de Pruebas
- Exactitud de la Prueba

7. Informe de Revisión

INFORME DE PRUEBAS
<Destinatario>
<Fecha>

CONTENIDO

1. TERMINOS DE REFERENCIA
< > Información general sobre el estado actual de las Pruebas Integrales.
2. PROCEDIMIENTO APLICADO
< > Procedimientos aplicados en la ejecución del Plan de Pruebas Integrales.
3. RESULTADOS OBTENIDOS
< > Resultados de las pruebas, observaciones y defectos encontrados.
4. RECOMENDACIONES
< > Recomendaciones para atenuar o eliminar los defectos encontrados en la prueba de integración de los módulos.
5. PERSONAL PARTICIPANTE
< > Personal que ha participado para llevar a cabo las pruebas.
6. RECURSOS EMPLEADOS
< > Registra los recursos humanos (horas/hombre), materiales, servicios y gastos incurridos en las pruebas.

Procedimiento
5.8 Pruebas del Sistema

Ejecuta las Pruebas del Sistema para evaluar si el software cumple con las características funcionales y de operación de acuerdo a la Especificación de Requerimientos.
En este formato se deberá registrar información de las personas que aprobaron el procedimiento, cambios que se han efectuado y los departamentos en los cuales debe ser aplicado.

Aprobado: _____
Cargo Nombre

Aprobado: _____
Cargo Nombre

Aprobado: _____
Cargo Nombre

Aprobado: _____
Cargo Nombre

Aprobado: _____
Cargo Nombre

Cambios

Revisión	Fecha	Persona Responsable	Descripción del Cambio
		Nombre	Revisión Inicial

Lista de Distribución

(lista de los departamentos que reciben copia del procedimiento)

1. Propósito

- Detectar defectos en el Software.
- Ejercitar profundamente y completamente todo el sistema
- Validar que el software como una entidad completa cumpla todos los requerimientos funcionales y de operación.
- Preparar y ejecutar tareas de verificación para que el producto del software cumpla con los requerimientos de corrección, integridad, consistencia y exactitud.
- Diseñar caminos de manejo de errores que prueben toda la información procedente de otros elementos del sistema.
- Llevar a cabo una serie de pruebas que simulen la presencia de datos en mal estado en la interfaz del software.
- Participar en la Planificación y el diseño de pruebas del sistema para asegurarse de que el software se prueba en forma adecuada.
- Generar reportes y resolución de anomalías.

2. Alcance

Las pruebas de Sistemas serán planificadas para todo el Software desde el ambiente de Sistema.

3. Responsabilidades

<*Supervisor de ACS*> encargado de la planificación de las pruebas de Integración del Software. Coordina con <*Gerente del Proyecto*> en el calendario de las pruebas.

<*Analista de ACS*> encargado de ejecutar las pruebas de Sistema.

<*Gerente del Proyecto*> encargado de proporcionar las facilidades para que el <*Equipo de ACS*> pueda realizar sus pruebas.

4. Procedimiento

4.1 Planifica las Pruebas

<*Supervisor de ACS*> determina el criterio de cumplimiento con los requerimientos funcionales.

<*Supervisor de ACS*> evalúa el tiempo estimado, las dimensiones y la exactitud.

<*Supervisor de ACS*> determina la performance en los limites e interfaces bajo condiciones de estrés y error.

<*Supervisor de ACS*> determina las Medidas de cobertura de pruebas, fiabilidad, y mantenibilidad.

<*Supervisor de ACS*> coordina el Plan de Pruebas del Sistema con el <*Jefe Proyectos*> para determinar observaciones.

<*Supervisor de ACS*> coordina el Plan de Pruebas del Sistema con <*Soporte Técnico*> para determinar aspectos técnicos del ambiente.

<*Supervisor de ACS*> determina los tipos de pruebas a realizar.

<*Supervisor de ACS*> define el calendario de las pruebas del Sistema. Coordina con los <*Analista de ACS*> para la ejecución de los mismos.

<*Analista de ACS*> ejecuta las pruebas del Sistema de acuerdo a los procedimientos de pruebas determinados según el calendario especificado previamente.

4.2 Prueba de Recuperación

<*Supervisor de ACS*> fuerza el fallo del software de muchas formas y verifica que la recuperación se lleva a cabo apropiadamente. Si la

recuperación es automática, evalúa la corrección de la inicialización, de los mecanismos de recuperación de estado del sistema, de la recuperación de datos y del proceso rearranque.

4.3 Prueba de Seguridad

<*Supervisor de ACS*> verifica que los mecanismos de protección incorporados en el sistema lo protegerán de accesos impropios, debe probar su invulnerabilidad frente a un ataque frontal, pero también debe probarse de su invulnerabilidad a ataques por los flancos o por la retaguardia.

4.4 Prueba de Resistencia

<*Supervisor de ACS*> diseña pruebas en situaciones anormales como (1) diseñar pruebas especiales que generen interrupciones superiores a las que normalmente usa; (2) incrementar las frecuencias de datos de entrada en un orden de magnitud con el fin de comprobar como responden las funciones de entrada; (3) ejecutar casos de prueba que requieran el máximo de memoria o de otros recursos; (4) diseñar casos de pruebas que puedan dar problemas en un sistema operativo virtual o (5) diseñar casos de pruebas que produzcan excesivas búsquedas de datos residentes en disco.

4.5 Prueba de Rendimiento

<*Supervisor de ACS*> prueba el rendimiento del software en tiempo de ejecución dentro del contexto de un sistema integrado. La prueba de rendimiento se da durante los pasos de prueba. Incluso al nivel de la unidad se debe asegurar el rendimiento de los módulos individuales a medida que se llevan a cabo las pruebas de la caja blanca.

4.6 Reporte de Pruebas

<Analista de ACS> identifica que pruebas fueron exitosas y no exitosas. Para las pruebas que fallaron, identifica el problema y documenta la solución.

<Analista de ACS> documenta y realiza seguimiento a los resultados como se requiere en el Plan de Pruebas del sistema

4.7 Analiza los resultados obtenidos

<Analista o Supervisor de ACS> registra los resultados obtenidos en *<formatos>* o *<archivos>* los cuales almacena en *<almacén>*. En caso de tener un mecanismo automatizado registrar los datos e información pertinente.

<Analista o Supervisor de ACS> analiza los resultados para determinar que el software implementa correctamente el diseño.

4.8 Informa sobre las pruebas

<Analista o Supervisor de ACS> analiza los registros de las pruebas para determinar si se requiere cambiar los cronogramas de pruebas y del proyecto.

<Analista o Supervisor de ACS> verifica y evalúa los resultados de la prueba, elabora tus recomendaciones y dirige el informe al *<Gerente del Proyecto>* con copia al *<Gerente de ACS>*

<Gerente de Proyecto> lee las observaciones efectuadas por *< Supervisor de ACS>* y apoya la implantación de las recomendaciones.

<Gerente de ACS> controla que las recomendaciones hayan sido implantadas.

5. Documentación relacionada

<Entradas>

1. Cronograma de Actividades.
2. Requerimientos del Negocio
3. Especificación de Requerimientos Funcionales y no Funcionales.
4. Especificación de Requerimientos de Interfaces.
5. Documentación del Usuario.
6. Plan de Pruebas - Sistema.
7. Casos de Prueba de Sistema
8. Código Fuente y Ejecutable

<Entregables>

1. Plan de Pruebas - Sistema (actualizado).
2. Casos de Pruebas de Sistema (actualizado)
3. Reporte de Anomalías.
4. Informe de Pruebas de Sistema.

6. Análisis de Medición

Aplica las siguientes métricas, de acuerdo a las indicaciones dadas en el Anexo B:

- Indice de Defectos
- Densidad de Defectos
- Estadística de Errores por Fase
- Distribución del Error
- Horas Hombre por principales Defectos detectados
- Cobertura de Pruebas

7. Informe de Revisión

INFORME DE PRUEBAS
<Destinatario>
<Fecha>

CONTENIDO

1. TERMINOS DE REFERENCIA
< > Información general sobre el estado actual de las Pruebas del Software.
2. PROCEDIMIENTO APLICADO
< > Procedimientos aplicados en la ejecución del Plan de Pruebas del Sistema.
3. RESULTADOS OBTENIDOS
< > Resultados de las pruebas, observaciones y defectos encontrados.
4. RECOMENDACIONES
< > Recomendaciones para atenuar o eliminar los defectos encontrados en la prueba del software.
5. PERSONAL PARTICIPANTE
< > Personal que ha participado para llevar a cabo las pruebas.
6. RECURSOS EMPLEADOS
< > Registra los recursos humanos (horas/hombre), materiales, servicios y gastos incurridos en las pruebas.

Procedimiento
5.9 Pruebas de Aceptación

Ejecuta las Pruebas de Aceptación para evaluar si el software cumple con las características requeridas por el Cliente o Usuario teniendo en cuenta los Criterios de Aceptación definidos en las etapas de Planificación del Proyecto y Análisis.
En este formato se deberá registrar información de las personas que aprobaron el procedimiento, cambios que se han efectuado y los departamentos en los cuales debe ser aplicado.

Aprobado: _____
Cargo Nombre

Aprobado: _____
Cargo Nombre

Aprobado: _____
Cargo Nombre

Aprobado: _____
Cargo Nombre

Aprobado: _____
Cargo Nombre

Cambios

Revisión	Fecha	Persona Responsable	Descripción del Cambio
		Nombre	Revisión Inicial

Lista de Distribución

(lista de los departamentos que reciben copia del procedimiento)

1. Propósito

- Verificar que la validación del software por parte del usuario cumpla con las especificaciones reflejadas en los criterios de aceptación del usuario de la fase de requerimientos.
- Dar cumplimiento a los criterios de aceptación.
- Determinar si el software implementa correctamente los requerimientos de software y hardware en el ambiente operacional.
- Verificar la documentación del usuario.
- Verificar que el software pueda ser exitosamente instalado y operado por el usuario.
- Generar reportes y resolución de anomalías.

2. Alcance

Las pruebas de aceptación serán planificadas para todo el Software desde el ambiente del Sistema. Las pruebas de aceptación se darán al nivel de los requerimientos del negocio o requerimientos funcionales.

3. Responsabilidades

<Jefe de Proyecto> Encargado de proporcionar las facilidades para que el *<Equipo de ACS>* pueda realizar sus pruebas.

<Supervisor de ACS> Encargado de la planificación de las pruebas de Aceptación del usuario. Coordina con el *<Jefe de Proyecto>* en el calendario de las pruebas.

<Gerente de Proyecto> Encargado del Proyecto de Desarrollo o implementación.

<Cliente o Usuario> Encargado de ejecutar y comprobar que el software funciona de acuerdo a las actividades que realiza o requerimientos funcionales.

4. Procedimiento

4.1 Criterios de Aceptación

Aceptación es la fase en que se demuestra que el software esta conforme a los requerimientos.

<Supervisor de ACS> define la comprobación de la aceptación de la solución desde la Fase de los Requerimientos del Proyecto y comprueba estén de acuerdo ambas partes (Cliente y Equipo de Proyecto). Identifica diferencias entre el sistema y sus requerimientos por medio de una prueba de campo o prueba de aceptación del Cliente.

<Cliente> como procedimiento de la aceptación firmará formalmente un documento para denotar la aceptación de cada entregable del proyecto. Se ejecuta la comprobación de la aceptación durante cada fase. Las áreas que se cubren en la aceptación incluye:

Area	Explicación
Funcionalidad	Todas las características de los requerimientos.
Expectativas de Calidad	Todo lo aplicable de las listas de ACS ejm: Performance, fiabilidad, factor humano, seguridad, etc.
Ambiente	Equipos, Configuraciones, interfaces con otros sistemas.
Compatibilidad	Con versiones anteriores.
Prueba de Esfuerzo	Carga máxima (en corto tiempo).
Prueba de Volumen	Registros máximos (en corto tiempo).
Documentos	Todos los requerimientos, exactos y correctos.

4.2 Planificación de las Pruebas

<Supervisor de ACS> revisa los criterios de aceptación definidos en la fase del análisis de requerimientos.

<*Supervisor de ACS*> evalúa el tiempo estimado, las dimensiones y la exactitud.

<*Supervisor de ACS*> desarrolla los procedimientos de aceptación para la prueba de aceptación.

<*Supervisor de ACS*> coordina el Plan de Pruebas de aceptación con el <*Jefe de Proyecto*> para determinar observaciones.

<*Supervisor de ACS*> determina el tipo de pruebas y casos de pruebas de aceptación.

<*Supervisor de ACS*> verifica la adecuación de la documentación del usuario.

<*Supervisor de ACS*> ejecuta las pruebas de aceptación en concordancia con los procedimientos de pruebas.

<*Analista de ACS*> asegura que el ambiente de prueba es el mismo que el ambiente real de producción y que el mismo <*Cliente o Usuario*> sea quien ejecuta las pruebas.

4.3 Analiza y Documenta los Resultados

<*Analista de ACS*> analiza los resultados para determinar si el software satisface el criterio de aceptación.

<*Analista o Supervisor de ACS*> documenta y controla los resultados de las pruebas.

4.4 Reporte de Pruebas

<*Analista de ACS*> documenta y realiza un seguimiento a los resultados como se requiere en el Plan de Aceptación.

4.5 Logra la Aceptación del Cliente

<Analista de ACS> formaliza la aprobación o rechazo del *<Cliente o Usuario>*. Documenta y determina el impacto de los comentarios del *<Cliente o Usuario>*.

4.6 Registra los resultados obtenidos

<Analista o Supervisor de ACS> registra los resultados obtenidos en *<formatos>* o *<archivos>* los cuales almacena en *<almacén>*. En caso de tener un mecanismo automatizado registrar los datos e información pertinente.

4.7 Informa sobre las pruebas

<Analista o Supervisor de ACS> verifica y evalúa los resultados de la prueba y dirige el *<informe>* al *<Gerente del Proyecto>* con copia al *<Gerente de ACS>*

<Gerente de Proyecto> lee las observaciones efectuadas por *<Supervisor de ACS>* y apoya la implantación de las recomendaciones.

<Gerente de ACS> controla que las recomendaciones hayan sido implantadas.

5. Documentación relacionada

<Entradas>

1. Cronograma de Actividades.
2. Requerimientos del Negocio
3. Especificación de Requerimientos Funcionales y no Funcionales.
4. Especificación de Requerimientos de Interfaces.
5. Documentación del Usuario.
6. Plan de Pruebas - Aceptación.
7. Casos de Prueba de Aceptación.
8. Criterios de Aceptación

<Entregables>

1. Plan de Pruebas - Aceptación (actualizado).
2. Casos de Pruebas de Aceptación (actualizado).
3. Reporte de Anomalías.
4. Informe de Pruebas de Aceptación

6. Análisis de Medición

Aplica las siguientes métricas, de acuerdo a las indicaciones dadas en el Anexo B:

- Indice de Defectos
- Densidad de Defectos
- Estadística de Errores por Fase
- Distribución del Error
- Horas Hombre por principales Defectos detectados
- Cobertura de Pruebas

7. Informe de Revisión

INFORME DE PRUEBAS
<Destinatario>
<Fecha>

CONTENIDO

1. TERMINOS DE REFERENCIA
< > Información general sobre el estado actual de las Pruebas efectuadas al software.
2. PUNTOS DE ATENCION
< > Comentarios del <Cliente o Usuario> sobre la aceptación o rechazo del software desarrollo.
4. RECOMENDACIONES
< > Recomendaciones a seguir para que el <Cliente o Usuario> logre dar su aceptación al software desarrollado.
5. PERSONAL PARTICIPANTE
< > Personal que ha participado para llevar a cabo las pruebas.
6. RECURSOS EMPLEADOS
< > Registra los recursos humanos (horas/hombre), materiales, servicios y gastos incurridos en las pruebas.

Capitulo 6

PLAN DE CONFIGURACION DEL SOFTWARE

6.1 Propósito

El Propósito del Plan de Configuraciones es definir un esquema general para organizar, administrar y evaluar la Gestión de Configuraciones y Controlar los cambios para el mantenimiento de la integridad de los elementos y/o versiones del Software a lo largo del Ciclo de Vida.

6.2 Alcance

Este plan se aplica al Software desarrollado en la empresa o comprado a terceros que estén bajo el control de la Gestión de Configuraciones.

6.3 Areas Funcionales involucradas

<Areas Usuarias> Usuarios del Aplicativo o Software

<Sistemas> Gerencia de Sistemas o Informática.

<Soporte Técnico> Usuario de encargado del Soporte operativo del Sistema de Configuraciones.

<Control de Cambios> Administrador del Sistema de Gestión de Configuraciones.

<Departamento de ACS> Verifica las actividades de Control de Cambios y Configuraciones.

6.4 Actividades

- **Revisión del Ambiente de Configuración**

  Permite verificar que existan los recursos de computo necesarias para establecer una efectiva Gestión de Configuraciones en el Proyecto de Software.

- **Control de Cambios**

  Permite verificar que los controles para los cambios se encuentren incorporados en forma adecuada para la efectiva Gestión de los cambios de programas.

- **Control de Versiones**

  Permite verificar que los controles para las versiones se encuentren incorporados para que el programador pueda mantener versiones y niveles de programas y ejecutables.

- **Control de Release's**

  Permite verificar la existencia de controles en la Gestión de los paquetes liberados de software para distribución al cliente o usuario.

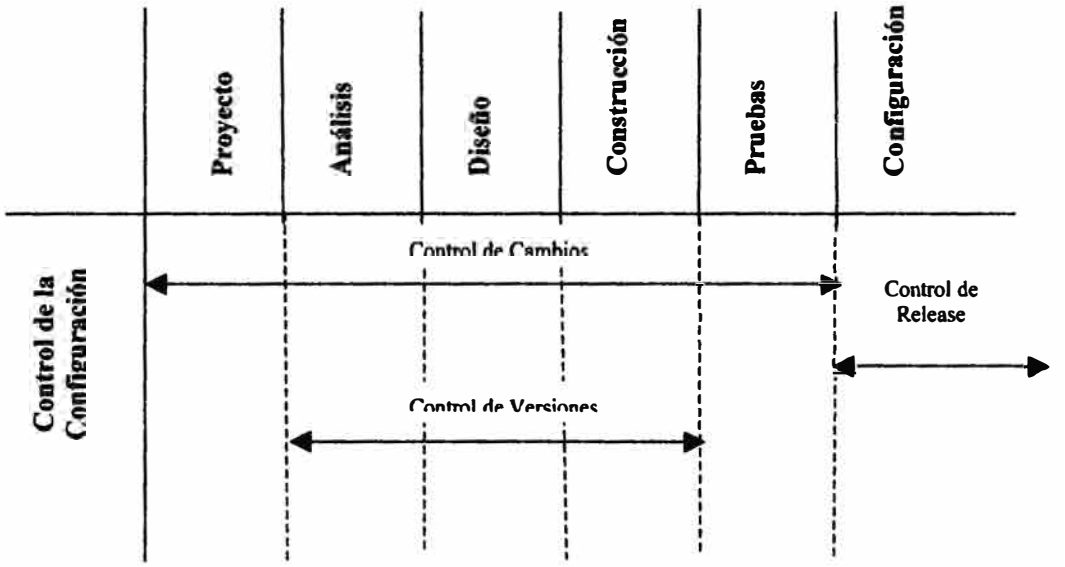


Fig. Proceso del Control de Configuración del Software

Fuente: Elaboración Propia

Procedimiento
6.5 Revisión del Ambiente de Configuración

La Revisión del Ambiente de Configuración es importante; pues, se podrá conocer el ambiente de control existente durante la ejecución de los cambios que se efectúen al software o sus elementos.
En este formato se deberá registrar información de las personas que aprobaron el procedimiento, cambios que se han efectuado y los departamentos en los cuales debe ser aplicado.

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Cambios

Revisión	Fecha	Persona Responsable	Descripción del Cambio
		Nombre	Revisión Inicial

Lista de Distribución

(lista de los departamentos que reciben copia del procedimiento)

1. Propósito

- Verificar que la instalación del Cliente o Usuario tenga las condiciones de uso, operatividad, seguridad, y satisfacción de los objetivos del Plan de Configuraciones; de tal manera, que permita mantener la Calidad en el Software.

- Verificar la implementación y definición de las etapas del desarrollo de Software en el ambiente de Configuración.

- Definir las acciones para definir, adicionar, actualizar, y recuperar los elementos de las configuraciones desde los repositorios controlados.

- Evaluar los requerimientos necesarios de las herramientas para la automatización de las actividades de la Gestión de las Configuraciones.

2. Alcance

Este procedimiento será utilizado por el equipo de ACS y se aplica durante la etapa de Planificación del Proyecto de Software.

3. Responsabilidades

<*Analista de ACS*> Ejecuta la revisión.

<*Control de Cambios*> Proporciona al <Analista ACS> los procedimientos operativos y documentación necesaria para la ejecución de la revisión. Es encargada de la operación del Control de Pases de nuevas aplicaciones a la Gestión de Configuraciones, Control de Cambios y Control de Versiones.

<*Soporte Técnico*> Realiza operaciones de mantenimiento al Software de Gestión de Configuraciones y encargados de los respaldos y procedimientos técnicos.

<*Gerente del Proyecto*> Encargado del Proyecto de Desarrollo o implementación.

4. Procedimiento

4.1 General

Ver Especificaciones Técnicas del Software de Configuración

Ver Anexo de Herramientas de Gestión de Configuración

<*Analista de ACS*> verifica si la instalación cumple con los requisitos funcionales necesarios para la Gestión de Configuraciones.

<*Supervisor de ACS*> define por cada tipo de aplicación y/o proyecto de software los elementos de la configuración.

4.2 Identificación de los elementos de la Configuración

<*Analista o Supervisor de ACS*> identifica los elementos de la Configuración a ser controlados en el proyecto. Registra y toma un inventario de los componentes, elementos o programas a ser controlados; entre estos se consideran los siguientes:

Elementos de Configuración del Software
Plan del Proyecto de Software
Especificaciones del Requerimiento Funcional
Especificaciones del Requerimiento Técnico
Prototipos ejecutables o en papel
Diagrama Lógico de Datos
Diagrama físico de Datos
Diagrama de Flujo de Datos
Descripción de las Interfaces
Descripción de Objetos
Diccionario de Datos
Estructura de Archivos
Contenido Inicial
Programas ejecutables
Módulos y Códigos ejecutables
Módulos enlazados
Procedimientos almacenados
Plan y Procedimientos de Pruebas
Casos de Pruebas y resultados de pruebas
Manual de Usuario Final

Manuales de Operación e Instalación
Informes de Problemas de Software
Requerimientos y Ordenes de Control de Cambios
Estándares y procedimientos de ingeniería de Software
Lenguaje de Programación utilizado en el desarrollo
Herramientas, editores, depuradores, CASE, etc.
Documentación Técnica del Hardware

4.3 Instalación del Ambiente de Configuración

<Analista de ACS> verificación de seguridad funcional, física y disponibilidad operativa de repositorios o librerías.

<Analista de ACS> solicita información de las herramientas actuales instaladas para la Gestión de Configuraciones, procedimientos existentes.

<Analista de ACS> verifica si la instalación cumple con los requisitos funcionales necesarios para la Gestión de Configuraciones.

<Analista de ACS> solicita reuniones de coordinación con *<Soporte Técnico>* para la comprobación de la instalación y operatividad de las herramientas usadas.

4.4 Verifica la Herramienta de Gestión de la Configuración

<Analista o Supervisor de ACS> verifica que los requisitos generales del Software de Configuraciones cumpla con las siguientes características:

- **Manejo de Inventarios**

  Deberá permitir organizar las aplicaciones en forma lógica para efectos de Identificación y localización de los objetos individuales y relacionados. Definición de Ordenes o Identificadores de Control de Cambios.

- **Manejo de Ciclo de Vida**

  Deberá permitir definir los ambientes y etapas de acuerdo al proceso de desarrollo de la Organización, definir las Rutas de Promoción del Software a través de los ambientes y etapas definidas.

- **Manejo de Cambios**

  Manejar automáticamente los cambios de todos los elementos y las relaciones de sus componentes estén bajo su Control.

- **Manejo de Versiones**

  Control de versión para archivos tipo textos o binarios, que permita registrar, recuperar y comparar las versiones de los componentes de las configuraciones. Proporcionar el manejo de paquetes de elementos, para mover configuraciones o elementos individuales. Prever posibles conflictos en el desarrollo paralelo, permitiendo detectar cambios no previstos, notificando a los afectados y efectuar una conciliación adecuada.

- **Manejo Desarrollo Distribuido**

  Permita el desarrollo en forma distribuida o a través de múltiples plataformas, las que se utilizan en la Organización.

- **Manejo Desarrollo Paralelo o Concurrente**

  Detectar problemas sobre cambios concurrentes, identificación y detección automática, comparación, y notificación de conflictos de código fuentes.

- **Manejo de Componentes**

  Proporcionar características que permitan definir la relación fuente - ejecutables. Controlar la dependencia de los componentes en las configuraciones de los programas afectados a los cambios. Mantenimiento de Historia de Configuraciones.

- **Interfaces a Herramientas de Desarrollo**

  Integrarse con herramientas de Desarrollo de Software comúnmente usados.

- **Informes de Estado y Auditoria**

  Reportes de auditoria que permita determinar ¿Quién?, ¿Qué cambios?, ¿Dónde?, ¿Cuándo?, ¿Por qué del cambio?.

- **Seguridad**

  Soportar seguridad para la función que se desea ser ejecutada. El acceso es autorizado o denegado a aplicaciones o grupos de archivos. Proporcionar Repositorios de Seguridad, para proteger el código fuente y la historia de cambios. Asignación de elementos a uno, ninguno o múltiples usuarios.

4.5 Aplica lista de comprobación

<Analista de ACS> aplica las preguntas de la Lista de Comprobación de acuerdo a los tópicos de control de interés para la organización.

4.6 Evalúa y mide los riesgos

<Analista o Supervisor de ACS> detecta las áreas criticas y los problemas potenciales que se puedan suscitar en la Gestión de Configuraciones.

<Analista o Supervisor de ACS> efectúa una evaluación de los resultados obtenidos; determina los riesgos en que se pueda estar incurriendo.

<Analista o Supervisor de ACS> define la prioridad de los riesgos. Indica las recomendaciones necesarias para atenuar o eliminar los riesgos.

4.7 Registra los resultados obtenidos

<Analista o Supervisor de ACS> registra los resultados obtenidos en *<formatos>* o *<archivos>* los cuales almacena en *<almacén>*. En caso de tener un mecanismo automatizado registrar los datos e información pertinente.

4.8 Informa sobre la revisión

<Analista o Supervisor de ACS> verifica y evalúa la Revisión de la Gestión de la Configuración y dirige el informe al *<Gerente del Proyecto>* con copia al *<Gerente de ACS>*.

<Gerente de Proyecto> lee las observaciones efectuadas por *<Supervisor de ACS>* y apoya la implantación de las recomendaciones.

<Gerente de ACS> controla que las recomendaciones hayan sido implantadas.

5. Documentación relacionada

<Entradas>

1. Plan de Gestión de Configuraciones.
2. Procedimientos de la Gestión de Configuraciones.
3. Plan de Revisión de ACS.
4. Lista de Comprobación.

<Entregables>

1. Análisis y evaluación de riesgos.
2. Informe de la revisión.
3. Reporte de Anomalías.
4. Plan de Revisión de ACS (Actualizado).

6. Lista de Comprobación

Responda a cada una de las preguntas y anote el puntaje correspondiente de acuerdo a lo observado y verificado. Luego, obtenga el Puntaje Total sumando los puntajes asignados, dividir el Puntaje Total entre el número de preguntas, redondee el valor obtenido. Este valor compárelo con la tabla de valores del puntaje y conocerá la calificación de la fase revisada o inspeccionada.

Valores aplicados al Puntaje:

1= No tiene, no existe 2= Malo 3= Regular

4= Bueno 5= Excelente, Si tiene

Actividades (Preguntas)	**Puntaje** (1-5)
Gestión de Configuraciones en la Organización 1. ¿Existe incorporada la función de Gestión de Configuraciones? 2. ¿Controla la organización los cambios antes y después de que el software sea distribuido al cliente o usuarios? 3. ¿Están definidas las etapas del Ciclo de Vida en el Sistema de Gestión de Configuraciones? 4. ¿Están identificados los elementos y las aplicaciones que estarán bajo el control de Configuraciones? 5. ¿Existe un documento de Control de Cambios? 6. ¿Existe un Control de Ordenes de Cambio? 7. ¿Existe un procedimiento de Pases a Producción? 8. ¿Identifica y gestiona las versiones existentes de un programa y su documentación de forma que se pueda introducir cambios eficientemente? 9. ¿Quién tiene responsabilidad de aprobar y de asignar prioridades a los cambios? 10. ¿Podemos garantizar que los cambios se han llevado adecuadamente? 11. ¿Se han considerado los efectos laterales asociados con el cambio? 12. ¿Se ha documentado, evaluado y aprobado la orden de cambios? 13. ¿Se ha documentado el cambio, una vez ejecutado, e informado a las partes interesadas?	
Plan de Gestión de Configuraciones 1. ¿Existe un Plan de Configuraciones de Software? 2. ¿Están identificadas las responsabilidades y autoridad para el cumplimiento de estas actividades (Quien)? 3. ¿Están identificadas las actividades a ser ejecutadas en aplicación del proyecto (Que)? 4. ¿Están identificadas las coordinaciones requeridas de las	

actividades de Gestión de Configuraciones en el Proyecto, (Cuando)? 5. ¿Están identificadas las herramientas, y los recursos físicos, humanos requeridos para la ejecución del Plan (Como)? 6. ¿Se identifica quien realizara el mantenimiento del Plan?	
Herramienta de Gestión de Configuraciones 1. ¿Existe una herramienta automatizada de Gestión de Configuraciones? 2. ¿Satisfacen los requerimientos funcionales de Gestión de Configuraciones? 3. ¿Se encuentra correctamente instalada e implementada? 4. ¿Permite Control de Cambios? 5. ¿Permite Control de niveles de Versiones? 6. ¿Permite Desarrollo paralelo o multiusuario? 7. ¿Qué mecanismo se usa para avisar a otros de los cambios realizados? 8. ¿Permite Determinar sincronización fuente - ejecutable? 9. ¿Existe un mecanismo de seguridad de repositorios protegidos de accesos indebidos? 10. ¿Se ha realizado un programa de entrenamiento de uso de la herramienta?	
Identificación de elementos en la Configuración del Software 1. ¿Están definidos los tipos de objetos en la Configuración del Software? 2. ¿Son registrados e identificados en forma única? 3. ¿Se organizan mediante un enfoque orientado a objetos? 4. ¿Esta definido quien y como realizara el registro de las nuevas configuraciones? 5. ¿Es posible obtener una historia de cambios de cada objeto de la configuración? 6. ¿Es posible tener versiones y variantes de las configuraciones? 7. ¿Se están registrando las configuraciones de software en el proyecto de software? 8. ¿Quién es el encargado de actualizar las definiciones del inventario de configuraciones? 9. ¿Existe un procedimiento que permita incorporar nuevas configuraciones a la Gestión e Configuraciones? 10. ¿Podemos relacionar todos los elementos de una configuración? 11. ¿Podemos relacionar la fuente y sus ejecutables?	

7. Informe de Revisión

INFORME DE REVISION
<Destinatario>
<Fecha>

CONTENIDO

1. TERMINOS DE REFERENCIA
< > Información general sobre el estado actual de la Gestión de Configuraciones, observaciones y restricciones de las herramientas usadas, del proyecto, del personal involucrado, de los requerimientos y necesidades actuales.
2. PROCEDIMIENTO APLICADO
< > Procedimiento de Revisión de Ambiente de Configuración.
3. RESULTADOS OBTENIDOS
< > Riesgos detectados en la revisión del Ambiente de Configuración, lista de requerimientos y necesidades.
4. RECOMENDACIONES
< > Recomendaciones para atenuar o eliminar los riesgos en la Gestión de Configuraciones del Software.
5. PERSONAL PARTICIPANTE
< > Personal que ha participado para llevar a cabo la revisión.
6. RECURSOS EMPLEADOS
< > Registra los recursos humanos (horas/hombre), materiales, servicios y gastos incurridos en la revisión.

Procedimiento

6.6 Revisión del Control de Cambios

La Revisión del Control de Cambios se ejecuta para evaluar el control existente en prevenir la introducción de cambios no autorizados o errores, debiendo verificar que estos pasen por un proceso de prueba y aceptación antes de su pase a producción.
En este formato se deberá registrar información de las personas que aprobaron el procedimiento, cambios que se han efectuado y los departamentos en los cuales debe ser aplicado.

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Cambios

Revisión	Fecha	Persona Responsable	Descripción del Cambio
		Nombre	Revisión Inicial

Lista de Distribución

(lista de los departamentos que reciben copia del procedimiento)

1. Propósito

- Verificar el cumplimiento de las funciones de la Gestión de Configuración durante el proceso de desarrollo del Proyecto de Software.
- Organizar y Definir que aplicaciones y elementos de programas estarán bajo el control de configuraciones.
- Identificar y determinar el origen del cambio.
- Controlar la actualización simultanea de un determinado elemento del software por mas de una persona.
- Identificar y controlar todas las acciones y cambios resultantes de los requerimientos de cambio de inicio a fin.
- Verificar la ejecución de procedimientos operativos de Control de Cambios y promoción a Producción.

2. Alcance

Este Procedimiento es utilizado por el equipo de ACS y se utilizara para realizar una revisión del Control de cambios de las Configuraciones en la Organización y se aplica en cualquier etapa del Desarrollo del Software.

3. Responsabilidades

<Analista o Supervisor de ACS> Verifica que se apliquen las funciones de Control de Cambios en la Instalación.

<Control de Cambios> Encargado de la Gestión de Control de Cambios, obtiene reportes de auditoria.

<Soporte Técnico> Encargada del Seguridad funcional y operativa de la herramienta de Gestión de Configuraciones.

<Gerente del Proyecto> Encargado del Desarrollo, Mantenimiento o implementación del Proyecto de Software.

4. Procedimiento

4.1 General

<Analista de ACS> solicita información de las herramientas actuales instaladas y procedimientos existentes para la Gestión de Configuraciones.

4.2 Revisa y analiza la solicitud del cambio

<Analista o Supervisor de ACS> determina sí el análisis de factibilidad, el esfuerzo técnico, los efectos secundarios, el impacto global sobre otras funciones del sistema y los costos estimados, han sido elaborados, analizados y estimados adecuadamente.

<Analista o Supervisor de ACS> verifica que la *<Solicitud de Cambio>* siga por un procedimiento formal y adecuado, en que de acuerdo a la etapa en que se encuentre se registre información tal como: elementos de programas que estarán sujetos al cambio, librerías de congelamiento, fecha de aprobación, indicador de urgencia, el área solicitante, responsables del cambio, descripción del cambio, versiones afectadas, la fecha de solicitud, etc.

4.3 Identifica y determina el tipo del Cambio

<Analista de ACS> verifica que en la orden del cambio se haya especificado el tipo del cambio, el cual puede ser por: **defectos y/o correcciones**, mejoras, cambios tecnológicos, la severidad y prioridad del cambio entre otros.

4.4 Identifica y verifica los Cambios efectuados

<Analista de ACS> verifica que los elementos de la configuración hallan sido recuperados de las librerías de Producción y puestos en las librerías de Desarrollo con la seguridad y restricción de accesos adecuados.

<Analista de ACS> verifica que se haya efectuado el cambio al elemento de software y actualizado la *<Solicitud de Cambio>* correspondiente.

<Analista de ACS> verifica que la *<Solicitud de Cambio>* tenga en visado de aceptado por *<Jefe del Proyecto>*, donde se indiquen claramente los elementos de la configuración a ser reemplazados en las librerías de producción.

<Analista de ACS> verifica que las modificaciones en los programas estén explicados en la *<Solicitud de Cambios>*, así como los responsables, la fecha del cambio, firmas, etc.

4.5 Garantiza los Cambios

<Analista de ACS> verifica que los cambios realizados por *<Sistemas>* hayan sido probados por la respectiva *<Area de Pruebas>* y que corresponden a los requerimientos de las *<Areas Usuarias>*.

<Analista de ACS> verifica que los programas y/o elementos necesarios para el cambio estén disponibles al *<Usuario>;* habiendo sido promocionados en forma correcta a la respectiva Area de Producción.

<Analista de ACS> verifica la seguridad funcional y a las librerías de producción estén restringidas al *<Equipo de Desarrollo>*.

<Analista de ACS> verifica que la documentación del cambio sea registrada y entregada al área pertinente. Asimismo, verifica que la *<Solicitud de Cambio>* no será utilizada en adelante.

4.6 Audita los Cambios

<Analista de ACS> verifica que *<Control de Cambios>* disponga de información de auditoria de los cambios para determinar ¿Quién?, ¿Cómo?, ¿Cuándo?, ¿Dónde?, se realizaron los cambios.

4.7 Controla el Pase a Producción

<Analista de ACS> valida si la corrección se realizo de acuerdo a las instrucciones de la *<Orden de Cambio>*, de no ser así deberá ser devuelto al *<Equipo de Proyecto>* para su corrección, en caso de ser conforme verifica el correcto pase a producción.

4.8 Aplica lista de comprobación

<Analista de ACS> aplica las preguntas de la Lista de Comprobación de acuerdo a los tópicos de control de interés para la organización.

4.9 Evalúa y mide los riesgos

<Analista o Supervisor de ACS> detecta las áreas criticas y los problemas potenciales que se puedan suscitar en la Gestión de Configuraciones.

<Analista o Supervisor de ACS> efectúa una evaluación de los resultados obtenidos; determina los riesgos en que se pueda estar incurriendo.

<Analista o Supervisor de ACS> define la prioridad de los riesgos. Indica las recomendaciones necesarias para atenuar o eliminar los riesgos.

4.10 Informa sobre la revisión

<Analista o Supervisor de ACS> verifica y evalúa la Revisión de la Gestión de la Configuración y dirige el informe al *<Gerente del Proyecto>* con copia al *<Gerente de ACS>*

<Gerente de Proyecto> lee las observaciones efectuadas por él *<Supervisor de ACS>* y apoya la implantación de las recomendaciones.

<Gerente de ACS> controla que las recomendaciones hayan sido implantadas.

4.11 Registra los resultados obtenidos

<Analista o Supervisor de ACS> registra los resultados obtenidos en *<formatos>* o *<archivos>* los cuales almacena en *<almacén>*. En caso de tener un mecanismo automatizado registrar los datos e información pertinente.

5. Documentación relacionada

<Entradas>

1. Documentos u Ordenes de Cambios.
2. Procedimiento de Cambios
3. Procedimiento de Promoción.
4. Análisis y evaluación de riesgos.
5. Plan de Revisión de ACS.
6. Lista de Comprobación.

<Entregables>

1. Análisis y evaluación de riesgos
2. Informe de la revisión
3. Reporte de Anomalías
4. Plan de Revisión de ACS (Actualizado)

6. Lista de Comprobación

Responda a cada una de las preguntas y anote el puntaje correspondiente de acuerdo a lo observado y verificado. Luego, obtenga el Puntaje Total sumando los puntajes asignados, dividir el Puntaje Total entre el número de preguntas, redondee el valor obtenido. Este valor compárelo con la tabla de valores del puntaje y conocerá la calificación de la fase revisada o inspeccionada.

Valores aplicados al Puntaje:

1= No tiene, no existe 2= Malo 3= Regular

4= Bueno 5= Excelente, Si tiene

Actividades (Preguntas)	**Puntaje** (1-5)
1. ¿Existe una autoridad de Control de Cambios en la Organización?	
2. ¿Existen una política de Ordenes de Cambios?	
3. ¿Existe un Procedimiento de Control de Cambios y/o Pases a Producción?	
4. ¿Esta operativa la herramienta automática de Control de Cambios?	
5. ¿Es utilizada la herramienta de Control de Cambios?	
6. ¿Sé esta utilizando el Documento de Control de Cambios?	
7. ¿Sé aperturo una Orden de Cambios?	
8. ¿Las Ordenes de cambio son utilizadas solamente en el proyecto?	
9. ¿Están aprobadas y revisadas los cambios en el nivel de la supervisión?	
10. ¿Esta debidamente lleno el Documento de Control de Cambios?	
11. ¿Se han especificado los cambios, la fecha del cambio, el autor, se identifican las versiones, etc.?	
12. ¿Se han hecho los cambios de acuerdo a lo especificado en la Orden de Cambios, se han hecho modificaciones adicionales?	
13. ¿Se han especificado los cambios, la fecha del cambio, el autor, se identifican las versiones, etc.?	
14. ¿Los cambios solos se realizan en áreas de desarrollo y no de producción?	
15. ¿Están separadas las bibliotecas de desarrollo/prueba/producción?	
16. ¿Se realizo una Revisión técnica de los cambios?	
17. ¿Son los cambios los que solicito el usuario?	
18. ¿Los cambios en los programas fuentes reflejan en la documentación?	
19. ¿Es posible obtener reportes de auditoria?	

20. ¿Es posible determinar quien, como, cuando y donde se realizo el cambio? 21. ¿Se puede asegurar que los programas en código fuente corresponden al código objeto? 22. ¿Esta completa la documentación de todos los cambios a los programas? 23. ¿Existen restricciones de acceso y uso a los programadores a las bibliotecas de producción? 24. ¿Existe la seguridad necesaria de que los cambios serán protegidos?	

7. Análisis de Medición

Aplica las siguientes métricas, de acuerdo a las indicaciones dadas en el Anexo B:

- Indice de Defectos
- Densidad de Defectos
- Estadística de Errores por Fase
- Distribución del Error
- Horas Hombre por principales Defectos detectados
- Número de Días de la Falla

8. Informe de Revisión

INFORME DE REVISION
<Destinatario>
<Fecha>

CONTENIDO

1. TERMINOS DE REFERENCIA
< > Información general sobre actividades actuales efectuadas para el manejo y control de cambios solicitados por el Cliente o Usuario, cambios en el software, entre otros.
2. PROCEDIMIENTO APLICADO
< > Procedimiento aplicado en la revisión del proceso de cambios.
3. RESULTADOS OBTENIDOS
< > Riesgos detectados en la revisión del Proceso de Cambios, lista de comprobación y observaciones.
4. RECOMENDACIONES
< > Recomendaciones para atenuar o eliminar los riesgos en el proceso de cambios.
5. PERSONAL PARTICIPANTE
< > Personal que ha participado para llevar a cabo la revisión.
6. RECURSOS EMPLEADOS
< > Registra los recursos humanos (horas/hombre), materiales, servicios y gastos empleados.

8. Formatos

<Formato de Solicitud de Cambio> Este formato consta de 1 página, deberá ser utilizado para hacer el pedido formal del cambio al software en etapa de explotación. Sirve de control como rastro de auditoria, registrándose datos generales sobre la inspección efectuada; tales como: nombre del proyecto, encargados de la inspección, resultados, observaciones, comentarios, defectos o errores encontrados, herramientas utilizadas. Tener en cuenta que los datos del Solicitante y Descripción Detallada del Cambio serán llenados por el Usuario o Cliente, los demás datos serán completados por el *<Equipo de Desarrollo>*. El contenido de este formato servirá de referencia para la elaboración de informes sobre las inspecciones.

<Formato de Autorización de Cambio> Este formato consta de 1 página, deberá ser utilizado para autorizar el pase a producción del cambio realizado por el *<Equipo de Desarrollo>*. Este formato deberá utilizarse junto al formato *<solicitud de cambio>*, el contenido de este formato servirá de referencia para la elaboración de informes sobre las inspecciones y control de los cambios efectuados al software.

QZ	**Solicitud de Cambio**	Num: ____ Pag:1/2

Solicitado por _____ Nombre del Proyecto _____ Programa / Documento _____ Versión / Release	Tipo de la Solicitud de Cambio (1) Correctivo (2) Adaptativo (3) Perfectivo Fecha: ___/___/___ Descripción Corta de la Tarea _____

Descripción Detallada _____ _____

Prioridad (1) Critico (2) Muy Importante (3) Importante (4) Inoportuno (5) Interesante Acción _____ Asignado a	Acción _____ Fecha de entrega: ___/___/___ Fecha del actual Release: ___/___/___ N° de Ref. de la Autorización de Cambio

Comentarios de la Solución _____ _____

Programas afectados _____ _____

Aprobado	Desarrollo	Control de Cambios	SQA	Producción
Firma				
Fecha	___/___/___	___/___/___	___/___/___	___/___/___

Q Z	**Autorización del Cambio _____ Num: ___ Pag: 2/2**

Solicitud _____	Versión del Producto _____
Sistema _____	Nombre del Servidor _____
Fecha / /	Computador

Código / Elemento	Release / Versión	Tipo de Módulo	Detalle del Cambio

Comentarios
_____
_____
_____
_____

Aprobado	Desarrollo	Control de Cambios	SQA	Producción
Firma				
Cargo				
Fecha	___/___/___	___/___/___	___/___/___	___/___/___

QZ

Procedimiento
6.7 Revisión del Control de Versiones

La Revisión del Control de Versiones se ejecuta para evaluar el control existente en prevenir cambios por parte del Equipo de Desarrollo a versiones incorrectas o desactualizadas de elementos del software.
En este formato se deberá registrar información de las personas que aprobaron el procedimiento, cambios que se han efectuado y los departamentos en los cuales debe ser aplicado.

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Aprobado: _____
cargo Nombre

Cambios

Revisión	Fecha	Persona Responsable	Descripción del Cambio
		Nombre	Revisión Inicial

Lista de Distribución

(lista de los departamentos que reciben copia del procedimiento)

1. Propósito

- Identificar las versiones de los elementos del software los cuales juntos constituyen una versión especifica de un producto completo.
- Garantizar que los cambios simultáneos que se haga al código fuente no se vea afectado por conflictos.
- Identificar el nivel de construcción del producto de software en el desarrollo, instalación y/o distribución.

2. Alcance

Este Procedimiento es utilizado por el <*Equipo de ACS*> y se utilizara para verificar el Control de Versiones del software desarrollado internamente y/o por terceros.

3. Responsabilidades

<*Analista o Supervisor de ACS*> verifica que se apliquen las funciones de Control de Versiones en la Instalación.

<*Soporte Técnico*> encargada del Seguridad funcional y operativa de la herramienta de Gestión de Configuraciones.

<*Gerente del Proyecto*> encargado del Desarrollo, Mantenimiento o implementación del Proyecto de Software.

<*Control de Cambios*> verifica que los cambios en el Código Fuente se realicen de acuerdo a normas internas.

4. Procedimiento

4.1 General

<Analista de ACS> solicita información de las herramientas actuales instaladas y procedimientos existentes para la Gestión de Configuraciones.

<Analista de ACS> verifica la confirmación de los requerimientos necesarios en la gestión de versiones de la herramienta de Gestión de Configuraciones.

4.2 Verifica las facilidades de la Herramienta

<Analista de ACS> verifica que la opción de Control de Versiones permite identificar, registrar y recuperar versiones del código fuente, mover elementos individuales o paquetes através del ciclo.

4.3 Verifica el uso de la Herramienta

<Analista de ACS> verifica que la opción de Control de Versiones este disponible en la estación de trabajo del *<Equipo de Proyecto>* para el manejo del código fuente.

<Analista de ACS> verifica que la herramienta se utilice.

4.4 Verifica el uso de Estándares

<Analista o Supervisor de ACS> verifica que en los cambios realizados al código fuentes se utilicen estándares que documenten estos cambios.

4.5 Informa sobre la revisión

<Analista o Supervisor de ACS> verifica y evalúa la Revisión de la Gestión de la Configuración y dirige el informe al *<Gerente del Proyecto>* con copia al *<Gerente de ACS>*.

<Gerente de Proyecto> lee las observaciones efectuadas por *<Supervisor de ACS>* y apoya la implantación de las recomendaciones.

<Gerente de ACS> controla que las recomendaciones hayan sido implantadas.

5. Documentación relacionada

<Entradas>

1. Documentos u Ordenes de Cambios para las Versiones.
2. Procedimiento de Cambios para las Versiones
3. Procedimiento de Promoción.
4. Análisis y evaluación de riesgos.
5. Plan de Revisión de ACS
6. Lista de Comprobación

<Entregables>

1. Análisis y evaluación de riesgos
2. Informe de la revisión
3. Reporte de Anomalías
4. Plan de Revisión de ACS (Actualizado)

6. Lista de Comprobación

Responda a cada una de las preguntas y anote el puntaje correspondiente de acuerdo a lo observado y verificado. Luego, obtenga el Puntaje Total sumando los puntajes asignados, dividir el Puntaje Total entre el número de preguntas, redondee el valor obtenido. Este valor compárelo con la tabla de valores del puntaje y conocerá la calificación de la fase revisada o inspeccionada.

Valores aplicados al Puntaje:

1= No tiene, no existe 2= Malo 3= Regular

4= Bueno 5= Excelente, Si tiene

Actividades (Preguntas)	**Puntaje** (1-5)
1. ¿Los desarrolladores tienen acceso a las herramientas de versiones? 2. ¿Los elementos de la Configuración pueden ser registrados en mas de una versión? 3. ¿Es posible recuperar versiones anteriores?	

4. ¿Es utilizada la herramienta de Control de Versiones? 5. ¿Permite el Control de Desarrollo paralelo o simultaneo? 6. ¿Es posible notificar a los programadores implicados, conciliar los cambios?	

7. Análisis de Medición

Aplica las siguientes métricas, de acuerdo a las indicaciones dadas en el Anexo B:

- Indice de Defectos
- Densidad de Defectos
- Estadística de Errores por Fase
- Distribución del Error
- Horas Hombre por principales Defectos detectados

8. Informe de Revisión

INFORME DE REVISION
<Destinatario>
<Fecha>

CONTENIDO

1. TERMINOS DE REFERENCIA
< > Información general sobre actividades actuales en el proceso de cambio de versiones.
2. PROCEDIMIENTO APLICADO
< > Procedimiento aplicado en la revisión de cambios de versión.
3. RESULTADOS OBTENIDOS
< > Riesgos detectados en la Revisión de cambios de versión, lista de comprobación y observaciones.
4. RECOMENDACIONES
< > Recomendaciones para atenuar o eliminar los riesgos en el proceso de cambios de versión.
5. PERSONAL PARTICIPANTE
< > Personal que ha participado para llevar a cabo la revisión.
6. RECURSOS EMPLEADOS
< > Registra los recursos humanos (horas/hombre), materiales, servicios y gastos empleados.

Procedimiento
6.8 Revisión del Control de Releases

La Revisión del Control de Releases se ejecuta para evaluar el control existente en prevenir la distribución de software al Cliente o Usuario con funciones erróneas y verificando que el código fuente corresponda al software distribuido.
En este formato se deberá registrar información de las personas que aprobaron el procedimiento, cambios que se han efectuado y los departamentos en los cuales debe ser aplicado.

Aprobado:		
cargo		Nombre
Aprobado:		
cargo		Nombre
Aprobado:		
cargo		Nombre
Aprobado:		
cargo		Nombre
Aprobado:		
cargo		Nombre

Cambios

Revisión	Fecha	Persona Responsable	Descripción del Cambio
		Nombre	Revisión Inicial

Lista de Distribución

(lista de los departamentos que reciben copia del procedimiento)

1. Propósito

- Determinar las reglas para que las correcciones o "parches" puedan ser incorporados o liberados en una copia del software.
- Determinar un criterio de tipos o clases de releases dependiendo de la frecuencia y/o impacto en las operaciones del cliente y la habilidad de implementar los cambios en cualquier punto de tiempo.
- Determinar que el comprador será advertido de futuros cambios.
- Determinar que los cambios implementados no producirán otros problemas.
- Registrar los cambios que han sido implementados y en que locaciones y sitios.

2. Alcance

Este procedimiento es utilizado por el <*Equipo de ACS*> y se utilizara como base de las actividades de mantenimiento del software.

3. Responsabilidades

<*Analista de ACS*> Verifica las políticas de gestión de liberación de releases del software

<*Soporte Técnico*> Encargado de la operación y mantenimiento del software liberados a los clientes desde las librerías de producción.

<*Gerente Proyecto*> Encargado del manejo del proyecto de software.

<*Gerente de ACS*> Responsable de las políticas de la gestión de releases.

4. Procedimiento

4.1 Verifica la Naturaleza del Cambio

<Analista de ACS> verifica los requerimientos de cambios al producto final por parte de los clientes y/o usuarios.

<Analista de ACS> verifica la validez de la solicitud e informa a *<Jefe de Proyecto>* y *<Gerencia de ACS>* sobre los nuevos requerimientos.

4.2 Analiza el Cambio

<Analista de ACS> coordina la frecuencia y el calendario de implementación de los nuevos cambios.

<Analista de ACS> verifica el impacto de las solicitudes de cambios al producto final.

<Analista de ACS> verifica que la nueva funcionabilidad haya sido realizados por el *<Equipo de Proyecto>*.

4.3 Controla el cumplimiento del Cambio

<Analista de ACS> coordina las pruebas para verificar el cumplimiento de los requerimientos de acuerdo a los criterios de aceptación definidos previamente.

<Analista de ACS> verifica la aceptación del usuario o cliente.

<Analista de ACS> informa sobre la generación del nuevo release al *<Gerente de ACS>*.

4.4 Verifica los elementos del Release

<Analista de ACS> verifica que los cambios efectuados a los elementos del software se hayan incorporado en el nuevo release.

<Analista de ACS> verifica que el código fuente corresponda al release liberado.

4.5 Informa sobre la revisión

<Analista o Supervisor de ACS> verifica y evalúa la Revisión de la Gestión de la Configuración y dirige el informe al *<Gerente del Proyecto>* con copia al *<Gerente de ACS>*.

<Gerente de Proyecto> lee las observaciones efectuadas por *<Supervisor de ACS>* y apoya la implantación de las recomendaciones.

<Gerente de ACS> controla que las recomendaciones hayan sido implantadas.

4.6 Registra los resultados obtenidos

<Analista de ACS> registra la información en *<formato de cambios>*.

5. Documentación relacionada

<Entradas>

1. Documentos u Ordenes de Cambios para los Releases.
2. Procedimiento de Cambios para los Releases
3. Análisis y evaluación de riesgos.
4. Plan de Revisión de ACS
5. Lista de Comprobación

<Entregables>

1. Análisis y evaluación de riesgos
2. Informe de la revisión
3. Reporte de Anomalías
4. Plan de Revisión de ACS (Actualizado)

6. Lista de Comprobación

Responda a cada una de las preguntas y anote el puntaje correspondiente de acuerdo a lo observado y verificado. Luego, obtenga el Puntaje Total sumando los puntajes asignados, dividir el Puntaje Total entre el número de preguntas, redondee el valor obtenido. Este valor compárelo con la tabla de valores del puntaje y conocerá la calificación de la fase revisada o inspeccionada.

Valores aplicados al Puntaje:

1= No tiene, no existe 2= Malo 3= Regular

4= Bueno 5= Excelente, Si tiene

Actividades (Preguntas)	**Puntaje** (1-5)
1. ¿Los desarrolladores tienen acceso a las herramientas de releases? 2. ¿Los elementos de la Configuración pueden ser registrados en mas de un releases? 3. ¿Es posible recuperar versiones anteriores? 4. ¿Es utilizada la herramienta de Control de Releases? 5. ¿Permite el Control de Desarrollo paralelo o simultaneo? 6. ¿Es posible notificar a los programadores implicados, conciliar los cambios?	

7. Análisis de Medición

Aplica las siguientes métricas, de acuerdo a las indicaciones dadas en el Anexo B:

- Indice de Defectos
- Densidad de Defectos
- Estadística de Errores por Fase
- Distribución del Error
- Horas Hombre por principales Defectos detectados
- Indice de la Madurez del Software

8. Informe de Revisión

INFORME DE REVISION
<Destinatario>
<Fecha>

CONTENIDO

1. TERMINOS DE REFERENCIA
< > Información general sobre actividades actuales en el proceso de cambio de release.
2. PROCEDIMIENTO APLICADO
< > Procedimiento aplicado en la revisión del proceso de cambio de release.
3. RESULTADOS OBTENIDOS
< > Riesgos detectados en el proceso de cambio de release, lista de comprobación y observaciones.
4. RECOMENDACIONES
< > Recomendaciones para atenuar o eliminar los riesgos en el proceso de cambios de release.
5. PERSONAL PARTICIPANTE
< > Personal que ha participado para llevar a cabo la revisión.
6. RECURSOS EMPLEADOS
< > Registra los recursos humanos (horas/hombre), materiales, servicios y gastos empleados.

Capitulo 7

EVALUACION FINAL

Las metas formuladas en Plan para el Aseguramiento de la Calidad en el Desarrollo de Software son las siguientes:

Asegurar la Calidad del Software
Mejoramiento Continuo del Proceso de Desarrollo de Software.

Nuestro proceso de evaluación y verificación lo dividimos en dos etapas: el periodo de evaluación, que comprende las fases del análisis de requerimientos, Diseño, construcción y pruebas previas a la distribución e instalación en el cliente y el periodo Post- Evaluación, el cual comprende la etapa de evaluación posterior a la distribución e instalación en el cliente.

Para este fin el período de "post- evaluación" tiene una duración de un año para grandes proyectos, 6 meses para proyectos medianos y 3 meses para proyectos pequeños. Tomándose en cuenta que es el tiempo a partir del cual el producto ha sido instalado y puesto en producción.

7.1 Calidad del Software

Lo primero es calcular los Indices de Defectos por Fases del Proyecto de acuerdo con los datos obtenidos del Registro de Calidad; luego, tabularlos como se muestra en el cuadro siguiente (los cálculos se realizan de acuerdo al Anexo B).

Fase	Indice de defectos Durante la Evaluación	Indice de defectos Durante la Post- Evaluación
Requerimientos	PI_1	PI_1
Diseño	PI_2	PI_2
Construcción	PI_3	PI_3
Pruebas Unitarias	PI_4	PI_4
Pruebas Sistemas	PI_5	PI_5
Pruebas Integrales	PI_6	PI_6
Pruebas aceptación	PI_7	PI_7

Se determina un Indice de Defectos para los periodos de evaluación y Post-Evaluación:

$$DI_{eva} = \sum (i * PI_i) / PS_{eva}$$
$$= (PI_1 + 2PI_2 + ... + 7PI_7) / PS_{eva}$$

$$DI_{post} = \sum (i * PI_i) / PS_{post}$$
$$= (PI_1 + 2PI_2 + ... + 7PI_7) / PS_{post}$$

Donde: PS_{eva} y PS_{post}, es el tamaño del producto ejecutable dado en Kb.

Interpretación: El Indice PS_{eva} nos dará una idea de cuán efectivo fue el proceso de desarrollo del software y el Indice PS_{post}, nos dará una idea de cuán correcto es el producto del software y por tanto, la corrección del producto.

Una vez efectuados los cálculos anteriores, determinamos el **Indice de Calidad del Software QZ**, de acuerdo a la siguiente fórmula:

$$QZ = 1 - (DI_{post} / DI_{eva})$$

Interpretación: De a los valores que presente nuestro índice podemos afirmar lo siguiente:

Si el valor de QZ = 1, esto significa que el software esta libre defectos y que es altamente fiable, correcto y completo.

Si el valor de QZ = 0, puede ser que los defectos detectados durante el periodo de evaluación no hayan sido corregidos.

Si el valor de QZ < 0, el proceso de detección de errores en el periodo de evaluación no se ha realizado en forma correcta, siendo el software poco fiable.

Si el valor de QZ se hace indeterminado quiere decir que el producto estuvo libre de errores desde su periodo de evaluación o se realizo un al proceso de detección de errores durante este periodo.

7.2 Mejoramiento del Proceso de Desarrollo

Para obtener una evaluación del mejoramiento del Proceso de Desarrollo no solo se deberá tener en cuenta estimaciones de la performance; sino también, conocer como mejora el proceso en cuanto a las innovaciones que se incorporan al proceso para obtener mejores productos, para esto será necesario entrenar al personal, investigar y adquirir herramientas que usen tecnología punta.

Para calcular como mejoran los procesos según nuestro criterio, verifica la lista de comprobación que aparece en cada una de nuestras revisiones. Para esto se suma el número de preguntas y se multiplica por 5 (el puntaje máximo por pregunta) **Puntaje Máximo**; luego, suma el **Puntaje Obtenido** de acuerdo a la suma de puntos obtenidos al responder cada pregunta y procede a realizar la siguiente división:

$$\text{Indice de Mejoramiento } \mathbf{QZ} = \frac{\text{Puntaje Obtenido}}{\text{Puntaje Máximo}}$$

Indice de Performance del Proceso

$$\text{Exactitud del Estimación de la Planificación} = \frac{\text{Duración del Proyecto Desarrollo Software}}{\text{Duración del Proyecto Desarrollo Estimado}}$$

$$\text{Productividad Desarrollo} = \frac{\text{Líneas de Código del Proyecto}}{\text{Número Personas - mes}}$$

Capitulo 8

COSTOS DE LA CALIDAD

Para hallar el costo de elaborar un Plan de Aseguramiento de la Calidad, lo primero que tenemos que hacer es una descomposición en actividades que se llevaran a cabo en nuestro Plan, para esto nosotros proponemos la descomposición que presentamos a continuación: (Esta descomposición de actividades y sus tiempos puede variar de acuerdo con los criterios del evaluador o inspector).

ACTIVIDADES DEL PLAN DE ASEGURAMIENTO DE LA CALIDAD

(Tiempos y Recursos asignados a un Plan de Actividades de un Proyecto de ACS de nivel Básico)

Estudio Preliminar

▪ Identifica los Objetivos	1 Gerente	04h
▪ Diseño del Plan de ACS	1 Gerente	16h
	1 Supervisor	16h
▪ Cronograma de actividades	1 Supervisor	08h
▪ Organización del Equipo de ACS	1 Supervisor	04h
Total Horas		**48h**

PLAN DE CALIDAD

Revisión de la Organización

▪ Supervisión y Control	1 Gerente	02h
▪ Recopila información y documentos relacionados al negocio	1 Analista	16h
▪ Identificación de los objetivos y misión del negocio	1 Supervisor	04h
▪ Revisa la Información y documentos obtenidos	1 Supervisor	16h
▪ Aplica lista de comprobación	1 Analista	08h
▪ Evalúa y mide los riesgos	1 Supervisor	08h
▪ Informa sobre la revisión	1 Supervisor	08h
▪ Registra los resultados obtenidos	1 Analista	02h
Total Horas		**64h**

Revisión de la Planificación del Proyecto

▪ Supervisión y Control	1 Gerente	02h
▪ Identifica los Objetivos del Proyecto	1 Supervisor	04h
▪ Recopila información y documentos relacionados al Proyecto	1 Analista	08h
▪ Revisa la Información y documentos	1 Supervisor	08h
▪ Aplica lista de comprobación	1 Analista	02h
▪ Verifica los Criterios de Aceptación del Producto Final	1 Supervisor	08h
▪ Evalúa y mide los riesgos	1 Supervisor	08h
▪ Informa sobre la revisión	1 Supervisor	04h
▪ Registra los resultados obtenidos	1 Analista	02h
Total Horas		**46h**

Revisión del Contrato

▪ Supervisión y Control	1 Gerente	08h
▪ Recopila información y documentos	1 Supervisor	24h

relacionados al Contrato		
▪ Revisa y verifica el contenido del Contrato	1 Supervisor	08h
▪ Aplica lista de comprobación	1 Supervisor	04h
▪ Evalúa y mide los riesgos.	1 Gerente	02h
	1 Supervisor	06h
▪ Informa sobre la revisión	1 Supervisor	04h
▪ Registra los resultados obtenidos	1 Supervisor	02h
Total Horas		**58h**

Revisión de Estándares

▪ Supervisión y Control	1 Gerente	02h
▪ Verifica la existencia de Estándares	1 Supervisor	02h
▪ Revisa y evalúa los Estándares	1 Supervisor	08h
▪ Aplica lista de comprobación	1 Analista	08h
▪ Evalúa y mide los riesgos	1 Supervisor	08h
▪ Registra los resultados obtenidos	1 Analista	02h
▪ Informa sobre la revisión	1 Supervisor	04h
Total Horas		**34h**

Análisis de Requerimientos

▪ Supervisión y Control	1 Gerente	02h
▪ Verifica las fuentes del Requerimiento	1 Supervisor	04h
▪ Verifica la naturaleza del Requerimiento	1 Supervisor	04h
▪ Verifica las características del Requerimiento	1 Analista	24h
▪ Verifica como fue preparado el Requerimiento	1 Analista	02h
▪ Verifica el uso de Prototipos	1 Analista	08h
▪ Verifica los Diagramas de Flujo de Datos	1 Analista	04h x *#Diagrama*
▪ Verifica Diccionario de Datos	1 Analista	04h x *#Módulo*
▪ Verifica el Diagrama de Proceso	1 Analista	04h x *#Diagrama*
▪ Verifica el análisis de interfaces	1 Analista	04h x *#Interfaces*
▪ Verifica si el Diseño esta incluido en el Análisis de Requerimientos	1 Analista	04h

▪ Verifica si los requerimientos del negocio están incluidos en la Especificación del Requerimiento	1 Supervisor	04h
▪ Determina los Criterios de Validación	1 Supervisor	16h
▪ Evalúa y mide los riesgos	1 Supervisor	08h
▪ Registra los resultados obtenidos	1 Analista	02h
▪ Informa sobre la revisión	1 Supervisor	08h
Total Horas		**102h**

Revisión del Diseño

▪ Supervisión y Control	1 Gerente	02h
▪ Verifica el Método de Diseño	1 Analista	02h
▪ Los Requerimientos integrados al Diseño	1 Supervisor	04h
▪ Evalúa las características del Diseño	1 Analista	24h
▪ Verifica el Diagrama de Flujo de Datos	1 Analista	04h x *#Diagrama*
▪ Verifica el Diagrama de Transición de Estados	1 Analista	04h x *#Diagrama*
▪ Verifica el Diagrama de Estructuras	1 Analista	04h x *#Diagrama*
▪ Verifica el Diagrama de Entidad - Relación	1 Analista	04h x *#Diagrama*
▪ Verifica el Diseño de Interfaces	1 Analista	04h x *#Interfaces*
▪ Verifica el Diseño de Vistas	1 Analista	02h x *#Vista*
▪ Verifica el Diseño de Formatos	1 Analista	02h x #Formato
▪ Evalúa la Configuración	1 Analista	08h
▪ Verifica la Documentación del Diseño	1 Analista	08h
▪ Verifica los resultados de la revisión	1 Supervisor	08h
▪ Revisión Final	1 Supervisor	08h
▪ Reevalua el Diseño	1 Supervisor	08h
▪ Revisión de Control	1 Supervisor	08h
▪ Determina los Criterios de Validación	1 Supervisor	08h
Total Horas		**112h**

Inspección del Código

▪ Identifica los elementos del Software	1 Supervisor	04h

▪ Evalúa el uso de Estándares	1 Analista	02h
▪ Verifica que el Código corresponda y cumpla con los Requerimientos del Diseño	1 Analista	04 x *#Programa*
▪ Análisis Estático	1 Analista	02hx*#Programa*
▪ Análisis la Estructura de Control	1 Analista	02h x *#Programa*
▪ Análisis de Interfaces	1 Analista	02h x *#Programa*
▪ Documentación del Programa	1 Analista	02h x *#Programa*
▪ Registra los resultados obtenidos	1 Analista	02h
▪ Informa sobre la Inspección	1 Supervisor	08h
Total Horas		**28h**

PLAN DE PRUEBAS

Administración de Pruebas

▪ Supervisión y Control	1 Gerente	02h
▪ Diseña las Pruebas de Evaluación	1 Supervisor	16h
▪ Prepara el Ambiente de Pruebas	1 Supervisor	08h
▪ Reporte y Resolución de anomalías.	1 Supervisor	08h
▪ Política de Cambio de tareas	1 Supervisor	08h
▪ Política de Desviaciones	1 Supervisor	08h
▪ Registro y Seguridad	1 Supervisor	08h
Total Horas		**58h**

Casos de Prueba

▪ Diseña casos de Pruebas Unitarias	1 Supervisor	08h
▪ Diseña casos de Pruebas de Integración	1 Supervisor	08h
▪ Diseña casos de Prueba del Sistema	1 Supervisor	08h
▪ Diseña casos de Prueba de Aceptación	1 Supervisor	08h
Total Horas		**32h**

Pruebas Unitarias

▪ Planifica las Pruebas	1 Supervisor	08h
▪ Prueba de Interfaz	1 Analista	08h x *#Módulos*

▪ Prueba Estructura de Datos	1 Analista	08h x *#Módulos*
▪ Prueba de Condiciones Limites	1 Analista	08h x *#Módulos*
▪ Prueba de Caminos independientes	1 Analista	08h x *#Módulos*
▪ Reporte de Pruebas	1 Analista	04h
▪ Registra los resultados obtenidos	1 Analista	04h
▪ Informa sobre las pruebas	1 Supervisor	04h
Total Horas		**52h**

Pruebas de Integración

▪ Planifica las Pruebas	1 Supervisor	08h
▪ Prueba de Integración descendente.	1 Analista	08h x *#Módulos*
▪ Prueba de Integración ascendente.	1 Analista	08h x *#Módulos*
▪ Prueba de Regresión.	1 Analista	08h x *#Módulos*
▪ Reporte de Pruebas	1 Analista	04h
▪ Registra los resultados obtenidos	1 Analista	04h
▪ Informa sobre las pruebas	1 Supervisor	04h
Total Horas		**44h**

Pruebas de Sistemas

▪ Supervisión y Control	1 Gerente	02h
▪ Planifica las Pruebas	1 Supervisor	08h
▪ Prueba de Recuperación	1 Analista	08h x *#Módulos*
▪ Prueba de Seguridad	1 Analista	08h x *#Módulos*
▪ Prueba de Resistencia	1 Analista	08h x *#Módulos*
▪ Prueba de Rendimiento	1 Analista	08h x *#Módulos*
▪ Reporte de Pruebas	1 Analista	04h
▪ Analiza los resultados obtenidos	1 Supervisor	04h
▪ Informa sobre las pruebas	1 Supervisor	04h
Total Horas		**54h**

Pruebas de Aceptación

▪ Supervisión y Control	1 Gerente	02h

▪ Criterios de Aceptación	1 Supervisor	08h
▪ Planificación de las Pruebas	1 Supervisor	08h
▪ Analiza y Documenta los Resultados	1 Supervisor	04h
▪ Reporte de Pruebas	1 Analista	04h
▪ Logra la Aceptación del Cliente	1 Supervisor	02h
▪ Registra los resultados obtenidos	1 Analista	02h
▪ Informa sobre las pruebas	1 Supervisor	04h
Total Horas		**34h**

PLAN DE CONFIGURACION

Revisión de la Configuración

▪ Supervisión y Control	1 Gerente	02h
▪ Identificación de los elementos de la Configuración	1 Supervisor	04h
▪ Instalación del Ambiente de Configuración	1 Analista	08h
▪ Verifica la Herramienta de Gestión de la Configuración	1 Analista	08h
▪ Aplica lista de comprobación	1 Analista	04h
▪ Evalúa y mide los riesgos	1 Supervisor	04h
▪ Registra los resultados obtenidos	1 Analista	04h
▪ Informa sobre la revisión	1 Supervisor	04h
Total Horas		**38h**

Revisión del Control de Cambios

▪ Revisa y analiza la solicitud del cambio	1 Analista	08h
▪ Identifica y verifica los Cambios efectuados	1 Analista	04h
▪ Garantiza los Cambios	1 Analista	04h
▪ Audita los Cambios	1 Analista	04h
▪ Controla el Pase a Producción	1 Analista	05h
▪ Aplica lista de comprobación	1 Analista	04h
▪ Evalúa y mide los riesgos	1 Supervisor	04h

▪ Informa sobre la revisión	1 Supervisor	04h
▪ Registra los resultados obtenidos	1 Analista	02h
Total Horas		**39h**

Control de Versiones

▪ Verifica las facilidades de la Herramienta	1 Analista	08h
▪ Verifica el uso de la Herramienta	1 Analista	08h
▪ Informa sobre la revisión	1 Supervisor	04h
Total Horas		**20h**

Control de Release

▪ Verifica la Naturaleza del Cambio	1 Analista	04h
▪ Analiza el Cambio	1 Analista	04h
▪ Controla el cumplimiento del Cambio	1 Analista	04h
▪ Verifica los elementos del Release	1 Analista	04h
▪ Informa sobre la revisión	1 Supervisor	04h
▪ Registra los resultados obtenidos	1 Analista	02h
Total Horas		**22h**
TOTAL GENERAL (Sin aplicar parámetros iniciales)		**885h**

Para llevar a cabo nuestros cálculos tenemos aplicaremos los siguientes parámetros, cuyo valores están dados por nuestra experiencia y deberán ser ajustados conforme se vayan obteniendo estadísticas que nos permitan obtener una mejor aproximación de estos:

PARAMETROS:

Del Proyecto

Cronograma de Actividades del Proyecto de Desarrollo conocido.	P		
#Personal de ACS asignado al proyecto			
Nivel de Complejidad del Proyecto	Alto = 5	Medio = 3	Bajo = 1
Nivel de Calidad en la Organización	Alto = 2	Medio = 4	Bajo = 6

Costo del Personal	Gerente	= US$ 80.00 x hora
(varia de acuerdo a la Organización)	Supervisor	= US$ 50.00 x hora
	Analista	= US$ 30.00 x hora

Del Análisis de Requerimientos

#Diagramas : Cantidad de Diagramas estimados

#Módulos : Cantidad de Diagramas estimados

#Interfaces : Cantidad de Diagramas estimados

Del Diseño

#Diagramas : Cantidad de Diagramas estimados

#Interfaces : Cantidad de Diagramas estimados

#Vistas : Cantidad de Diagramas estimados

#Formatos : Cantidad de Diagramas estimados

De la Construcción

 # Programa : Cantidad de Diagramas estimados (con 100 líneas de código en promedio por elemento de software)

De las Pruebas

 # Módulos : Cantidad de Diagramas estimados

Para calcular los estimados de los *#Diagramas*, *#Interfaces*, *#Vistas*, *#Formatos*, *# Programa*, *# Módulos*, se puede utilizar el siguiente método:

1. Elabora un cuadro teniendo como filas las tareas de revisión o inspección de diagramas, programas, formatos, módulos. Como columnas los tiempos óptimos, probables y pesimistas.
2. Aplicar a cada una de las filas la siguiente fórmula:

$$T = \frac{t_{óptimo} + 4 \times t_{probable} + t_{pesimista}}{6}$$

Una vez obtenidos estos tiempos aplicarlo a cada una de nuestras actividades para obtener la cantidad de horas que se insumira en cada una de nuestras actividades como sigue:

1. Elabora un cuadro teniendo como filas las funciones del personal de ACS (gerente, supervisor, analista y un total) como columnas el tiempo insumido, el costo, total (tiempo costo).
2. Trasladar los valores obtenidos de nuestro cuadro de actividades a la columna de tiempo insumido y el costo de los parámetros asumidos por nosotros.
3. Multiplicar las columnas tiempo insumido por el costo del personal y se obtendrá el costo por cada uno de ellos.
4. Totalizar las columnas de **tiempo Insumido** y **Costo Total**.

	Tiempo Insumido	Costo Personal	Costo Total
Gerente	TI_1	80	TI_1 x 80
Supervisor	TI_2	50	TI_2 x 50
Analista	TI_3	30	TI_3 x 30
Total	**Tiempo Σ Insumido Calculado**		**Costo Total Σ Calculado**

Para obtener el esfuerzo que debe existir, se debe tomar en cuenta el flujo de actividades presentado por nosotros para conocer las actividades que se puede llevar a cabo en forma paralela y teniendo en cuenta los plazos proporcionados por en el cronograma de actividades del Equipo de Proyecto se puede obtener el número de Analistas o Supervisores necesarios para llevar a cabo el Plan de Aseguramiento en el Desarrollo de Software. Para obtener este valor se deberá realizar el siguiente cálculo:

$$E = \text{Redondea } \frac{\text{(tiempo del personal de ACS en la actividad)}}{\text{tiempo del personal del proyecto en la actividad}}$$

Considera que los equipos de proyectos y ACS deberán acabar a la vez cada una de las actividades (Análisis de requerimiento, Diseño, Construcción, Pruebas).

El Tiempo Insumido debe ser ajustado por un factor según la cantidad de personas que se vayan incluyendo en el Proyecto de Aseguramiento (debido a que el tiempo asignado es para una persona). Este factor es aplicado debido a que el personal requiere un tiempo adicional para fines de comunicación y no es simplemente dividir el trabajo y se reducirá según el número de personas asignadas al Proyecto de Aseguramiento. Nosotros proponemos el uso de los siguientes factores:

# Personas	Factor
2	.98
3	.96
4	.93
5	.90
6	.86
7	.82
Más de 7	.75

Se debe considerar que cada Equipo formado (en este no deberá incluirse la labor del Gerente ACS) deberá estar integrado por 5 ó 6 personas como un número conveniente. El tiempo insumido podrá calcularse de la siguiente manera:

$$\textbf{Tiempo Insumido Estimado} = \frac{\text{Tiempo Insumido Calculado}}{\text{factor}}$$

Al Final, del proyecto se debe efectuar un seguimiento de los costos y esfuerzo involucrados; para tal fin, en cada uno de los informes de revisión o inspección se registra el personal y tiempo que se necesito para llevar a cabo dicha actividad. Al totalizar todos estos valores nos permitirá obtener el esfuerzo y tiempo real involucrados en el proyecto, con lo cual en otras revisiones podremos efectuar una mejor estimación de dichos valores.

Capitulo 9

CONCLUSIONES

Después de haber culminado con la investigación llegamos a las siguientes conclusiones:

1. En las organizaciones se hace necesario tener una cultura de calidad, ya que todo Sistema de Calidad generará un ambiente de calidad y control del negocio, en el que todo sus miembros se sentirán motivados a sumarse al esfuerzo de lograr el mejoramiento continuo y satisfacción del cliente interno y externo. Esto permitirá que todo proyecto pueda ser ejecutado con la seguridad de obtener éxito y lograr los objetivos por los cuales fue desarrollado, alcanzando de esta manera las metas del negocio.

2. En las organizaciones, se hace necesario aplicar el control sobre los procesos. Teniendo en cuenta los principios estadísticos se logrará mejorar la ejecución, capacidad y estabilidad de los procesos, lo que permita predecir las características de sus productos y/o servicios, costos y cronogramas, mejorando la efectividad, eficacia, tecnología y rentabilidad.

3. Las organizaciones gracias al control que incorporen a sus procesos les permitirá ser capaces de asegurar la Calidad del Software y mejorar el Proceso de

Desarrollo de Software. Teniendo en cuenta que para asegurar el mejoramiento continuo del Proceso de Desarrollo de Software, tendrán que invertir en entrenamiento al personal, investigación y adquisición de nuevas herramientas.

4. En las actividades de Desarrollo de Software el Equipo de Aseguramiento de Calidad del Software, deberá participar activamente evaluando el trabajo del Equipo de Desarrollo, sin esperar a que este último culmine su labor para iniciar sus actividades.

5. Para el Desarrollo de Software es necesario el uso de herramientas CASE; pues, se tiene un mejor control del desarrollo de software, mejorando la estandarización y la facilidad para detectar omisiones e inconsistencias.

6. La aplicación de métricas permitirán conocer los niveles de calidad y productividad los cuales ayuden a tomar decisiones y acciones correctivas en forma anticipada. Con lo cual se puede lograr el mejoramiento continuo del Proceso de Desarrollo de Software.

7. El software como producto debe tener un esquema de medición de la calidad basada en la detección de defectos, siendo la forma más directa y concreta de hacer la medición, permitiendo obtener el nivel de confianza con respecto al software.

8. La estimación de futuros proyectos de Aseguramiento de la Calidad debe estar basado en estadísticas o información histórica de proyectos anteriores de similares características. Para lo cual, el Equipo de Aseguramiento de la Calidad del Software debe registrar información de la experiencia obtenida en sus actividades, tales como: mediciones efectuadas al producto y al proceso, debiendo estar disponibles para futuros proyectos o procesos en los cuales se desee aplicar calidad y mejorar de manera continua el Proceso de Desarrollo de Software o efectuar comparaciones entre proyectos similares.

9. El Plan de Aseguramiento de la Calidad servirá para enfrentar un proceso de certificación ISO-9000 para la Calidad del Proceso de Desarrollo de Software.

Capitulo 10

RECOMENDACIONES

1. La Alta Gerencia deberá estar involucrada en la aplicación de la filosofía de Calidad en toda la organización, para que todo plan que tenga relación con el Aseguramiento de la Calidad pueda llevarse a cabo.

2. El Equipo de Aseguramiento de Calidad del Software deberá tener un mayor nivel jerárquico que el Equipo de Desarrollo del Proyecto.

3. Los riesgos del proyecto de Desarrollo de software deberán ser definidos estableciendo su probabilidad de aparición e impacto, sugerimos se clasifiquen y señalen sus medidas de control para futuros proyectos.

4. El Equipo que ejecute el Plan de Aseguramiento de la Calidad del Software deberá estar conformado por personas que tengan un alto nivel de conocimiento del negocio, amplia experiencia en la Planificación, Desarrollo de Software, Ejecución de Pruebas, uso de Estándares, conocimiento de tecnologías y aplicación de metodologías.

5. Las organizaciones deberán implementar las métricas que vayan de acuerdo a sus propias necesidades de medición, perspectivas e intereses en términos de atributos de los productos, procesos y recursos involucrados.

6. Para asegurar una buena gestión en la implementación de métricas recomendamos el uso de herramientas automáticas, gestión de reporte de problemas y acciones correctivas, que faciliten la recolección de datos y ejecución de métricas. Herramientas de prueba que permitan probar el software para disminuir el tiempo y aumentar la capacidad de detección de defectos. Asimismo, se hace necesario el uso de herramientas que faciliten la gestión de configuración del software.

7. La documentación es una actividad critica que debe ser ejecutada en cualquier Plan de Aseguramiento de la Calidad, sugerimos el uso de formatos de acuerdo a la necesidad de información de los Equipos de Desarrollo y de Aseguramiento de la Calidad del Software, los cuales deberán estar diseñados de acuerdo a estándares establecidos por la organización, en cuanto a su uso, contenido y forma.

Capitulo 11

BIBLIOGRAFIA

- [1] Marvin V. Zelkowitz — Advances in Computer Volumen 42. United States, Edited by Academic Press, 1996.
- [2] LEXUS Editores — Diccionario LEXUS. Barcelona - España, Editorial Trebol, 1997.
- [3] Roger S. Pressman — Ingeniería del Software: Un enfoque práctico. España, McGraw - Hill, 1998.
- [4] M. Sc. Alberto Un Jan — Sistemas de Información Gerencial. Lima - Perú, Editado por la Univ. Católica del Perú, 1996.
- [5] Víctor Cisneros — Taller de Análisis y Diseño de Sistemas. Lima Perú, Editado por la Univ. Católica del Perú,1996.
- [6] Kan, Stephen H. — Metric and Models ın Software Quality Engineering. United State, Ed. Reading, Mass: Addison - Wesley, 1995.
- [7] G. Gordon Schulmeyer — Zero Defect Software. New York - United State, Ed. McGraw - Hill, 1990.

- [8] IEEE — Software Engineering. New York - United State, Published by the IEEE, 1994.
- [9] Guillermo A. Stoll y Senlle, Andres — Calidad Total y Normalización ISO 9000: ISO 9000 las normas para la calidad en la práctica. Barcelona - España, Ed. Gestión 2000, 1994.
- [10] Alan M. Davis and Dean A. Leffingwell — Using Requirements Management to delivery of Higher Quality Applications. 1996.
- [11] Vincent K. Omachonu y Joel Ross — Principios de Calidad Total. México, editorial Diana, 1995.
- [12] Timothy Braithwaite — The Power of IT. Maximizing Your Technology Investments. Wisconsin - United State, Editorial ASQC Quality Press Milwaukee, 1996.
- [13] Charles H. Schmauch — ISO 9000 for Software Developers. Wisconsin - United State, Ed. ASQC Quality Press Milwaukee, 1995.
- [14] A. Bernillón / O.Cérutti — Implantar y Gestionar la Calidad Total. 2da. Edición.
- [15] Paul Oman / Sharon Laurence — Applying Software Metrics. New York - United State, IEEE, 1998.
- [16] Jack Woodall, Deborath K. Rebuck y Frank Voehl — Total Quality in Information Systems and Technology. United State, Ed. St. Lucie Press, 1997.

ANEXOS

Anexo A

NORMAS DE CALIDAD

International Standard

ISO 9000-3

First edition
1991-06-01

Correct and reprinted
1993-05-01

Quality management and quality assurance standards

PART 3:
Guidelines for the application of ISO 9001 to the development, supply and maintenance of software

Normes pour la gestion de la qualité et l'assurance de la qualité - Partie 3: lgnes directrices pour l'aplication de l'ISO 9001 au développement, á la mise á disposition et á la maintenance du logiciel

---

Reference number
ISO 9000-3: 1991(E)

Foreword

ISO (the international Organization for standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committee. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

Draft International Standards adopted by the technical committees are circulated to the member bodies for voting. Publication as an International Standard requires approval by at least 75% of the member bodies casting a vote.

International Standard ISO 9000-3 was prepared by Technical Committee ISO/TC 176, *Quality management and quality assurance*.

ISO 9000 consists of the following parts, under the general title *Quality management and quality assurance standards:*

Part 1: Guidelines for selection and use
Part 2: Generic guidelines for the application of ISO 9001, ISO 9002 and ISO 9003
Part 3: Guidelines for the application of ISO 9001 to the development, supply and maintenance of software

Part 1 will be a revision of ISO 9000:1987, Part 2 is to be published.

Introduction

With the progress of information technology, the amount of software products has been increasing and the quality management of software products is essential. One of the means of establishing a quality management system is to provide guidance for software quality assurance.

The requirements for a generic quality system for two-party contractual situations have already been published: ISO 9001:1987, *Quality systems - Model for quality assurance in design/development, production, installation and servicing.*

However, the process of development and maintenance of software is different from that of most other types of industrial products. In such a rapidly evolving technology field it is therefore necessary to provide additional guidance for quality systems where software products are involved, taking into account the present status of this technology.

The nature of software development is such that some activities are related to particular phases of the development process, while others may apply throughout the process. These guidelines have therefore been structured to reflect these differences. This document thus does not correspond directly in format with ISO 9001 and cross-reference indexes (annex A and annex B) are provided to give assistance when referring to that standard.

Contract between two parties for software product development may occur in may variations. In certain cases of two-party contracts, these guidelines might not be applicable even if "tailored". It is therefore important to determine the adequacy of the application of this part of ISO 9000 to the contract.

This part of ISO 9000 deals primarily with situations where specific software is development as part of a contract according to purchaser's specifications. However, the concepts described may be equally of value in other situations.

NOTES

1 In English, of use of the masculine gender in this part of ISO 9000 is not meant to exclude the feminine gender where applied to persons. Similarly, use of the singular does not exclude the plural (and vice versa) when the sense allows.

2 Throughout this part of ISO 9000 where there is no further guidance, the next of the relevant ISO 9001 clause is given and printed in italics.

3 In this part of ISO 9000 there are a number of lists; none of these is presumed to be exhaustive - they are intended as example.

International Standard

ISO 9001

Second edition
1991-06-01

Quality systems – Model for quality assurance in design, development, production, installation and servicing

Systémes qualité - Modéle pour l'assurance de la qualité en conception, développement, production, installation et prestations associées

---

Reference number
ISO 9001: 1994(E)

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committee has been established has the right to the represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

Draft International Standards adopted by the technical committees are circulated to the member bodies for voting. Publication as an International Standard requires approval by at least 75% of the member bodies casting a vote.

International Standard ISO 9001 was prepared by Technical Committee ISO/TC 176, *Quality management and quality assurance, Subcommittee SC 2, Quality systems.*

This second edition cancels and replaces the first edition (ISO 9001:1987), which has been technically revised.

Introduction

This international Standard is one of three International Standards dealing with quality system requirements that can be used for external quality assurance purposes. The quality assurance models, set out in the three International Standards listed below, represent three distinct forms of quality system requirements suitable for the purpose of supplier demonstrating its capability, and for the assessment of the capability of a supplier by external parties.

a) ISO 9001, *Quality systems - Model for quality assurance in design, development, production, installation and servicing*

   for use when conformance to specified requirements is to be assured by the supplier during design, development, production, installation and servicing.

b) ISO 9002, *Quality systems - Model for quality assurance in production, installation and servicing*

   for use when conformance to specified requirements is to be assured by the supplier during production, installation and servicing.

c) ISO 9003, Quality systems - Model for quality assurance in final inspection and test

   for use when conformance to specified requirements is to be assured by the supplier solely at final inspection and test.

It is emphasized that the quality system requirements specified in this International Standard, ISO 9002 and ISO 9003 are complementary (not alternative) to the technical (product) specified requirements. They specified requirements which determine what elements quality systems have to encompass, but it is not the purpose of these International Standard to enforce uniformity of quality systems. They are generic and independent of any specific industry or economic sector. The design and implementation of a quality system will be influenced by the varying needs of an organization, its particular objectives, the products and services supplied, and the processes and specific practices employed.

It is intended that these International Standards will be adopted in their present form, but on occasions they may need to be tailored by adding or deleting certain quality system requirements for specific contractual situations. ISO 9000-1 provides guidance on such tailoring as well as on selection of the appropriate quality assurance model, viz. ISO 9001, ISO 9002 or ISO 9003.

Recognized as an
American National Standard (ANSI)
and

1989

IEEE 730-1989
(Revision of IEEE Std 730-1984

Redesignation of IEEE 730.1-

IEE Standard for
Software Quality Assurance Plans

Sponsor
Technical Committee on Software engineering
Of the
IEEE Computer Society

Approved August 17, 1989

IEEE Standards Board

Approved January 22, 1990

American National Standards Institute

@ Copyright 1989 by

The Institute of Electrical and electronics Engineers, Inc
345 East 47th Street, New York, NY 10017, USA

No part of this publication may be reproduced in any form,
In an electronical retrieval system or otherwise,
Without the prior written permission of the publisher.

IEEE Standards documents are developed within the Technical Committees of the IEEE Societies and the Standards Coordinating Committees of the IEEE Standards Board. Members of the committees serve voluntarily and without compensation. They are not necessarily members of the Institute. The standards developed within IEEE represent a consensus of the broad expertise on the subject within the Institute as well as those activities outside of IEEE which have expressed an interest in participating in the development of the standard.

Use of an IEEE Standard is wholly voluntary. The existence of an IEEE Standard does not imply that there no other ways to produce, test, measure, purchase, market, or provide other goods and services related to the scope of the IEEE Standard. Furthermore, the viewpoint expressed at the time a standard is approved and issued is subject to change brought about through developments in the state of the art and comments received from users of the standard. Every IEEE Standard is subjected to review at least every five years old, and has not been reaffirmed, it is reasonable to conclude that its contents, although still of some value, do not wholly reflect the present state of the art. Users are cautioned to check to determine that they have the latest edition of any IEEE Standard.

Comments for revision of IEEE Standards are welcome from any interested party, regardless of membership affiliation with IEEE. Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments. Interpretations: Occasionally questions may arise regarding the meaning of portions of standards as they relate to specific applications. When the need for interpretations is brought to the attention of IEEE, the institute will initiate action to prepare appropriate responses. Since IEEE Standards represent a consensus of all concerned interests, it is important to ensure that any interpretation has also received the concurrence of a balance of interests. For this reason IEEE and the members of its technical committees are not able to provide an instant response to interpretation requests except in those cases where the matter has previously received formal consideration.

Comments on standards and requests for interpretations should be addressed to:

Secretary, IEEE Standards Board
445 Hoes Lane
P.O. Box 1331
Piscataway, NJ 08855-1331
USA

IEE Standards documents are adopted by the Institute of Electrical and Electronics Engineers without regard to whether their adoption may involve patents on articles, materials, or processes. Such adoption does not assume any liability to any patent owner, nor does it assume any obligation whatever to parties adopting the standards documents.

Foreword

(This Foreword is not a part of IEEE Std. 730.1-1989, IEEE Standard for Software Quality Assurance Plans.)

This standard assists in the preparation and content of Software quality Assurance Plans and provides a standard against which such plans can be prepared and assessed. It is directed toward the development and maintenance of critical software-that is, where failure could impact safety or cause large financial or social losses.

The readers of this document are referred to ANSI/IEEE Std 983-1986, IEEE Guide for Software Quality Assurance Planning, for recommended approaches to good software quality assurance practices in support of this standard. While ANSI/IEEE Std 983-1986 specifically refers to ANSI/IEEE Std 730-1984, almost all of its content applies directly to this revision.

In this standard, firmware is considered to be software and is to be treated as such.

Footnotes are not part of the standard.

There are three groups to whom this standard applies: the user, the developer, and the public.

(1) The user, who may be another element of the same organization developing the software, has a need for the product. Further, the user needs the product to meet the requirements identified in the specification. The user thus cannot afford a "hands-off" attitude toward the developer and rely solely on a test to be executed at the end of the software development time period. If the product should fail, not only does the same need to obtain a reasonable degree of confidence that the product is in the process of acquiring required attributes during software development.

(2) The developer needs an established standard against which to plan and to be measured. It is unreasonable to expect a complete reorientation from project to project. Not only is it not cost effective, but, unless there exists a stable framework on which to base changes, improvement cannot be made.

(3) The public may be affected by the users' use of the product. These users include, for example, depositors at a bank or passengers using a reservation system. Users have requirements, such as legal rights, which preclude haphazard development of software. At some later date, the user and the developer may be required to show that they acted in a reasonable and prudent professional manner to ensure that required software attributes were acquired.

This standard was prepared by the Software Engineering Standards Subcommittee of the Software Engineering Technical Committee of the IEEE Computer Society. It was initially approved by the IEEE Standards Board for "trial use" in December of 1979, with a subsequent "full use" approval in September of 1981, and a revision approved on June 14, 1984.

Anexo B

METRICAS DE CALIDAD

1. Métricas de MC-CALL

Define un conjunto de factores de calidad que evalúan el software desde tres aspectos importantes: sus características operativas, su capacidad de soportar los cambios y su adaptabilidad a nuevos entornos. Son usadas para "obtener el grado" de los atributos específicos del software [3].

$$F_c = c_1 \text{ X } m_1 + m_2 + \ldots + c_n \text{ X } m_n \text{ [3]}$$

Donde:

F_c es un factor de calidad del software,

c_i son coeficientes de regresión y

m_i son las métricas que afectan al factor de calidad.

El esquema de graduación propuesto por McCall va en una escala de 0 (bajo)a 10 (alto).

McCall define los siguientes factores y métricas.

Factores / Métricas	Corrección	Fiabilidad	Eficiencia	Integridad	Facilidad de Mantenimiento	Flexibilidad	Facilidad Prueba	Portabilidad	Reusabilidad	Facilidad Inter Operación	Facilidad Uso
Facilidad de auditoría				X			X				
Exactitud		X									
Norm. De las comunicaciones										X	
Completitud	X										
Complejidad		X				X	X				
Concisión			X		X	X					
Consistencia	X	X			X	X					
Estandarización en los datos										X	
Tolerancia de errores		X									
Eficiencia en la ejecución			X								
Facilidad de expansión						X					
Generalidad						X		X	X	X	
Independencia del hardware								X	X		
Instrumentación				X	X		X				
Modularidad		X			X	X	X	X	X	X	
Facilidad de operación			X								X
Seguridad				X							
Autodocumentación					X	X	X	X	X		
Simplicidad		X			X	X	X				
Indep. del sistema de software								X	X		
Facilidad de Traza	X										
Formación											X

2. Indice de Defectos

Aplicación: Esta medida provee una continuidad, índice relativo de como obtener un software correcto durante el ciclo de desarrollo. Esta medida es bastante exacta, fase dependiente, de peso, para su cálculo se requiere poco conocimiento de matemáticas avanzadas o estadísticas. Si se aplica tempranamente esta medida durante el ciclo de la vida el usuario tendrá productos que puedan ser evaluados.

Entradas: La siguiente lista es aplicada a cada fase en el ciclo de vida:

Di	Número total de defectos detectados durante la fase iava.
Sí	Número de defectos serios encontrados.
Mi	Número de defectos medios encontrados.
PS	Tamaño del producto en la fase iava.
Ti	Número de defectos triviales encontrados.
W_1	Factor ajuste para defectos serios (Default=10)
W_2	Factor ajuste para defectos medios (Default=3)
W_3	Factor ajuste para defectos triviales (Default=1)

Implementación: La medida es generada como una suma de cálculos efectuados durante todo el desarrollo. Es una medida continúa aplicada al software procediendo desde el diseño hasta las pruebas finales.

A cada fase de desarrollo, se calcula el índice de la fase (PI_i) asociado con el número y severidad de defectos.

$$PI_i = W_1 \frac{S_i}{D_i} + W_2 \frac{M_i}{D_i} + W_3 \frac{T_i}{D_i}$$

El índice de defectos (DI) es calculado en cada fase y es acumulativo adicionando los cálculos de PI_i a través del ciclo de desarrollo.

$$DI = \sum (i * PI_i) / PS$$
$$= (PI_1 + 2PI_2 + ... + iPI_i + ...) / PS$$

Donde cada fase es el peso tal que conforme se desarrolle el software tiene que progresar, según la fase (ej. 2 ó 3, respectivamente) asignada.

Los datos coleccionados en proyectos anteriores se pueden usar en una gráfica como base de comparación.

Interpretación: La interpretación esta basada en la magnitud del Indice de Defectos: el número más pequeño es el mejor. Cada organización o proyecto debe computar números de base para cada proyectos anterior por cada tipo de software elaborado. Entonces, para cada proyecto se debe hacer un seguimiento y evaluar lo siguiente:

(1) Si la performance ha aumentado.

(2) Impacto de programas de educación para el programador.

(3) Impacto de nuevas prácticas en programación.
(4) Impacto de entrar a una área del negocio nueva o ejecutar tareas nuevas o poco familiares.
(5) Impacto de cambios a medias, requerimientos con mayores cambios.

Esta medida puede ser usada adjunta ha una evaluación del defecto o análisis de la causa del error. Cada usuario determinaría una clasificación única del defecto. Por ejemplo los defectos serios se describen como faltas que guarda el software desde su funcionamiento. Defectos medios se describen como faltas que causarán que el software opere en forma degradada. Defectos triviales serían definidos como aquellos que no afectarán la ejecución funcional.

Beneficio: Esta medida es usada como una medida al proceso y al producto. Como medida del proceso provee una indicación de si al eliminar el defecto del proceso en una fase del desarrollo se efectúo con suficiente efectividad. También permite al diseñador un medio de evaluar la efectividad de fase relacionada al proceso de eliminar el defecto.

Como medida del producto, permite en la administración del desarrollo un medio de evaluar el grado de exactitud del software en una fase dada, en consecuencia esto provee una indicación del crecimiento de la fiabilidad del producto desde las fases de desarrollo hasta su aceptación final.

Es fácil de usar e interpretar, y es sucesivamente aplicable a las fases de desarrollo. Se puede hacer manualmente o automatizado. Puede ser fácilmente extendido para incluir sub- fases, con cada sub- fase separados por prioridad. El índice es priorizado para que la mayor cantidad de defectos se hallen tempranamente y obtener mejores resultados. El uso del índice alienta al proceso de eliminación de defectos tempranamente; tal como, en las inspecciones.

Si se hacen requerimientos en un lenguaje estructural, se pueden comenzar las inspecciones al nivel de los requerimientos y este Indice Defectos (DI) se puede ajustar al comenzar la fase de requerimientos.

3. Densidad de Defectos

Aplicación : La medida puede ser usada antes del diseño e inspecciones del código, si la densidad de defectos esta fuera de lo normal después de varias inspecciones, es una indicación que el proceso de inspección requiere una mayor investigación.

Entradas: Establezca niveles de severidad para la designación del defecto

Di	Número total de defectos detectados durante el proceso de inspección, o en la iava fase del ciclo de vida
I	Numero total de inspecciones a la fecha
SIZE	En la fase de requerimientos y Diseño = Número total de requisitos En la Fase de Construcción = Número de líneas de código fuente, En la Fase de Implementacion = Número de líneas de código ejecutable y declaraciones de datos no ejecutable

Implementación: Establecer un esquema de clasificación para la severidad y clase de defectos. Por cada inspección, registre el tamaño del producto y el número total de defectos únicos. Por ejemplo, en la fase de diseño calcular el índice.

$$DD = \frac{\sum_{i=1}^{I}}{SIZE}$$

Esta medida asume que es usada para un diseño estructurado. De cualquier modo, si se usa otra metodología de diseño, entonces tiene que ser desarrollada otra unidad de densidad de defectos conforme a la metodología en que se expresa el diseño.

Interpretación: Esta medida tiene un grado de indeterminismo. Por ejemplo un valor bajo indicaría un proceso y producto buenos o indicaría un mal proceso. Si

el valor es bajo comparado a similares proyectos anteriores, el proceso de inspección deberá ser examinado. Si el proceso de inspección se halla que es el adecuado, se debe concluir entonces que el proceso de desarrollo tiene como resultado un producto relativamente libre de defectos.

Si el valor de la medida es demasiado alto comparado a similares proyectos anteriores, el proceso de desarrollo se debe examinar para determinar si una programación pobre ha originado elementos del software pobres. En tal caso, se debe hacer una lista de defectos por tipo y severidad según las categorías de defecto; tales como: diseño, lógica, normas, interfaces, comentarios, requerimientos, etc. La importancia de la densidad del defecto puede ser determinada y tomar una acción. Por ejemplo si los defectos son altos debido a los requerimientos, sería apropiado detener el desarrollo, por lo menos en una área, hasta que se corrigen los requerimientos. Si, en cambio, el proceso de desarrollo se juzga ser el adecuado, entonces el proceso de inspección esta sirviendo valiosamente como una función de pre- test.

Después de una significativa experiencia con esta medida, sería apropiado establecer valores bajos como metas para asegurar el crecimiento de la fiabilidad continua del proceso de desarrollo del software.

Beneficio: La medida es fácil computar y sirve como parámetro para evaluar la efectividad del proceso de inspección. Desviaciones del estándar le sirven para tomar atención inmediata del manejo y por eso le da una oportunidad al diseñador para hacer correcciones cualquiera que sea necesario para mantener en el producto un significativo control, fiabilidad y exactitud.

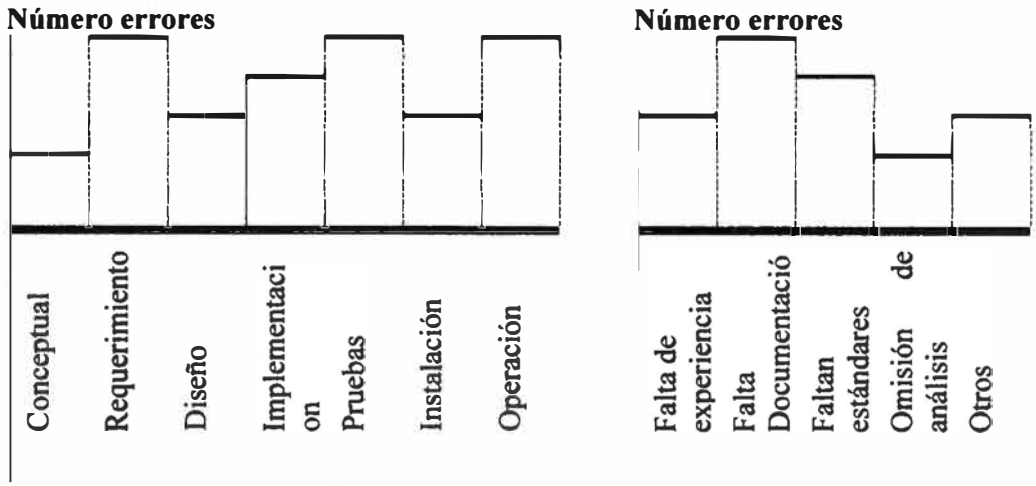
4. Estadística de Defectos por Fase

Aplicación: La búsqueda de las causas de las fallas del software y defectos involucrados en el análisis de defectos coleccionados durante cada fase del desarrollo de software. Distribución de los errores que permite ranking de los modos de falla predominante.

Entradas: La descripción de errores esta dado por los siguientes puntos:

(1) Fallas asociadas
(2) Tipos
(3) Severidad
(4) Fase i
(5) Medición preventiva
(6) Mecanismos de descubrimiento incluyendo razones para defectos no detectados a tiempo de fallas asociadas.

Implementación: Los errores son registrados y contabilizados de acuerdo a un criterio adoptado para cada clasificación. Él número de errores es graficado para cada clase de error, de tal manera, que se puedan prevenir errores recurrentes similares o descubrir anticipadamente las fallas.



(a) Errores de Distribución por Fase **(b) Errores por categoría de causa**

Interpretación: Gráficas similares se pueden analizar para el manejo de identificación de causas, riesgos y recomendar la naturaleza y prioridad de acciones correctivas para mejorar la eliminación de errores.

Esta medida es útil para recomendar que pasos para la prevención de la repetición de errores similares, eliminar las causas y descubrir fallas anticipadamente con el propósito de reducir el costo en el ciclo de la vida del software.

Beneficio: El Análisis de Errores por Distribución es un factor clave en la comprensión de las áreas donde intervienen y/o acciones correctivas son los pasos más efectivos hacia los errores y eliminación de las fallas.

5. Distribución del Error

Aplicación: Se encuentran las causas de fallas del software y fracasos involucrados en el análisis de defectos de la colección de datos, durante cada fase del desarrollo del software. La Distribución de los errores se clasifica jerárquicamente según los modos de fracasos predominantes.

Entradas: Descripción del error:

(1) fallas asociadas
(2) Tipos
(3) Severidad
(4) Fase de introducción
(5) Medida preventiva
(6) Mecanismo de descubrimiento, razones que incluyen la no detección anticipada de fallas asociadas.

Implementación: Las entradas por cada error son registradas y se cuentan según los criterios adoptados por cada clasificación. Los números de errores son graficados por cada clase. Los errores son clasificados y contados por fase, por la causa, y por la causa del descubrimiento de la falla detectada. Otra forma de clasificación similar puede ser usada; tal como: el tipo de pasos sugerido para prevenir el recurrencia de errores similares o tipo de pasos sugirió para descubrir anticipadamente las fallas correspondientes.

6. Horas Hombre por Principales Defectos Detectados

Aplicación: Los procesos de inspección del diseño y código son dos de los más efectivos procesos de eliminación de defectos disponibles. Eliminar anticipadamente los defectos de las fases de diseño e implementación, si hace efectivamente y eficazmente, significativamente mejorar la fiabilidad del desarrolló del producto y deja un mayor ambiente de control en las pruebas. Esta

medida proporciona una figura cuantitativa que puede ser usado para evaluar la eficiencia del proceso de los procesos de inspección del diseño y código.

Entradas:

T1 =	Tiempo gastado por el equipo de inspección en la preparación para las reuniones de inspección del diseño y código.
T2 =	Tiempo gastado por el equipo de inspección en la conducción de las reuniones de inspección del diseño y código.
S_i	Número de principales defectos (no triviales) detectados durante la iava inspección.
I	Número Total de inspecciones a la fecha

Implementación: En cada reunión de implementación registre el tiempo total de preparación consumido en reuniones por el equipo de inspección. También registre el tiempo total gastado en conducir la reunión de inspección. Todos los defectos son registrados y agrupados en categorías. (Un defecto principal es aquel que debe corregirse para que el producto o función este de acuerdo a lo especificado en los requerimientos).

Los tiempos de inspección son sumados y los defectos son acumulados. Se debe ejecutar el cálculo durante el diseño y la codificación. Si el diseño no está escrito en un lenguaje de diseño estructural, entonces esta medida puede ser sólo aplicado durante la fase de implementación.

El horas hombre por principales defectos descubiertos es:

$$M = \frac{\sum_{i=1}^{I} (T1 + T2)_i}{\sum_{i=1}^{I} S_i}.$$

Se debe comenzar con el cálculo después de aproximadamente 8,000 líneas de diseño detallado o se tiene que inspeccionar el código.

Interpretación: La medida es aplicada al desarrollo de nuevo código, no a código que se esta modificando, a menos que la modificación sea un cambio completo.

De experiencia empírica, la medida (M) para nuevo código caerá consecuentemente entre tres y cinco. Si se experimenta una significativa desviación de este rango, se debe investigar la materia. Si la medida es muy baja, la densidad del defecto estaría demasiado alta y debería examinarse todo el software. Si la medida es muy alta, el proceso de la inspección es probable que no sea el apropiado y, por eso, el impacto anticipado en la mejora de la fiabilidad esperada de la inspección no será realizado.

Beneficio: Esta medida es fácil de calcular. Sirve como un índice para evaluar la efectividad y eficacia del proceso de inspección. Desviaciones de la norma sirven como un indicador que requiere atención inmediata y, por eso, el diseñador debe hacer modificaciones necesarios para mantener la exactitud del software con suficiente corrección y fiabilidad durante el desarrollo.

7. Cumplimiento de los Requerimientos

Aplicación: Este análisis es usado para verificar el cumplimiento de los requerimientos usando diagramas de verificación (SVDs), una interconexión lógica de descomposición de elementos (ej. estímulos vs. respuesta) para detectar inconsistencias, incompletos y malas interpretaciones.

Entradas:

Descomposición de elementos.
Estímulos = Entrada Externa.
Función = Definido por una entrada / salida de proceso.
Respuesta = Resultado de la función.
Etiqueta = Identificador Numérico DE.
Referencia = Número del párrafo de la Especificación.

Errores del requerimiento detectados usando SVDs:

Número de inconsistencias.
Número de incompletos.
Número de mala interpretaciones.

Implementación: La implementación de un SVD esta compuesto de las siguientes fases:

La fase de descomposición es iniciada graficando las especificaciones de los requerimientos de sistemas dentro de elementos de estimulo/respuesta (DEs). Es decir, todas las palabras claves, frases, requerimientos funcionales o de performance, y salidas esperadas son documentadas en el formato de descomposición.

ESTIMULO	LABEL FUNCION	RESPUESTA
REFERENCIA DE LA ESPECIFICACION		

(2) La fase gráfica usa las DEs desde la fase de descomposición y lógicamente se conecta para formar el gráfico SVD.

(3) La fase de análisis examina el SVD desde la fase gráfica usando conectividad y matrices de confiabilidad. Los varios tipos de errores de requerimientos son determinados para experimentar el diagrama de verificación del sistema e identificar errores de inconsistencia, incompletos, mala interpretación.

Un análisis es también el porcentaje para los varios tipos de errores para la respectiva categoría: inconsistencia, incompletos, mala interpretación.

$$\text{Inconsistencias (\%)} = N_1 / (N_1 + N_2 + N_3) * 100$$
$$\text{Incompletos (\%)} = N_2 / (N_1 + N_2 + N_3) * 100$$
$$\text{Mala interpretación (\%)} = N_3 / (N_1 + N_2 + N_3) * 100$$

Este análisis puede ayudar en futuros esfuerzo en el desarrollo de software.

Interpretación: El SVDs es analizado para interpretar la conectividad y matrices del alcance. Las reglas que gobiernan su uso son:

Matriz de Conectividad: Las reglas que gobiernan su uso son las siguientes:

Regla 1: El tamaño de la matriz es N x N donde N es igual al número de elementos de la descomposición en el SVD.

Regla 2: Cualquier (i, j) elemento de la matriz es igual a, si DE_i transfiere directamente a DE_j. Es igual a cero en otros casos.

Regla 3: Si toda una columna de la matriz esta en ceros identifica un DE que no se transfiere o por otro DE y, por eso, debe ser una entrada DE.

Regla 4: Si toda una fila de la matriz esta en ceros identifica un DE que no se transfiere a otro DE y, por eso, debe ser un DE terminal.

Regla 5: Si hay un 1 a lo largo de la diagonal, indica que un DE particular esta recursividad él mismo. Aunque posiblemente, esto es improbable al nivel del requerimiento usualmente indica una falla.

Regla 6: Cualquiera DE asociado a un cero en la fila y un cero en la columna indica una posición de una función aislada del sistema. Aunque posiblemente, son raros y deben estar doblemente verificados para su autenticidad.

Matriz de Alcance: Las reglas gobiernan el uso de las matrices de alcance incluyen las reglas del 1 al 6 para la conectividad de matrices y la regla del alcance es la siguiente:

Regla 7: Cualquier (i, j) elemento de la matriz es igual a uno si se puede alargar DE_j (directamente o indirectamente) de DE_i. Es igual a poner en cero en otros casos.

Un análisis de los porcentajes ayudará en la especificación de requerimientos y futuras pruebas de la codificación. Las categorías con los porcentajes más altos deben recibir un mayor escrutinio para futuras especificaciones del requerimiento y diseño de casos de prueba.

Beneficio: SVDs es un buen dispositivo de comunicación para revisiones internas y externas. La matriz de alcance asociada con el SVDs puede ser usada

durante la fase del mantenimiento para evaluar el impacto de modificaciones y dirección de pruebas. Los requerimientos se pueden verificar en cada fase del ciclo de vida.

8. Integridad

Aplicación : El propósito de esta medida es determinar la integridad de la especificación del software durante la fase de requerimientos. También los valores determinados por las entradas asociados con la medida de integridad que puede ser usado para identificar áreas de problemas dentro de la especificación.

Entradas: La medida de integridad consiste en las siguientes entradas:

B1	Número de funciones no satisfactoriamente definidos
B2	Número de funciones
B3	Número de referencia de datos que no tienen un origen
B4	Número de referencia de datos
B5	Número de funciones definidas no usadas
B6	Número de funciones definidas
B7	Número de funciones referenciadas no definidas
B8	Número de funciones referenciadas
B9	Número de puntos de decisión que no usan todas las condiciones u opciones
B10	Número de puntos de decisión
B11	Número de opciones de condición sin procesamiento
B12	Número de opciones de condición
B13	Número de rutinas de llamadas con parámetros que no están de acuerdo con parámetros definidos
B14	Número de rutinas de llamadas
B15	Número de opciones de condición no definidas
B16	Número de opciones de condición que no tienen procesamiento
B17	Número de condiciones fijas
B18	Número de referencia de datos que no tienen destino

Implementación: la Medida de Integridad (MI) es la suma de pesos de diez derivadas expresados como:

$$MI = \sum_{i=1}^{10} w_i D_i$$

Donde por cada peso i=1, ..., 10, w_i tiene un valor entre 0 y 1, la suma de los pesos es igual a 1 y cada D_i, es un derivado con un valor entre 1 y 0.
Para calcular la medida de la integridad, se deben determinar las definiciones de las entradas para cada aplicación en particular y también la prioridad asociada con los derivados debe ser determina. Esta priorización afectaría los pesos usados para calcular la medida de la integridad.

Cada valor de entrada se determinaría por el número de ocurrencias relativas a la definición de la entrada. Cada derivada es determinada como sigue:

D1=(B2-B1)/B2	Número de funciones satisfactoriamente definidas.
D2=(B4-B3)/B4	Referencia de datos que tiene un origen.
D3=(B6-B5)/B6	Funciones definidas usadas.
D4=(B8-B7)/B8	Funciones referenciadas definidas.
D5=(B10-B9)/B10	Todas las opciones de condición en los puntos de decisión
D6=(B12-B11)/B12	Todas las opciones de condición con procesamiento en puntos de decisión son usadas.
D7=(B14-B13)/B4	Los parámetros de la rutina llamada están de acuerdo con los parámetros definidos en la rutina.
D8=(B12-B15)/B12	Todas las opciones de condiciones son definidas.
D9=(B17-B16)/B17	El procesamiento sigue un conjunto de opciones de condición
D10=(B4-B18)/B4	Referencia de datos tiene un destino

Interpretación: El valor de la medida de la integridad esta en una escala entre 0 y 1 para los pesos apropiados. Un puntaje cerca a 0, se debe rastrear a las entrada(s) sospechosa(s) destacando cualquier necesidad de cambio en la especificación del software. Como los cambios se hacen en la especificación, la medida de la Especificación Incremental se puede graficar mostrando valores si se hacen mejoras y cuán rápidamente. La comparación de medidas de la especificación entre proyectos diferentes permite una relativa evaluación de cumplimiento para mejorar la integridad

Beneficio: La medida de la integridad da una indicación de la integridad (valor cerca a 1) o no integridad (valor cerca a 0) de la especificación del software durante las fases del desarrollo.

9. Estructura del Diseño (ED)

Aplicación: Esta medida es usada para determinar la simplicidad del diseño detallado de un programa de software. También los valores determinados para los parámetros pueden ser usados para identificar problemas dentro del diseño de software.

Entradas:

S1= Número de módulos definidos en la Arquitectura del Programa.

S2= Número de módulos cuya función depende de la fuente de los datos de entrada o que producen datos que se usan en cualquier parte.

S3= Número de módulos cuya correcta función depende del procesamiento previo.

S4= Número de elementos de una base de datos. (objetos de datos, atributos que definen objetos).

S5= Número total de elementos de base de datos no único.

S6= Número de segmentos de base de datos (registros diferentes u individuales).

S7= Número de módulos con una sola entrada y una sola salida.

Implementación: La medida del diseño de estructura es la suma de 6 derivados: Las 6 derivadas son:

Métricas	Di	Peso (Pi) %
Estructura del programa: D1 Si el diseño arquitectónico utiliza un método formal (tal como Diseño de Flujo de Datos o Diseño orientado a objetos), entonces D1=1, caso contrario D1=0.	D1 = [0,1]	
Independencia de módulos:	D2 = 1 – (S2 / S1).	
Módulos no dependientes del procesamiento:	D3 = 1 – (S3 / S1).	
Tamaño de la base de datos:	D4 = 1 – (S5 / S4).	
Compartamentalización de la base de datos:	D5 = 1 – (S6 / S4).	
Característica de entrada/salida del módulo:	D6 = 1 – (S7 / S1).	

Donde: SUM[Pi] = 1, donde Pi es el peso relativo a la importancia

$$ED = \sum_{i=1}^{6} Pi\ Di$$

Interpretación: El valor de la medida de la ED esta entre 0 y 1. Un valor cercano al 0 es considerado mejor que un valor cercano a 1. Valores de la medida de la ED considerados altos se pueden rastrear "atacando" las entradas(s) para destacar prioridad de áreas particulares para los cambios en el Diseño. Como el cambio se hace al Diseño de Estructura, los valores de la medida de la ED pueden ser mostradas incrementalmente, si se hacen las mejoras y cuán rápidamente. El debe ser comparado con anteriores diseño, Si de la comparación el ED es menor que el promedio histórico, debe revisarse el diseño.

Beneficio: La medida de la ED indicara la simplicidad (valor cerca de 0) o complejidad (valor cerca de 1) del diseño de software.

10. La Ciencia del Software de Halstead

Aplicación: Esta medida es aplicada a las propiedades y estructura de programas de computo. Esta proporciona una medida de la complejidad del software existente, predice la longitud de un programa y estima la cantidad de tiempo que un programador en promedio puede demorar en implementar un determinado algoritmo.

Esta medida calcula la longitud del programa contando los operadores y operandos. La medida sugiere que la dificultad de un programa dado puede basar en los cálculos anteriores.

Entradas: La ciencia de Halstead usa un conjunto de entradas que se obtienen a partir del código fuente o terminado el diseño:

n_1 = Número de operadores distintos que aparecen en un programa
n_2 = Número de operandos distintos que aparecen en un programa

N_1 = Número total de ocurrencias de operadores.

N_2 = Número total de ocurrencias de operandos.

Implementación: Halstead define las siguientes métricas:

Vocabulario del Programa:	$n = n_1 + n_2$
Longitud observada del Programa:	$N = N_1 + N_2$
Longitud estimada del Programa:	$\hat{N} = n_1 (\log_2 n_1) + n_2 (\log_2 n_2)$
Volumen del Programa: Representa el volumen de información requerido para especificar un programa, depende del lenguaje de programación.	$V = N (\log_2 n)$
Dificultad del Programa:	$D = (n_1 / 2) (N_2 / n_2)$
Nivel del Programa:	$L = 1 / D$
Esfuerzo:	$E = V / L$
Número de Errores: donde E_0 es el índice de los errores de programación. $3000 \leq E_0 \leq 3200$	$\hat{B} = V / E_0$
Tiempo: donde S es un número que corresponde a dieciocho discriminaciones elementales mentales por segundo. $5 \leq S \leq 20$	$\hat{T} = E / S$

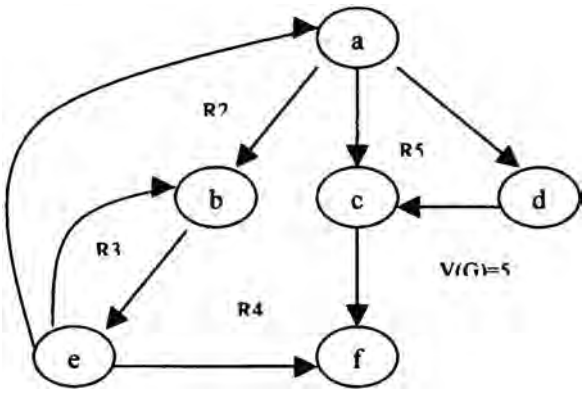
Interpretación: Existen tablas de valores para los índices creados para diversos lenguajes de uso común. Valores de los índices mayores a los valores en estas tablas son un indicador de serios problemas.

Beneficio: La aplicación de la medida de la ciencia del software en la etapa de codificación y pruebas puede ayudar a identificar programas que pueden ser exageradamente complejos y por tanto el esfuerzo de desarrollo es excesivo.

11. Medida de la Complejidad de McCABE

Aplicación: Es una medida de la complejidad del software que se basa en la complejidad ciclomática del grafo del programa de un módulo. El uso de esta medida esta diseñada para imitar la complejidad de un módulo. Se basa en la representación del flujo de control de un programa. La medida de la estructura

del flujo de control permite predecir acerca de un programa la dificultad de comprensión y la magnitud probable de contener defectos. Para ello utiliza unos gráficos llamados grafos de flujo de control.



Entradas:

N = Número de nodos (Grupos secuenciales de sentencias de programa)

E = Número de bordes (flujos de programas entre nodos)

SN = Número de rupturas de nodos (nodos con más de un borde que salen de el).

RG = Número de regiones (áreas limitadas por bordes sin cruces)

Implementación: Usar regiones o nodos y bordes, se requiere una gráfica del módulo fuertemente conectada. Una gráfica fuertemente conectada es cuando un nodo es alcanzable a cualquier otro nodo. Se logra esto por agregar un borde entre el nodo de la salida. Una vez construida la gráfica, se computa la medida como sigue:

$$C = E - N + 1$$

La complejidad ciclomatica es equivalente al número de regiones (RG) o el número de nodos l fraccionados más 1 (SN + 1). Si un programa contiene un N caminos predicativos, tal como una declaración del CASE con N casos, el N camino predicativo contribuye con N - 1 a la cuenta de SN.

Interpretación: Para un módulo típico, diez representan un máximo nivel de complejidad ideal. Para un sistema muy complejo el número ciclomático (complejidad) es más grande. Si el cálculo del número de la complejidad es demasiado grande, se requiere modularización de la codificación.

Beneficio: El número ciclomático puede ser un indicador bueno de la complejidad del módulo, y se usa mantener módulos dentro de un entendible y trabajable tamaño. Esta medida, además de los resultados en la computación del número de caminos distintos en un módulo no considera la complejidad de una sentencia individual.

12. Indice de Madurez del Software (IMS)

Aplicación: La medida usada para cuantificar la claridad del Software, proporcionando una indicación de la estabilidad de un producto de software, basado en los cambios que se producen en cada versión del producto. Los cambios de una base previa a una actual es una indicación de la estabilidad actual del producto. Una base puede ser tanto una versión interna o una entrega externa. El índice de Madurez puede ser calculado para determinar el relativo impacto que los cambios y adiciones tienen en la configuración del software, y cuán estable es esa configuración.

Entradas:

Mt = Número de módulos de la versión actual.

Fc = Número de módulos de la versión actual que han sido modificados.

Fa = Número de módulos de la versión actual que han sido añadidos.

Fdel = Número de módulos de la versión anterior que se han eliminado en la versión actual.

Implementación: El IMS puede ser calculado en dos maneras dependiendo de la data disponible.

1. Para la actual versión del software (entregado o modificado), cuente él numero de funciones (módulo) que han sido cambiados Fc.

2. Para la actual versión de software, cuente él numero funciones (módulos) que han sido adicionadas (Fa) o eliminadas (Fdel).
3. Para la actual versión de software, cuente él numero de funciones (módulos) creados en la actual distribución. (Mt)

$$\mathbf{IMS} = \frac{Mt - (Fa + Fc + Fdel)}{Mt}$$

Donde IMS tiende a 1, si el producto comienza a estabilizarse.

Interpretación: El IMS para cada software liberado o distribuido puede ser controlado para determinar el impacto relativo de los cambios y adiciones que tienen en la configuración del software y se acerca a una configuración estándar.

Beneficio: El índice de madurez proporciona una medida del impacto relativo que las adiciones y cambios en el software que inciden en la configuración actual del software. En resumen, el índice proporciona una indicación de la estabilidad del software durante el período de prueba y evaluación del usuario.

13. Número de Días de la falla

Aplicación: Esta medida representa el número de días que consumen las fallas detectadas en el desarrollo del software desde su detección hasta su eliminación.

Entradas:

(1) Fase en que el defecto fue introducido al sistema.
(2) Fecha en que el defecto fue introducido al sistema.
(3) Fase, fecha y hora en que el defecto es eliminado.

FD_i = días de la falla para la iva falla

Implementación: Por cada falla detectada y eliminada durante una fase, se determina el número de días desde su detección hasta su eliminación. Los días - falla son luego sumados para todas las fallas detectadas y eliminadas, para obtener el número de días - falla a nivel del sistema, incluyendo todas las fallas

detectadas y eliminadas hasta la fecha de entrega. En caso de desconocer la fecha de introducción de la falla, se debe asumir que fue introducida a la mitad de la fase. Esta medida es calculada como sigue:

$$(FD) = \sum_{i} (FD\ i)$$

Interpretación: La efectividad del diseño del software y el proceso de desarrollo depende en eliminar oportunamente las fallas durante todo el ciclo de vida. Esta medida es un indicador de la calidad del diseño del software y del proceso de desarrollo. Por ejemplo un número alto de días de fallas indica o muchas fallas (probablemente debido a un diseño pobre, indicativo de calidad del producto), o larga vida de fallas, o ambos (probablemente debido al esfuerzo de un desarrollo pobre, indicativo de la calidad del proceso).

Beneficio: Esta medida alienta a efectuar inspecciones oportunas y comprobaciones, y poder asistir a la administración en mejorar el diseño y el proceso de desarrollo.

14. Cobertura de Pruebas Modular o Funcional

Aplicación : Esta medida es usada para cuantificar un índice de cobertura de pruebas para la entrega de software. Los datos de entrada son las funciones o módulos. El usuario operacional es mas familiarizado con los requerimientos funcionales del sistema y reportara los problemas en términos de requerimientos funcionales en vez de requerimientos de prueba del modulo. Es la tarea del evaluador obtener o desarrollar los requerimientos funcionales y asociados al módulo con referencias cruzadas.

Entradas:

FE	Número de requerimientos funcionales (módulos) por la cual todos los casos de pruebas han sido satisfactoriamente completados.
FT	Número total de requerimientos funcionales (módulos).

Implementación: El índice de cobertura de pruebas es expresado como ratio del número de funciones (módulos) de software probados entre el número total de funciones (módulos) de software requeridos creados por los usuarios (desarrolladores).

Indice Cobertura de Pruebas Funcionales (modular) = FE/FT

Beneficio: El índice de cobertura de la prueba provee una medida del porcentaje de software probado en cualquiera momento. Para muchos programas de aplicación, la función (modular) índice de la cobertura de la prueba siempre puede estar relativamente bajo a causa de la aplicación del sistema (ej. equipo de prueba automático) y las restricciones del ambiente de la prueba (ej. sistemas electrónicos de guerra).

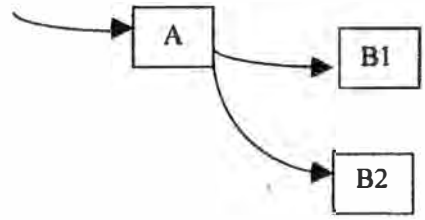
15. Número de Requerimientos en Conflicto

Aplicación : La medida es usada para determinar la fiabilidad del software, representado por una especificación basada en un modelo entidad - relación.

Entradas:

(1) Lista de entradas
(2) Lista de salidas
(3) Lista de funciones llamadas por un programa

(1) = 1
(2) = 2



Implementación: Se identifican los requerimientos desde las gráficas de la arquitectura del software. Las gráficas de similares elementos especificados para más de un requerimiento diferente se examina la inconsistencia de los requerimientos. Las gráficas de más de un elemento especificado para un requerimiento son examinados por inconsistencia de la especificación.

Interpretación: Si existe un mismo elemento del gráfico de la especificación para dos diferentes elementos del requerimiento entonces los requerimientos deben ser idénticos, sino serán inconsistentes. En caso de existir más de un elemento del gráfico de la especificación de un requerimiento, (este debe resultar solamente de la aplicación de técnicas de refinamiento) se debe verificar los elementos de la especificación por posibles inconsistencias.

Beneficio: La identificación de requerimientos en conflicto durante la fase de la arquitectura del software puede evitar problemas más tarde durante el ciclo de vida.

16. Estimado del Número de Fallas Restantes (por Incorporar)

Aplicación: El número estimado de fallas restantes en un programa es relacionado a la confiabilidad del programa. Esta medida puede ser aplicada en cualquier fase del ciclo. La búsqueda continua de fallas en un determinado periodo de tiempo puede ser menor que el requerido para encontrar todas las fallas sembradas. La medida no es calculada al menos que alguna falla no - sembrada sea encontrada.

Entradas:

N_s	Número de fallas sembradas
n_s	Número de fallas sembradas encontradas
N_F	Número de fallas encontradas que no han sido intencionalmente sembrada

Implementación: El Supervisor ACS es responsable de la inserción de errores. El Supervisor ACS inserta (siembra) N_s fallas representativas de defectos esperados. El equipo reporta al Supervisor ACS las fallas encontradas durante el periodo de prueba.

Antes, es necesario un análisis de fallas para determinar los tipos de fallas y su frecuencia relativa de ocurrencia esperada bajo un particular conjunto de condiciones para el desarrollo del software. Aunque una estimación del número de fallas restantes puede hacer basado en las pocas fallas insertadas, la exactitud

de la estimación (y de la confianza en el) de incrementos, con el número de incrementos de inserciones de fallas.

Las fallas deben haberse insertado al azar por todos los elementos del software. La insercción de fallas por el personal deben ser diferentes e independientes de estas personas se buscarán las fallas luego. El proceso de buscar fallas se debe llevar a cabo sin el conocimiento de la falla insertada. Se debe ejecutar la búsqueda para un período de tiempo (o esfuerzo) previamente determinado y cada falla debe ser reportada al Supervisor ACS.

Cada reporte de fallas debe ser revisado para determinar si está en la clase de fallas a estudiar, y si es así, se inserto una falla. La probabilidad máxima estimada del número de fallas (no insertadas) en la clase especificada es:

$$\hat{NF} = \frac{n_F N_s}{n_s}$$

donde $\hat{NF}$ es truncado al un valor entero. La estimación del número restante de fallas es:

$$\hat{NF}_{rem} = \hat{NF} - n_F$$

La probabilidad de hallar n_F de NF fallas y n_F de N_s fallas insertadas, dado que hay $n_F + Ns$ fallas halladas en el programa es

$$C(N_s, n_s)\, C(NF, n_F) / C(NF + N_s, n_F + n_s)$$

En donde; la función: C (x, y) = x! / (x - y)! y! es la combinación de "x" tomados de "y" en un tiempo. Usando esta relación uno puede calcular los intervalos de confianza.

Interpretación: El número de inserciones y fallas descubiertas habilitan el número de fallas restantes ha ser estimadas para el tipo de falla considerada. La frase "tipo de falla a ser considerada" es si, por ejemplo, uno inserta fallas de

tipográfica, uno no esperaría una predicción buena del número de funciones omitidas o fallas del algoritmo. Si uno inserta fallas de este tipo son halladas en las pruebas unitarias, uno no esperaría aplicar la estimación para estas fallas de interfaces halladas en prueba de integración.

Beneficio: Este procedimiento alienta el informe de fallas y competencia entre desarrolladores y equipo de ACS. Desde que las fallas son conocidas, el equipo de ACS sabe que hay algo que buscar. El informe de fallas no necesariamente admite un error por parte de los desarrolladores.

El procedimiento puede ser llevado a cabo por un Supervisor ACS con pequeño esfuerzo, alta efectividad y mínima interferencia con el diseño normal de los desarrolladores. Para el costo de un segundo equipo de prueba independiente, una estimación del número restantes puede obtenerse fallas sin insertar errores.

17. Cobertura de las Pruebas

Aplicación : Es una medida de la completitud del proceso de pruebas desde la perspectiva del desarrollador y del usuario. Esta medida se relaciona directamente con la fase de desarrollo, integración y pruebas unitarias, pruebas del sistema y pruebas de aceptación. La medida puede ser aplicada por desarrolladores en las pruebas unitarias para obtener una medida de la completitud de las pruebas estructurales. El desarrollador puede usar una clase del programa de entradas. El sistema de pruebas puede aplicar la medida de dos maneras: (1) enfocado a los datos del requerimiento, el sistema de prueba puede ganar una vista del usuario en las pruebas funcionales, o (2) la clase de programa de entradas que pueden ser usadas para medir la cantidad de implementaciones en el ambiente del operacional.

Entradas: Las entradas para la Prueba de Cobertura son de dos clases a los programas y a requerimientos. Para programas son de dos tipos: funcionales y de datos. Las entradas en los programas funcionales son: módulos, segmentos, sentencias, nodos, rutas. Las entradas en un programa de datos son equivalentes a clases de datos. Las entradas en los requerimientos son tanto los casos de pruebas o capacidades funcionales.

Implementación: La Prueba de Cobertura (CP) es el porcentaje de tiempos de incurrido en la implementación de los requerimientos ejecutados durante un conjunto de pruebas. Una interpretación simple de la Prueba de Cobertura puede ser expresada por la siguiente fórmula:

$$CP(\%) = \frac{\text{Capacidades Implementadas}}{\text{Capacidades requeridas}} \times \frac{\text{(Entradas de Programas probados)}}{\text{Total de Entradas de Programas}} \times 100$$

Interpretación: La Prueba de Cobertura es sólo una indicación de la completitud de la prueba. Debido al relación de los datos con la función, hay usualmente un infinito número de casos de prueba. Estos pueden reducirse significativamente, si se pueden definir relaciones de equivalencia lógica entre los datos y las funciones.

La confianza al nivel de pruebas de completitud es incrementada al usar entradas más refinadas. Por ejemplo un segmento de la cobertura de 100% es una sentencia más fuerte que 100% de la cobertura del módulo.

Beneficio: La Prueba de la Cobertura es una medida importante en regulación del esfuerzo espero para alcanzar beneficios crecientes de la eficacia del proceso de pruebas. Los casos de prueba pueden desarrollarse para maximizar la medida de la cobertura.

18. Suficiencia de las Pruebas

Aplicación : Esta medida evalúa la suficiencia del software en las pruebas de la integración de software comparando fallas actuales y predichas.

NF	Número Total de fallas predichas en el software.
F_{it}	Número Total de Fallas detectadas a la fecha en las pruebas de integración del software.
F_{pit}	Número de fallas detectadas posteriormente a las pruebas de integración de software.
M_{it}	Número de módulos integrados
M_{tot}	Número de módulos en la configuración final.

Implementación: La densidad de fallas (M1) debe ser usada para calcular el estimado total del número de fallas (NF) en el software. El número de fallas restantes en la porción integrada del software puede ser estimado usando la siguiente ecuación:

$$NF_{rem} = (NF - F_{pit}) M_{it} / M_{tot}$$

Los coeficientes de tolerancia mínima y máxima (l1 y l2) deberán ser establecidas basados en la experiencia. Valores entre 0.5 y 1.5 son los sugeridos para ser usados como valores iniciales.

Interpretación:

Resultado de la Medida	Interpretación	Acción Recomendada	Comentario
L1 NF_{rem} < F_{it} < l2 NF_{rem}	La Prueba de Suficiencia es adecuada	Procede	--
F_{it} > l2 NF_{rem}	Se han detectado más fallas de lo esperado	Procede, pero aplicar la métrica de Densidad de Fallas para cada módulo integrado. Recalcular NF.	Los módulos con error deben ser reemplazados por un módulo rediseñado
F_{it} < l1NF_{rem}	Se han detectado menos fallas de lo esperado	Aplicar la métrica Prueba de Cobertura, luego proceder	Quizás no tiene el número o variedad de pruebas necesarias.

Beneficio: La medida es relativamente simple de aplicar y provee una indicación a priori de la prueba de suficiencia.

19. Exactitud de las Pruebas

Aplicación: Esta medida puede ser usada para determinar la exactitud de los programas de prueba en la detección de fallas. Con una medida de la cobertura de pruebas, puede ser usado para estimar el porcentaje de fallas restantes.

N_s Número de fallas sembradas

N_s Número estimado de fallas sembradas detectadas

Implementación: El software a ser verificado es sembrado con fallas. El número de fallas sembradas detectadas en cada intervalo de tiempo de prueba es registrado a lo largo del tiempo de fallas detectadas y un modelo matemático es

seleccionado para representar las fallas sembradas acumuladas detectadas a lo largo de todo el periodo de prueba. Las fallas sembradas acumuladas detectadas en tiempo infinito es luego estimada y la exactitud de las pruebas es calculada:

$$ALFA = \hat{N}_s / N_s$$

Interpretación: Idealmente, ALFA es independiente de la duración de la prueba y aproximadamente 100%. De cualquier modo, ALFA se puede usar en relación con GAMMA, una medida de la prueba de la cobertura, para estimar el porcentaje de fallas inherentes que quedan al final de prueba.

$$NF_{rem}(\%) = \hat{N}_s / N_s$$

Beneficio: La medida de la exactitud de la prueba es un indicador de la habilidad de una prueba para detectar fallas exactamente, que identificarán áreas del problema en las prácticas de pruebas.

20. Punto de Función

Aplicación: Independiente del lenguaje de Programación, utilizado para estimar el esfuerzo requerido para el desarrollo de Software.

Implementación: Asigna y totaliza pesos a las funciones como sigue:

Parámetro de medida	**Factor de Peso (Simple, Medio, Complejo)**
Número de Entradas del Usuario	* FP (3,4,6) =___
Número de Salidas de Usuario	* FP(4,5,7) =___
Número de Peticiones al Usuario	* FP(3,4,6) =___
Número de Archivos	* FP(7,10,15)=___
Número de Interfaces externas	* FP(5,7,10) =___
Cuenta Total.....	_____

PF = Cuenta Total * [0.65 + 0.01 * SUM (F_i)]
F_i, valores de ajuste de complejidad

Factores de Peso

0 Sin influencia	1 Incidental	2 Moderado
3 Modelo	4 Significativo	5 Esenciales

Valores de ajuste de complejidad	Factor de Peso
1. Requiere el sistema copias de Seguridad y recuperación fiables.	( )
2. Se requieren comunicaciones de Datos.	( )
3. Existen funciones de procesamiento distribuido.	( )
4. Es critico el rendimiento.	( )
5. Será ejecutado el sistema en un entorno existente y fuertemente utilizado.	( )
6. Requiere el Sistema Datos entrada de datos interactivo.	( )
7. Requiere entrada de datos interactiva que las transacciones de entrada se lleven a cabo sobre múltiples pantallas o varias operaciones.	( )
8. Se actualizan los archivos maestros en forma interactiva.	( )
9. Son complejas las entradas, las salidas, los archivos o las peticiones.	( )
10. Es complejo el procesamiento interno.	( )
11. Se ha diseñado el código para ser reutilizable.	( )
12. Están incluidas en el diseño la conversión e instalación.	
13. Se ha diseñado el sistema para soportar múltiples instalaciones en diferentes instalaciones.	( )
14. Se ha diseñado la aplicación para facilitar los cambios y para ser fácilmente utilizada por el usuario.	
	SUMA (Fi)

Productividad = PF / persona - mes

Calidad = Errores / PF

Costo = Total ($) / PF

Documentación = Paginas Documentación / PF

Tamaño del Programa = PF / programas

Función para el tamaño = PF / LDC

21. Punto de Característica

Aplicaciones con complejidad algoritmica.

Parámetro de medida	Peso
Número de Entradas del Usuario	* 4 = ___
Número de Salidas del Usuario	* 5 = ___
Número de Peticiones al Usuario	* 4 = ___
Número de Archivos	* 7 = ___
Número de Interfaces externas	* 3 = ___
Algoritmos	* 3 = ___
Cuenta Total.....	_____

PC = Cuenta Total * [0.65 + 0.01 * SUM (P_i)]

P_i, Valor del Peso de ajuste

Productividad = PC / persona-mes

Calidad = Errores/PC

Costo = Total ($)/PC

Documentación = Paginas Documentación / PC

22. Métricas de Estimación de Costos del Proyecto - COCOMO

Aplicado bajo 3 modelos,(94)

a) **COCOMO Básico**, que calcula el esfuerzo en función del tamaño del programa y LDC

b) **COCOMO Intermedio**, además, considera atributos con una valoración subjetiva (hardware, personal y de los atributos del proyecto)

c) **COCOMO Avanzado,** además, lleva una evaluación del impacto de los conductores de costo en cada fase del desarrollo.

Modelo COCOMO Básico

Esfuerzo aplicado personas-mes (E) $E = a_i * (LDC) \exp (b_i)$

Tiempo Desarrollo-mes (D) $D = c_{ii} * (E) \exp (d_i)$

E es el esfuerzo aplicado en persona-mes,

LDC líneas de Código para el proyecto

Tipo de Proyecto de Software	a_i	b_i	$c_{,i}$	d_i
Orgánico	2.4	1.05	2.5	0.38
Semi-acoplado	3.0	1.12	2.5	0.35
Empotrado	3.6	1.2	2.5	0.32

1. Modo Orgánico, Proyectos relativamente pequeños, equipos pequeños, con buena experiencia en la aplicación, sobre un conjunto de requisitos poco rígido.
2. Semi-acoplado, Proyectos de Software intermedios, equipo con variable nivel de experiencia, deben satisfacer requerimientos poco rígidos.
3. Empotrado, Proyectos de Software desarrollados en un conjunto de Hardware, software y restricciones operativas muy restringidos.

Modelo COCOMO Intermedio

Se consideran los atributos de costo

Atributos del Producto	Escala
1. Fiabilidad del Software Adquirido	( )
2. Tamaño de la Base de Datos de la Aplicación	( )
3. Complejidad del Producto	( )
Atributos del Hardware	
4. Restricciones de Rendimiento en tiempo de ejecución	( )
5. Restricciones de memoria	( )
6. Volatilidad del entorno de la maquina virtual	( )
7. Tiempo de espera requerido	( )
Atributos de Personal	
8. Capacidad de Análisis	( )
9. Capacidad de Ingeniería de Software	( )
10.Experiencia de aplicaciones	( )
11.Experiencia de maquina virtual	( )
12.Experiencia con las maquinas de programación	( )
Atributos del Proyecto	
13.Utilizacion de herramientas de software	( )
14.Aplicacion de métodos de ingeniería de Software	( )
15.Planificacion Temporal de desarrollo requerida	( )
FAE, Factor de Ajuste de Esfuerzo	SUMA (F_i)

Escala (1:muy bajo, 2:bajo, 3:regular, 4:alto, 5:muy alto, 6:extra alto)

Esfuerzo aplicado personas - mes $E = a_i * (LDC)\ exp\ (b_i) * FAE$

LDC líneas de Código para el proyecto

Tipo de Proyecto de Software	a_i	b_i
Orgánico	3.2	1.05
Semi-acoplado	3.0	1.12
Empotrado	2.8	1.20

23. Métricas orientadas al tamaño

Parámetros, LDC es Número de líneas de código

Productividad = LDC / persona-mes

Calidad = Errores/LDC

Costo = Total ($)/LDC

Documentación = Páginas Documentación / LDC

24. Nº de Programas en Mantenimiento

Cuanto mayor el número de programas en mantenimiento con respecto al número total de programas en que se compone nuestro módulo o sistema, menor es el nivel de calidad de nuestro software. Para esto se puede tener un cociente:

$$\frac{\text{Nº Programas en Mantenimiento}}{\text{Nº Programas Total}}$$

El nivel mínimo (cero) mostrará que el sistema tiene un buen grado de calidad. Se puede llevar una estadística del número de Programas en Mantenimiento, en forma semanal ya que esto nos dará el grado de avance que tenemos con respecto al estado inicial.

25. Nº de Cambios / Corrección de Programas

Cuando el número de cambios o correcciones de programas es mayor menor será la calidad de nuestro software; pues, esto demostraría que no se ha llegado a cubrir en forma completa el requerimiento inicialmente solicitado.

Anexo C

HERRAMIENTAS PARA LA GESTION DE LA CALIDAD

1. SQA Suite Versión 6.1

SQA Suite es un conjunto completo de herramientas para pruebas funcionales, genera pruebas funcionales integradas para empresas con aplicaciones Cliente / Servidor. Construye pruebas basado en un repositorio escalable e integró para el servidor. SQA Suite combina el poder de las pruebas en límites principales y el manejo de herramientas comprensivas de un conjunto de estándares para la automatización de pruebas de aplicaciones Cliente / Servidor. SQA Suite esta compuesto de los siguientes componentes: SQA Robot (herramienta para prueba de registros), SQA SiteCheck (administración de Web Site), SQA Manager Web Entry (herramienta Web basada en defectos de entrada) y SQA Manager (planificación de pruebas, administración y herramienta de análisis).

SQA Suite esta preparado para:

SQA Suite - características especiales para Delphi

SQA Suite - características especiales para Power Builder

SQA Suite - características especiales para Visual Básico

SQA Suite - características especiales para diseñador del Oráculo/ 2000

SQA Suite - características especiales para Centura y SQL Windows

SQA Suite - características especiales para People Tools

SQA Suite - características especiales para SAP

Características y Beneficios:

Comprende sólo un conjunto de productos integrados para probar aplicaciones cliente / servidor en ambiente Windows NT, Windows® 98, Windows® 95 y Windows 3.x, e Internet entregando una solución principal de la industria para pruebas cruzadas Windows Cliente / Servidor e Internet.

Asegura que las aplicaciones Windows, Cliente / Servidor e Internet son entregadas listas para el negocio para proveer una solución de prueba de carga, esfuerzo y multi- usuario.

Usa Object Testing™ para prueba completas de 32- y 16-bit en ambiente Windows de objetos y componentes, incluyendo controles de ActiveX, OLE (OCXs), Visual Basic (VBXs), Win32, y objetos Visual Basic, Power Builder, Oracle Developer / 2000 , Delphi, y más.

Combina la rapidez de Object-Oriented Recording™ con un poderoso, integró, ambiente de programación extendible incluyendo Object Scripting™ para crear pruebas automáticas y rápidas con todo el poder de su programación.

Incluye pruebas integradas del Web Site con SQA Site Check™, para entregar un análisis comprensivo del Web Site, medida de la performance y tecnología de reparación.

Entrega un usuario virtual HTTP probador en SQA Load Test para proveer comprensivo de la carga del usuario virtual y prueba del esfuerzo para Servidores Web HTTP.

Otros productos de interés:

Requisite Pro

Clear Quest

Performance Studio

2. Rational Suite

¿Por qué hoy el desarrollo del software es de gran presión y riesgo? Es porque sus aplicaciones son vitales para su empresa. Si fallan, la empresa se tambalea. Cada proyecto de desarrollo tiene demandas por terminar más rápido y los

requerimientos cambian continuamente. Su desafío es distribuir software más frecuentemente sin perder la calidad.

¿Así cómo encuentra este desafío? Tener éxito necesita hacer más que sólo un empujón de productividad individual. Requiere herramientas que enfatizan en ambos trabajo, en equipo y productividad individual. Su desarrollo tiene que colaborar, en lugar de funcionar separadamente. Ud. necesita a su Equipo con Rational Suite.

Sin Compromiso, Rational Suite se unificará a sus equipos cruzado o funcionales con la Arquitectura de la Integración del Equipo única de Rational Suite y optimizando la ejecución de cada miembro del equipo con las mejores herramientas del mercado.

Unifique su Equipo, Rational Suite es una solución para el desarrollo del software diseñado para todo miembro de su equipo de desarrollo, analistas, diseñadores y profesionales de prueba.

Soporta su Ambiente, Rational Suite esta diseño para encajar con su ambiente del desarrollo existente. Apoyamos los ambientes del desarrollo que usa: IDEs (Ambiente integrado de Desarrollo), Visual Basic, tal como, C+, Java y más; y las plataformas que usa: Microsoft™ Windows™ NT, 95, 98 y UNIX.

Herramientas para Gestión de Configuraciones

	CA-Endevor for MVS	CA-Endevor Workstn	CA-Endevor for UNIX	CA-PDM	CA-PanAPT	CA-Panvalet	CA-Librarian	CA-Librarian/CCF	CA-Pan/Merge	CA-Pan/LCM Configuration Management for MVS/ESF	CA-Pan/LCM	Optima's Change Man	Platinum Technology's CCC/ Harvest	InterSolv's PVCS and PVCS Tracker	Princeton Softech Version Merger	Rational's Clearcase and Cleartrack
Sistema Operativo																
MVS	Si	No	No	Si	Sí	Sí	Sí	Sí	Si	Sí	No	Sí	Sí	No	Sí	No
VM	No	No	No	No	No	No	Sí	Sí	No	No	No	No	No	No	No	No
VSE	No	No	No	No	No	Sí	Sí	Sí	No	Sí	Sí	No	No	No	No	No
Windows 3.1	No	Sí	No	No	No	No	No	No	No	No	Si	Sí	Sí	Sí	No	Sí
OS/2	No	Sí	No	No	No	No	No	No	No	No	Sí	Sí	Sí	Sí	No	Sí
Windows NT	No	Sí	No	No	No	No	No	No	No	No	Sí	Sí	Sí	Sí	No	Sí
HP-UX	No	No	Sí	No	No	No	No	No	No	No	Sí	No	Sí	Sí	No	Sí
Sun OS	No	No	No	No	No	No	No	No	No	No	Sí	No	Sí	Sí	No	Sí
AIX	No	No	Sí	No	No	No	No	No	No	No	Sí	No	No	No	No	Si
Sun Solaris	No	No	Sí	No	No	No	No	No	No	No	No	No	Sí	No	No	Sí
Control Cambios																
Previene Regresión de Cambios	Sí	Si	Si	NA	Si	NA	NA	Si	NA	NA	Si	Sí	Si	Si	NA	Si
Seguridad por función	Sí	Si	Si	NA	Si	NA	NA	No	NA	NA	Si	Si	Si	Si	NA	Si
Revisiones y aprobaciones de cambios	Sí	No	Si	NA	Si	NA	NA	No	NA	NA	No	Sí	Si	Si	NA	Si
Consultar versiones anteriores	Sí	Si	Si	NA	NA	Si	Si	Si	NA	NA	Si	Sí	Si	Si	NA	Si
Limpieza de directorios después de migración	Si	Si	Si	NA	Si	NA	NA	Si	NA	NA	No	Sí	Si	Si	NA	Si
Auditoria de cambios	Sí	Si	Si	NA	Si	NA	NA	Si	NA	NA	Si	Sí	Si	Si	NA	Si
Reportes ad hoc y personalizados	Sí	No	No	NA	No	No	No	No	NA	No	Si	No	No	Sí	NA	No
Controla al menos 15 versiones	Sí	Si	Si	NA	NA	Si	Si	Si	NA	NA	Si	Sí	Si	Si	NA	Si
Historia del elemento	Sí	Si	Si	NA	Si	NA	NA	Si	NA	NA	Si	Si	Si	Si	NA	Si
Nombre de Elementos debe ser ingresado solo una vez al ciclo	No	Si	Si	NA	Si	NA	NA	Si	NA	NA	Si	Si	Si	Si	NA	Si
Soporte Control de Desarrollo	Si	Si	Si	Si	Si	NA	NA	Si	Si	NA	Si	Si	Si	Si	Si	Si

	CA-Endevor for MVS	CA-Endevor Workstn	CA-Endevor for Unix	CA-PDM	CA-PanAPT	CA-Panvalet	CA-Librarian	CA-Librarian/CCF	CA-Pan/Merge	CA-Pan/LCM Configuration Management for MVS	CA-Pan/LCM	Optima's Change Man	Platinum Technology's CCC/ Harvest	InterSolv's PVCS and PVCS Tracker	Princeton Softech Version Merger	Rational's Clearcase and Cleartrack
Concurrente o paralelo																
Control de Código																
Maneja Registros de mas de 80 bytes	Sí	Sí	Sí	Sí	Si	Sí	Si	Sí	Sí	Sí	Sí	Sí	Sí	Sí	Sí	Si
Soporta Nombres Largos	No	Sí	No	No	Sí	Sí	Sí	Sí	No	No	No	Sí	Sí	Sí	Si	Si
Compresión de Código Fuente	Sí	Si	Sí	NA	NA	Sí	Si	NA	NA	NA	Sí	Si	Si	Sí	NA	Si
Manejo de Librerías																
Identifica tipo de lenguaje fuente	Sí	Sí	Sí	NA	NA	Sí	Si	NA	NA	Si	Sí	Sí	Sí	Si	NA	Si
Establecer rutas de promoción para diferentes elementos de software	Sí	Sí	Sí	NA	Sí	NA	NA	Si	NA	NA	Sí	Si	Sí	Sí	NA	Si
Asegurar que lo probado es lo que se encuentra instalado.	Sí	Si	Si	NA	Si	NA	NA	Sí	NA	Si	Si	Si	Si	Si	NA	Si
Maneja todos los tipos de software, textos y binarios.	Sí	Si	Si	NA	Si	No	No	Sí	NA	Si	Sí	Si	Si	Si	NA	Si
Almacenar información del modulo fuente module dentro del modulo de carga	Sí	No	No	NA	NA	Si	Si	NA	NA	No	NA	Si	NA	NA	NA	NA
Almacenar en Formato Delta:	Sí	Si	Si	NA	NA	No	Si	NA	NA	NA	Si	No	No	Si	NA	Si
Forward	Sí	Si	No	NA	NA	No	No	NA	NA	NA	Si	No	No	No	NA	No
Reverse	Si	No	No	NA	NA	No	No	NA	NA	NA	No	No	No	Si	NA	No
Acceso al código fuente es controlado y puede ser leído solamente	Sí	Si	Si	NA	NA	Si	Si	NA	NA	NA	Si	NA	NA	Si	NA	Si
Manejo de Configuraciones																
Análisis de Impacto y referencias cruzadas	Sí	No	No	NA	Si	NA	NA	No	NA	Si	Si					
Colecciona y almacena las relaciones y configuraciones de elementos asociados	Si	No	No	NA	No	NA	NA	No	NA	No	No					
Habilidad para mover individuales o todos los módulos, como un paquete involucrado en un cambio.	Si	Si	Si	NA	Si	NA	NA	No	NA	NA	No					

Anexo D

DEFINICIONES

1. El Software

En nuestro trabajo empezamos por definir que es el Software, ya que es nuestro objetivo proporcionar de calidad ha este producto.

"Conjunto de programas internos del ordenador que permiten realizar las funciones asignadas por el usuario" [2].

"Instrucciones de programas de computadoras que cuando se ejecutan proporcionan la función y el comportamiento deseado" [3].

2. Software Crítico

Software cuya falla puede tener un impacto en la seguridad, o puede causar grandes perdidas financieras o sociales.

3. Sistemas de Información

"Un Sistema de Información gerencial sirve para convertir datos de fuentes internas y externas en información la cual es comunicada en una forma apropiada a los administradores de todo nivel en todas las funciones para

permitirles tomar a tiempo las decisiones efectivas de planeamiento y control de las que son responsables"[4].

"Conjunto de componentes que realizan operaciones de procesamiento de datos para: cubrir necesidades diversas a nivel de transacciones, proporcionar información para planeamiento, control y toma de decisiones, producir informes específicos según necesidades" [5].

4. Ingeniería del Software

Una primera definición de ingeniería del software fue propuesta por Fritz Bauer en la primera conferencia importante dedicada al tema:

"El establecimiento y uso de principios de ingeniería robustos, orientados a obtener económicamente software que sea fiable y funcione eficientemente sobre máquinas reales" [3].

Los métodos de la ingeniería del software suministran el "cómo" construir técnicamente el software, abarcan un amplio espectro de tareas que incluyen: Planificación y estimación de proyectos; análisis de los requerimientos del sistema y del software; diseño de estructura de datos, arquitectura de programas y procedimientos algorítmicos; codificación; prueba y mantenimiento. Los métodos de la ingeniería del software introducen frecuentemente una notación especial orientada a lenguaje o gráfica y un conjunto de criterios para la calidad el software.

Las herramientas de la ingeniería del software suministran un soporte automático o semiautomático para los métodos. Hoy, existen herramientas para soportar cada uno de los métodos mencionados anteriormente. Estas herramientas utilizadas para el soporte del desarrollo del software, se les conoce como ingeniería del software asistido por computadora (CASE: acrónimo en inglés de computer–aided software engineering). CASE combina el software, hardware y base de datos de la ingeniería del software (una estructura de datos

que contenga la información relevante sobre el análisis, diseño, codificación y prueba) para crear un entorno de ingeniería del software.

Los procedimientos de la ingeniería del software son la cola que pega a los métodos y herramientas y facilita un desarrollo racional y oportuno del software de computadora. Los procedimientos definen la secuencia en la que se aplican los métodos, las entregas (documentos, informes, formas, etc.) que se requieren, los controles que ayudan a asegurar la calidad y coordinar los cambios, y las guías que facilitan a los desarrolladores del software establecer su desarrollo.

5. Calidad Total

"La Calidad Total, es una filosofía por el cual la administración de sistemas se puede dirigir a la realización eficaz de los objetivos de una organización, asegurar la satisfacción del cliente y llevar hasta el máximo valor al software. Se logra esto mejorando continuamente el sistema de calidad, que consta del sistema social, el sistema técnico y el sistema administrativo. Así, llegar a ser una manera de vida para hacer negocios en toda organización"[16].

6. Calidad del Software

"La calidad del software se define como: concordancia de los requisitos funcionales y de rendimiento explícitamente establecido con los estándares de desarrollo explícitamente documentados y con las características implícitas que se espera de todo software desarrollado profesionalmente" [7].

7. Garantía de la Calidad del Software

"Actividad de protección que se aplica a lo largo de todo proceso de ingeniería de software" [7].

8. Etapa o Fase

Fase del Desarrollo de Software, por donde los elementos de la configuración tendrá que pasar en su Ciclo de Vida. Por ejemplo; Desarrollo, Control de Cambios, Pruebas, Producción, etc.

9. Análisis

Es el estudio en el que se establece los requerimientos de todos los elementos del sistema, para comprender la naturaleza de los programas que se deberán construir, para esto se debe obtener información detallada del dominio de la información y de la función, rendimiento e interfaces requeridas. Esto es, durante la definición, el que desarrolla el software intenta identificar qué información ha de ser procesada, qué función y rendimiento se desea, qué interfaces han de establecerse, qué enlaces de diseño existen y qué criterios de validación se necesitan para definir un sistema correcto y completo. El análisis del sistema abarca los requerimientos globales a nivel del sistema con una pequeña cantidad de diseño de nivel superior. Los requerimientos tanto del sistema como del software se documentan y revisan con el cliente.

10. Diseño

El diseño del software es realmente un proceso que se enfoca sobre tres atributos del programa: estructura de datos, arquitectura del software y detalle procedimental. El proceso de diseño traduce los requerimientos en un conjunto de representaciones (algunas gráficas, otras tabulares o basadas en lenguajes) que describen la estructura de datos, arquitectura y procedimiento algorítmico, de forma que se obtenga la calidad requerida antes que comience la programación. Como los requerimientos, el diseño se documenta y forma parte de la configuración del software.

11. Programación o Construcción

Es el diseño traducido en una forma legible para la máquina (un lenguaje de programación convencional o un lenguaje visual), teniendo como resultado instrucciones ejecutables por la computadora. Si el diseño se describe de una manera detallada, la programación puede realizarse mecánicamente.

12. Pruebas

Terminado de generar el código, se inician las pruebas, este debe ser probado para descubrir los defectos que puedan existir en la función, lógica e implementación. La prueba se enfoca sobre la lógica interna del software,

asegurando que todas las sentencias se han probado, realizando pruebas para asegurar que la entrada definida producirá los resultados que realmente se requieren.

13. Pruebas Unitarias

Pruebas conducentes a verificar la implementacion del diseño de un elemento de Software (ej. Modulo) o una colección de elementos de Software.

14. Pruebas de Integración

Una progresión ordenada de pruebas en el cual los elementos de software, de hardware o ambos son combinados y probados hasta que todo el sistema haya sido integrado.

15. Pruebas de Sistema

El proceso de probar un hardware integrado y un sistema de software para verificar que el sistema cumple los requerimientos especificados.

16. Pruebas de Aceptación

Pruebas Formales conducentes a determinar si o no un sistema satisface su criterio de aceptación y permite al cliente o usuario determinar si o no acepta el sistema.

17. Mantenimiento

Es el cambio que se realiza al Software asociado con una corrección de errores, adaptaciones requeridas por la evolución del entorno del software y modificaciones debidas a los cambios de los requerimientos del cliente para reforzar, mejorar el rendimiento o aumentar funcionalmente el sistema. La fase de mantenimiento replica los pasos de las fases de definición y desarrollo, pero en el contexto del software existente.

18. Verificación

"El proceso de evaluar los productos de una determinada fase para asegurar la exactitud y consistencia con respecto a los productos y estándares proporcionados en esa fase" [Anexo A: ISO 9000-3].

19. Validación

"El proceso de evaluar el software para asegurar el cumplimiento con lo especificado en los requisitos"[Anexo A: ISO 9000-3].

20. Elemento

Objeto en la Configuración; por ejemplo: programa fuente, ejecutable, modulo carga, modulo objeto, documentación, etc.

21. Error

La diferencia entre un valor calculado u observado y el valor verdadero o teóricamente correcto.

22. Defecto

Carencia o falta de los requerimientos especificados en una fase del desarrollo de software, detectado en fases siguientes a la fase que lo origino.

23. Falla

Un resultado incorrecto debido a un mal proceso o mala definición de datos. Inhabilidad del sistema o componente para ejecutar una función de acuerdo a los requerimientos de performance requeridos. Un defecto en el Hardware o algún componente.

24. Equivocación

Acción humana que produce un resultado incorrecto.

Anexo E

LISTA DE DEFECTOS

1. Planificación del Proyecto

Problemas en la Planificación

- No se elaboro un Análisis de riesgos.
- No se elaboro una correcta estimación del esfuerzo.
- No se elaboro un Análisis costo / beneficio.
- No se elaboro un correcto cronograma de actividades.

Problemas en la Conceptualización

- No se identifico correctamente el objetivo del proyecto.
- Incorrecta concepción del problema.
- Mala interpretación de lo comunicado por el Cliente.

2. Análisis de Requerimientos

Problema en la Especificación de Requerimientos

- Requerimientos ambiguos
- Requerimientos duplicados
- Requerimientos inconsistentes.
- Requerimientos incompletos
- Requerimientos incorrectos.

3. Diseño

Problema en la Especificación del Diseño

- Diseño inconsistente.
- Diseño incompleto.
- Diseño incorrecto.
- Desviación deliberada de la Especificación de Requerimientos.

Problema en el manejo de los datos

Error en la relación de entidades

Incorrecta inicialización de los datos

Acceso o Almacenamiento de los datos incorrecto

- Flag o índice incorrecto.
- Empaquetado/Desempaquetado de los datos incorrecto.
- Mala referencia de datos variables.
- Límites de las referencias externas de los datos.

Escalas o Unidades incorrectas de los datos

Incorrecta dimensión de los datos

- Flag o índice incorrecto.
- Tipo variable incorrecto.
- Variable suscrita incorrecta.

Incorrecto alcance de los datos

Problema en las Interfaces

- Interfaces incorrectas.
- Interfaces del modulo inconsistentes.
- Interfaces hombre - máquina inconsistentes.

4. Construcción

Problemas en la Lógica

- Lógica duplica.
- Olvido de casos o pasos.
- Abandono de condiciones extremas.
- Funciones innecesarias.
- Mala interpretación del requerimiento.

- Prueba de la condición perdida.
- Verificación incorrecta de variables.
- Loop iterativo Incorrecto.
- Elementos de Software referenciados pero no definidos.
- Elementos de Software definidos pero no referenciados.

Problemas en las variables

- Variables no definidas.
- Variables no utilizadas.
- Incorrecto número de argumentos.
- Valor asignado no corresponde con el tipo de variable (carácter, número, lógico)
- Comparación de variables de diferente tipo.

Problemas en los Cálculos

Ecuaciones insuficientes o incorrectas

- Cálculos perdidos.
- Operandos incorrectas en las ecuaciones.
- Operadores incorrectas en las ecuaciones.
- Uso incorrecto de paréntesis.

Perdida de Precisión

- Falla en el redondeo o truncamiento.
- Modos mixtos.

Falla en la convención de signos

Problemas de interfaces/tiempo

Subprograma / Módulos mal utilizados

- Mala llamada de subprogramas.
- Localización incorrecta de las llamadas a subprogramas.
- Llamadas a subprogramas que no existen.
- Argumentos del subprograma incoherentes.

Otros problemas

- Comentarios inapropiados.
- Violación de los estándares de programación.
- Error en la representación de datos.

5. Pruebas

Problemas con las Entradas

- Entradas correctas no son aceptadas.
- Entradas erradas son aceptadas.
- Descripción incorrecta o errados.
- Parámetros incorrectos o errados.

Problemas con las Salidas

- Formatos errados.
- Resultados incorrectos.
- Salidas incorrectas / perdidas.
- Ejecución requerida fallida.

Problemas de interfaces/tiempo

Manejo incorrecto de interrupciones

Tiempo de I/O incorrecto

- Tiempo de fallas en que existe pérdida de los datos.

Problemas en los datos

- Datos del sensor incorrectos o perdidos
- Datos del operador incorrectos o perdidos
- Datos de las tablas incorrectos o perdidos
- Datos externos incorrectos o perdidos
- Datos del rendimiento incorrectos o perdidos
- Datos de la entrada incorrectos o perdidos

Otros problemas

- Inapropiada eficiencia del código.
- Inapropiada facilidad de uso.
- El software no puede manejar los problemas por hardware.
- Fallas causadas por arreglos previos.
- Problemas de interoperabilidad.
- Problemas de la ejecución.
- Problemas de conformidad con las normas.

6. Cambios en los programas

- Adición de capacidades innecesarias.
- Eliminación de capacidades necesarias.
- Mala actualización de las capacidades.

7. Documentación

Problemas en los documentación

- Instrucciones ambiguas
- Temas incompletos
- Temas incorrectos
- Temas malos
- Temas en conflicto
- Temas redundates
- Temas confusos
- Temas ilógicos
- Temas no verificables
- Temas no archivables
- Incumplimiento de los cambios en la edición.

Problemas de calidad en la documentación

- No se encontró la aplicación de estándares.
- No se puede hacer seguimiento de los temas.
- Falta de temas importantes.
- Difícil de comprender.
- Documentación Inconsistente.
- Documentación Incorrecta.
- Documentación Incompleta.