**Trabajo de Campo: Resumen para el 1er parcial**

**Capitulo 1. INGENIERIA DEL SOFTWARE:**

**Que es?**

El software de computadora es el producto que los ingenieros de software construyen y después mantienen en el largo plazo.

**Quien lo hace?**

Los ingenieros de software lo construyen y mantienen

**Cuales son los pasos?**

El soft de computadoras se construye de la misma forma que cualquier producto de éxito: mediante la aplicación de un proceso que conduzca a un resultado de alta calidad que satisfaga las necesidades de la gente que usara el producto. Se aplica en un enfoque de ingeniería de software.

**Cual es el producto obtenido?**

Desde el punto de vista del ingeniero de software, el producto obtenido lo forman los programas, el contenido y los doc. que constituyen el software. Desde el punto de vista del usuario, el producto obtenido es la info. resultante que de alguna manera mejora el mundo del user.

El marco que incluye un proceso, un conjunto de métodos y una serie de herramientas se llama ingeniería del software.

**Software**

Para entender el software (y su ingeniería) es importante examinar las características que lo hacen diferente de otras cosas que construye el ser humano. El software es un elemento lógico. El mismo tiene características muy diferentes a las del hard.

1. el soft se desarrolla o construye, no se manufactura en el sentido clásico. Los costos del soft, se concentran en la ingeniería. Esto significa que los proyectos del software no se pueden manejar como si fueran proyectos de manufactura
2. el software no se desgasta, pero si se deteriora. El hard se desgasta. Durante su vida el soft experimenta cambios, debido a esos cambios es que se deteriora. Cualquier falla de soft implica un error en el diseño o proceso mediante el cual se paso del diseño al código maquina ejecutable. Por lo tanto, el mantenimiento del soft implica de manera considerable una complejidad mayor que el del hard.
3. A pesar de que la industria tiene una tendencia hacia la construcción por componentes, la mayoría del soft aun se construye a medida.

**1.3 La Naturaleza cambiante del Software**

**7 grandes categorías** del software de computadora que presenta retos continuos para los ingenieros de software.

1. **Software de Sistemas**: El soft de sistemas es una colección de programas escritos para servir a otros programas
2. **Software de Aplicación**: El soft de aplicación consiste en programas independientes que resuelven una necesidad de negocios especifica.
3. **Software Científico y de ingeniería**: Se caracteriza por algoritmos “devorados de números”.
4. **Software empotrado(Fimware):** Este reside dentro de la memoria de solo lectura del sistema y con el se implementan y controlan características y funciones para el usuario final y el sistema mismo.
5. **Software de línea de productos:** Se puede enfocar en un nicho de mercado limitado
6. **Aplicaciones Basadas en WEB:** Las “WebsApps” son apenas un poco mas que un conjunto de archivos de hipertexto ligados que presentan información mediante texto y algunas graficas.
7. **Software de ingeniería artificial**: este soft utiliza algoritmos no numéricos en la resolución de problemas complejos que es imposible abordar por medio de un análisis directo. En algunos casos se construyen sistemas nuevos, pero en otras las aplicaciones existentes se corrigen, adaptan y mejoran. En el horizonte han aparecido retos nuevos:
   1. Computación ubicua: el reto para los ingenieros de soft será desarrollar soft de sistema y aplicación que permita que dispositivos pequeños, computadoras personales, y sistemas de empresa se comuniquen a través de grandes redes.
   2. Alimentación de la red: la Word wide web se convierte con rapidez en un dispositivo computacional
   3. Fuente abierta. El reto para los ingenieros de soft es construir un código fuente que sea descriptivo en si mismo, pero aun mas importante, desarrollar técnicas que permitan tanto a los clientes como a los diseñadores conocer los cambios realizados y la forma en que se manifiestan dentro del soft.
   4. La nueva economía: La caracterizara la comunicación y la distribución masiva. El reto para los ingenieros de soft es construir aplicaciones que faciliten la comunicación y la distribución de productos en masa mediante productos apenas en formación.

**1.4 Software Heredado:**

Se denomina a el software desarrollado hace décadas, de mantenimiento costoso y evolución riesgosa. Lo caracteriza su longevidad y el ser critico para los negocios, en cuanto a la calidad de los mismos posee código complejos, documentación escasa o inexistencias, historial de cambio manejado con pobreza, etc.

**Evolución del Software:**

El cambio (mantenimiento), conduce este proceso y se presenta cuando se corrigen errores.

Leyes subyacentes derivadas de el estudio de Manny Lehman, en los últimos 30 años:

* **Ley del cambio continuo**.
* **Ley de la complejidad creciente**.
* **Ley de la autorregulación**: se autorregula con la distribución del producto y las mediciones del proceso cercanas a la normal.
* **Ley de la conservación de la estabilidad organizacional**: la tasa de actividad global efectiva promedio en un sistema de tipo electrónico, en evolución, no varia a lo largo del periodo de vida del producto
* **Ley de la conservación de la familiaridad**: todos los involucrados con el sistema, deben mantener el dominio sobre su contenido y su comportamiento.
* **Ley del crecimiento continuo**.
* **Ley de la calidad decreciente.**
* **Ley de sistema de retroalimentación**: deben tratarse de forma que se obtengan mejorías significativas sobre cualquier base razonable.

**Capitulo 2: El PROCESO: UNA VISION GENERAL**

**Que es un proceso de software desde el punto de vista técnico?**

Un proceso de soft se define como un marco de trabajo para las tareas que se requieran en la construcción de software de alta calidad.

Un proceso de soft define el enfoque que se adopta mientras el soft esta en desarrollo. Pero la ingeniería del soft también abarca las tecnologías que requiere el proceso (métodos técnicos, y herramientas automatizadas)

La ingeniería del software es una tecnología **estratificada,** debe estar sustentado en un compromiso con la calidad. La base que soporta la ingeniería del soft es la calidad.

La base de la ingeniería del software es el estrato del proceso.

**El proceso de la ingeniería del software** es el elemento que mantiene junto los estratos de la tecnología y que permite el desarrollo racional y a un tiempo del software de computadora.

**El proceso** define un marco de trabajo, que debe establecerse para la entrega efectiva de la tecnología de la ingeniería del software.

**El proceso de soft** forma la base para el control de la gestión de los proyectos del soft y establece el contexto en el cual se aplican los métodos técnicos, se generan los productos del trabajo, se establecen los fundamentos, se asegura la calidad y el cambio se maneja de manera apropiada.

**Los métodos de la ingeniería del** soft proporcionan los “como” técnicos para construir el soft. **Abarcan un amplio espectro de tareas que incluyen la comunicación, el análisis de requisitos, el modelado del diseño, la contracción del programa y la realización de pruebas y el soporte.**

**Las herramientas de la ingeniería del soft** proporcionan el soporte automatizado o semiautomatizado para el proceso y los métodos. Cuando las herramientas se integran de forma que la información que cree una de ellas pueda usarla la otra, se dice que se ha establecido un sistema para el soporte del desarrollo del soft, que con frecuencia se denomina ingeniería del soft asistida por computadora.

**Marco de Trabajo Para El Proceso:**

Un marco de trabajo establece la base para un proceso de software completo al identificar un numero pequeño de actividades del marco de trabajo aplicables a todos los proyectos de software

El marco de trabajo del proceso abarca un conjunto de actividades sombrilla aplicables a lo largo del proceso de soft.

**Cuales son las 5 actividades del marco de trabajo del proceso general?**

1. **Comunicación**: intensa colaboración y comunicación con los clientes. Abarca la investigación de requisitos y otras actividades relacionadas.
2. **Planeación**: esta actividad establece un plan para el trabajo de la ingeniería del soft. Describe las tareas técnicas, los riesgos probables, recursos q serán requeridos, los productos del trabajo que han de producirse y un programa de trabajo.
3. **Modelado:** abarca la creación de modelos que permiten al desarrollador y al cliente entender mejor los requisitos del soft y el diseño que lograra satisfacerlos.
4. **Construcción:** combina la generación del código y la realización de pruebas necesarias para descubrir errores en el código.
5. **Despliegue:** el soft se entrega al cliente, quien evalúa el producto recibido y proporciona información basada en su evolución.

La actividad de elaboración del modelo la componen dos acciones de la ingeniera del soft: **análisis y Diseño**.

**Análisis**: abarca un conjunto de tareas de trabajo que conducen a la creación del modelo de análisis.

**Diseño:** abarca tareas de trabajo que crean un modelo de diseño

El marco de trabajo descrito en la visión general de la ingeniería del soft lo completa una serie de actividades sombrilla, que incluyen:

Seguimiento y control del proyecto del soft

* gestión del riesgo
* Aseguramiento de la calidad del soft.
* Revisión técnicas formales
* Medición
* gestión de la configuración del soft.
* gestión de la reutilización
* Preparación y producción del producto de trabajo

Las actividades sombrilla se aplican durante el proceso de software.

La IMCM representa un modelo completo de proceso en dos formas diferentes

1. Como un modelo Continuo: describe el proceso en dos dimensiones. Cada área del proceso se evalúa de manera formal contra las metas y practicas especificas y se clasifica de acuerdo con los siguientes niveles de capacidad:
   * **NIVEL 0: INCOMPLETO.** El área de proceso no se realiza o todavía no alcanza todas las metas y objetivos definidos para el nivel 1 de capacidad.
   * **NIVEL 1: REALIZADO.** Todas las metas especificas del área del proceso han sido satisfechas, las tareas de trabajo requeridas para producir el producto especifico han sido realizadas.
   * **NIVEL 2: ADMINSITRADO.** Todos los criterios del nivel 1 han sido satisfechos, todo el trabajo asociado con el área de proceso se ajusta a una política organizacional definida.
   * **NIVEL 3: DEFINIDO.** Todos los criterios del nivel 2 se han cumplido. El proceso esta adaptado al conjunto de procesos estándar de la organización, de acuerdo con las políticas de adaptación de esta misma y contribuye a la información de los productos del trabajo, mediciones y otras mejorías del proceso para los activos del proceso organizacional
   * **NIVEL 4: ADMINISTRADO EN FORMA CUANTITATIVA.** Todos los criterios del nivel 3 han sido cumplidos. El área del proceso se controla y mejora mediante mediciones y evaluación cuantitativa.
   * **NIVEL 5: MEJORADO**. Todos los criterios del nivel 4 han sido satisfechos. Además el área del proceso se adapta y mejora mediante el uso de medios cuantitativos para conocer las necesidades cambiantes del cliente y mejorar de manera continua la eficacia del área del proceso que se esta considerando.

La IMCM define cada área del proceso en función de “metas especificas” y de las “practicas especificas” requeridas para alcanzar dichas metas.

Las metas especificas establecen las características que deben existir para que las actividades implicadas por un área de proceso sean efectivas.

Las practicas especificas convierten una meta en un conjunto de actividades relacionadas con el proceso

1. Como un modelo Discreto

**Áreas de proceso requeridas para alcanzar un nivel de madurez:**

**Nivel: De Optimización**

**Enfoque**: Mejora continua del proceso

**Áreas del proceso**: innovación organizacional y despliegue, análisis casual y resolución.

**Nivel: Gestionado de modo cuantitativo**

**Enfoque**: Gestión cuantitativa.

**Áreas del proceso**: Ejecución del proceso organizacional, Gestión cuantitativa del proyecto.

**Nivel: Definido**

**Enfoque**: Estandarización del proceso

**Áreas del proceso**:

* + - Desarrollo de requisitos.
    - Solución técnica.
    - Integración del producto.
    - Verificación.
    - Validación.
    - Enfoque del proceso organizacional.
    - Definición del proceso organizacional.
    - Capacitación organizacional.
    - Gestión integrada del proyecto.
    - Gestión integrada del proveedor.
    - Gestión del riesgo.
    - análisis y resolución de la decisión.
    - Ambiente organizacional para la integración.
    - Equipo integrado

**Nivel: Gestionado**

**Enfoque**: Gestión básica del proyecto

**Áreas del proceso**:

* Gestión de requisitos.
* Planeación del proyecto.
* Monitoreo y control del proyecto.
* Gestión de acuerdos del proveedor.
* Medición y análisis.
* Aseguramiento de la calidad del producto y del proceso.
* Gestión del a configuración.

**Nivel: Ejecutado**

**Patrones del Proceso:**

**Que es un patrón del proceso?**

Un patrón de proceso, ofrece una plantilla: un método consistente para describir una característica importante del proceso del software.

Mediante la combinación de patrones, un equipo de software puede construir un proceso que satisfaga lo mejor posible las necesidades de un proyecto.

Los patrones pueden definirse en cualquier grado de abstracción.

Ambler, propuso la siguiente plantilla para describir un patrón de proceso:

* **Nombre del patrón**.
* **Propósito:** se describe con brevedad el objetivo del patrón
* **Tipo:** sugiere 3

Patrones de Tarea: definen una acción de la ingeniería del soft o una tarea de trabajo que es parte del proceso y relevante para una practica exitosa de la ingeniería del soft

Patrones de Escenario: definen una actividad del marco de trabajo para el proceso.

Patrones de Fase: definen la secuencia de actividades del marco de trabajo que ocurre junto con el proceso.

* **Contexto Inicial**: Se describen las condiciones en las cuales se aplica el patrón. Que actividades organizacionales o relativas al equipo han ocurrido. Cual es el estado de entrada para el proceso y que información de ingeniería del soft o información del proyecto ya existe.
* **Problema:** se describe el problema que debe resolver el patrón.
* **Solución:** se describe la implementación del patrón.
* **Contexto resultante:** Se describe las condiciones que habrá una vez que el patrón haya sido implementado con éxito.
* **Patrones relacionados:** se proporciona una lista de todos los patrones de proceso directamente relacionados con este, en forma jerárquica o de alguna otra forma.
* **Usos conocidos/Ejemplos:** se indican los ejemplos específicos en los cuales el patrón es aplicable

Los patrones de proceso proporcionan un mecanismo efectivo para describir cualquier proceso de software.

**Evaluación del proceso:**

En la evolución se pretende comprender el estado actual del proceso de software y se intenta mejorarlo

**El Proceso del software** es examinado por **la evolución del proceso de software** que conduce a:

* Mejoramiento del proceso de software y Determinación de la capacidad.

La Determinación de la capacidad a su vez, motiva el mejoramiento del proceso de software, y este identifica modificaciones al proceso de software.

La Determinación de la capacidad también identifica capacidades y riesgos de proceso de software

**Proceso de Software Personal (PSP):**

Resalta la medida personal del producto de trabajo que se produce y la calidad resultante del producto de trabajo. El PSP además, responsabiliza al profesional encargado de la plantación del proyecto y le confiere el poder de controlar la calidad de todos los productos de trabajo del soft que se desarrollan.

El modelo PSP define 5 actividades del marco de trabajo

* Plantación
* Diseño de alto nivel
* Revisión del diseño de alto nivel.
* Desarrollo.
* Análisis de resultado

El PSP destaca la necesidad de registrar y analizar los tipos de errores que se cometen para desarrollar estrategias encaminadas a eliminarlos.

**El Proceso de Software en equipo (PSE):**

La meta del PSE es construir un equipo de proyecto “autodirigido” que se organice para producir un soft de alta calidad.

El PSE define las siguientes actividades del marco de trabajo:

* Lanzamiento
* Diseño de alto nivel.
* Implementación.
* Integración y prueba.
* Análisis de resultado.

Estas actividades permiten al equipo planear, diseñar, y construir un software de una manera disciplinada al mismo tiempo que miden de modo cuantitativo el proceso y el producto.

Tanto el PSP como el PSE son enfoques riguroso para la ingeniería del software que ofrece beneficios distintos y cuantificables en producción y calidad.

**Tecnología del Proceso:**

Las herramientas de tecnología de proceso permiten que una organización de software construya un modelo automatizado del marco de trabajo común el proceso, de los conjuntos de tareas y las actividades sombrilla explicadas en la sección.

**Herramientas de modelado**: se utilizan para representar los elementos clave de un proceso para que este pueda entenderse con mayor claridad.

**Producto y Proceso:**

Si el proceso es débil, sin duda el producto final sufrirá las consecuencias.

**Capitulo 3. MODELOS PRESCIPTIVOS DEL PROCESO:**

Los ingenieros del sofá han elegido de manera tradicional un marco de trabajo genérico para el proceso, el cual incluye las siguientes actividades dentro del marco:

* Comunicación
* Plantación
* Modelado
* Construcción
* Desarrollo

Se les llama prescriptivos porque prescriben un conjunto de elementos del proceso: actividades del marco de trabajo, acciones de ingeniería del software, tareas, productos del trabajo, aseguramiento del a calidad y mecanismo de control del cambio para cada proyecto.

**Modelo en Cascada:**

Enfoque sistemático, secuencial hacia el desarrollo del software, que se inicia con la especificación de requerimientos del cliente y que continua con la plantación, el modelado, la construcción ye l despliegue para culminare n el soporte del soft terminado.

**Desventajas**:

* A pesar de que el modelo lineal incluye iteraciones, lo hacen de manera indirecta. Como resultado, los cambios confunden mientras el equipo de proyecto actúa.
* Es difícil parea el cliente establecer todos los requisitos de manera explicita.
* Un error grave será desastroso sin o se detecta antes del a revisión del programa

**Ventajas:**

Modelo de proceso útil en situaciones donde los requerimientos están fijos y donde el trabajos e realiza, hasta su conclusión, de manera lineal.

**Modelo de Proceso Incremental:**

El modelo incremental, es iterativo por naturaleza. Se aplica en situaciones en las cuales los requisitos están bien definidos y son estables

**Ventajas:**

* Entrega una serie de lanzamientos, llamados incrementos, que proporcionan en forma progresiva mas funcionalidad para los clientes a medida que se entrega cada uno del os incrementos.
* Los incrementos se pueden planear para manejar los riesgos técnicos.

**El Modelo DRA: (Desarrollo rápido de aplicaciones):**

Proceso incremental, que resalta un ciclo de desarrollo corto. Sea plica para proyectos grandes que se deben entregar en marcos de tiempo muy reducidos.

**Desventajas:**

* Para proyectos grandes, pero escalables, el DRA necesita suficientes recursos humanos para creare l numero correcto de equipos DRA.
* Si los desarrolladores y clientes no se comprometen con las actividades rápidas necesarias para completar el sistema en un marco de tiempo muy breve, los proyectos de DRA fallan
* Si un sistema no se puede modular en forma apropiad ala construcción del os componentes necesarios para el DRA será problemática.
* Si el alto rendimiento es un aspecto importante, y se alcanzara al convertir interfaces en componentes del sistema, el enfoque DRA podría no funcionar
* El DRA seria inapropiado cuando los riesgos técnicos son altos.

**Modelos del Proceso Evolutivos:**

Los modelos evolutivos son iterativos, los caracteriza la forma en que permiten que los ingenieros de software desarrollen versiones cada vez mas completas del software

Están diseñados para ajustarse al cambio.

**Construcción de Prototipos**

El paradigma de construcción de prototipos ayuda al ingeniero de sistemas ya l cliente a entender de mejor manera cual será el resultado del a construcción cuando los requisitos estén satisfechos.

El paradigma de construcción de prototipos se inicia con la comunicación.

El prototipo debería servir como un mecanismo para identificar los requisitos del software.

El prototipo puede servir como “Primer sistema”

**Desventajas:**

* Perdida de calidad en el producto, cuando el cliente pides obre el mismo la aplicación de unos pequeños ajustes.
* Utilización en algunas ocasiones de lenguaje o sistema operativo inapropiado por parte del desarrollador.
* Implican un problema para la plantación del proyecto debido al número incierto de ciclos requeridos para construir el producto.
* No establece establecen la velocidad máxima del a evolución
* Se deben enfocar en la flexibilidad y extensibilidad en vez de en la alta calidad

La construcción de prototipos puede ser un paradigma efectivo para la ingeniería del soft. La clave es definir las reglas del juego desde el principio

**Modelo Espiral:**

Proceso evolutivo que conjuga la naturaleza iterativa del a construcción del prototipo con los aspectos controlados y sistemáticos del modelo en cascada.

Un proceso en espiral se divide en un conjunto de actividades de marco de trabajo que define el equipo de ingeniería del software.

El modelo en espiral se puede adaptar aplicando a través del ciclo de vida completo de una aplicación, desde el desarrollo del concepto hasta el mantenimiento.

El modelo en espiral es un enfoque realista para el desarrollo de software.

**Ventajas:**

* Emplea la construcción de prototipos como un mecanismo encaminado a reducir riesgos
* Permite al desarrollador la aplicación del enfoque del a construcción de prototipos en cualquier etapa evolutiva del producto.

**Desventajas:**

* Difícil convencer al cliente de que el enfoque evolutivo es controlable.
* Si un riesgo importante no se descubre y administra, sin duda surgirán problemas.
* Implican un problema para la plantación del proyecto debido al numero incierto de ciclos requeridos para construir el producto
* No establece establecen la velocidad máxima del a evolución.
* Se deben enfocar en la flexibilidad y extensibilidad en vez de en la alta calidad

**El modelo de desarrollo concurrente:**

Se representa en forma esquemática como una serie de actividades del marco de trabajo, acciones y tareas de la ingeniería del software y sus estados asociados.

Este modelo define una serie de eventos que dispararan transiciones de estado a estado para cada una de las actividades, acciones o tareas del a ingeniería del software.

Este modelo se aplica a todos los tipos de desarrollo del software y proporciona una visión exacta del estado actual de un proyecto, define una red de actividades

**Modelos especializados de Proceso:**

Tienen a aplicarse cuando se ha elegido un enfoque de ingeniería del software definido de una manera muy estrecha.

**Desarrollo basado en Componentes:**

**NCSC**: Se pueden emplear cuando el soft esta en proceso de construcción. Estos componentes proporcionan funcionalidad dirigida con interfaces bien definidas que permiten que el componentes e integre en el software.

**DBC:** Incorpora muchas características del modelo en espiral. Es evolutivo por naturaleza y exige un enfoque iterativo para la creación del soft.

El modelo de desarrollo basado en componentes incorpora los siguientes pasos:

* Productos basados en componentes disponibles se investiga y evalúan para el dominio de aplicación en cuestión.
* Se consideran los aspectos de integración de componentes.
* Se diseña una arquitectura de software para adaptar los componentes.
* Los componentes se integran en la arquitectura
* Se realizan pruebas detalladas para asegurar una funcionalidad apropiada.

**Ventajas:**

Reutilización del software, la cual proporciona beneficiosa los ingenieros del software.

**Modelo de Métodos Formales:**

Comprende un conjunto de actividades que conducen a la especificación matemática del software de computadora. Los métodos formales permiten que un ingeniero de software especifique, desarrolle y verifique un sistema basado en computadora al aplicar una notación matemática rigurosa.

Cuando los métodos formales se utilizan durante el diseño, sirven como base para la verificación de programas y por consiguiente, permiten que el ingeniero de software descubra y corrija errores que de otra manera podrían no haberse detectado.

Los modelos de métodos formales ofrecen la promesa de un software libre de defectos.

**Si los métodos formales pueden demostrar la corrección del software, porque no se utiliza en forma extensa?**

* Caro y consume mucho tiempo
* Se requiere de una capacidad detallada
* Difícil de utilización de estos modelos como un mecanismo de comunicación con clientes que no tienen muchos conocimientos técnicos.

**Desarrollo del software orientado a aspectos: (DSOA)**

Es un paradigma del a ingeniería del soft relativamente nuevo que proporciona un proceso y enfoque metodológico para definir, especificar, diseñar, y construir aspectos

Quizá este proceso adopte características del os modelos de proceso en espiral y concurrente

**Proceso Unificado: (PUD)**

Es un intento encaminado a reunir los mejores rasgos y características de modelos de procesos de software.

El PU reconoce la importancia del a comunicación con el cliente y los métodos encaminados a describir el punto de vista del cliente con respecto a 1 sistema.

El PU enfatiza el importante papel del a arquitectura de software y “ayuda al arquitecto a enfocarse en las metas correctas, como el entendimiento, el ajuste del os cambios futuros y la reutilización”.

Sugiere un flujo de proceso iterativo e incremental y proporciona el sentido evolutivo esencial ene l desarrollo del soft moderno.

PU, un marco de trabajo para la ingeniería de soft orientada a objetos, mediante la utilización del UML

En la actualizada el PU y el UML, se emplean de forma amplia en proyectos OO de todos los tipos.

**Fases del Proceso Unificado:**

**Inicio:** Abarca la comunicación con el cliente y las actividades de planeación. Producto mas importante en esta fase: Modelo de caso de uso

**Elaboración:** Abarca la comunicación con el cliente y las actividades de modelado genérico del proceso. Producto: productos de trabajo que elabora requisitos, así como una descripción arquitectónica y un diseño preliminar. Se revisan en esta fase los riesgos y el plan de proyecto para asegurar que cada uno de ellos conserve su validez.

**Construcción**: Igual a la actividad de construcción definida para el proceso genérico del soft. Produce un modelo de implementación que traduce las clases de diseño en componentes de soft que se construirán para ejecutar el sistema y un modelo de despliegue convierte los componentes ene l ambiente físico de computación. Por ultimo un modelo de prueba describe las pruebas empleadas para asegurar que los casos de usos e reflejen de manera apropiada ene l soft que se ha construido.

**Transición:** abarca las ultimas etapas del a actividad genérica de construcción y la primera parte del a actividad genérica de despliegue. Esta fase entrega el incremento del soft y evalúa los productos de trabajo elaborados durante la etapa en que los usuarios finales trabajan con el soft. Se produce además la retroalimentación proveniente del as pruebas beta y los requerimientos cualitativos de cambio.

**Producción:** coincide con la actividad de despliegue del proceso genérico. Durante esta fase se monitorea e luso subsiguiente del soft, se proporciona el soporte para el ambiente operativo, y se reciben y evalúan los informes de defectos y los requerimientos de cambios.

Es probable que mientras se realizan las fases de construcción, transición y producción y ase hayan iniciado los trabajos para el siguiente incremento del software.

Esto significa que las 5 fases del P Uno suceden en una secuencia, sino en una concurrencia por etapas.

**Capitulo 4: PROCESO DE SOFTWARE Y MÉTRICAS DE PROYECTOS:**

4 Razones para medir los procesos del software, los productos y los recursos.

* **Caracterizar**: para comprender mejor los procesos, los productos, los recursos y los entornos y para establecer las líneas base para las comparaciones con evaluaciones futuras.
* **Evaluar**: para determinar el estado con respecto al diseño. También para valorar la consecución de los objetivos de calidad, y para evaluar el impacto de la tecnología y las mejoras del proceso en los productos y procesos.
* **Predecir**: para poder planificar
* **Mejorar**

**Medidas, Métricas e Indicadores**

**Medida**: proporciona una indicación cuantitativa de la extensión, cantidad, dimensiones, capacidad o tamaño de algunos atributos de un proceso o producto.

**Medición**: ese l acto de determina runa medida

**Métrica del soft:** relatad e alguna formal as medidas individuales sobre algún aspecto.

Un ingeniero de soft recopila medidas, y desarrolla métricas para obtener indicadores.

**Un indicador** es una métrica o una combinación de métricas que proporcionan una visión profundad el proceso del soft, del proyecto del soft o del producto en si.

Los indicadores de proceso permiten a una organización de ingeniería del soft tener una visión profunda de la eficacia de un proceso ya existente. También permite que los gestores evalúen lo que funciona y lo que no.

Permiten al gestor de proyectos de soft:

* Evaluar el estado del proyecto en curso
* Seguir la pistad e los riesgos potenciales
* Detectar lasa reas de problemas antes de que se conviertan en criticas
* Ajustar el flujo y las tareas del trabajo
* Evaluar la habilidad del equipo del proyecto en controlar la calidad del os productos de trabajo del soft

**Métricas del proyecto:**

Se utilizan para propósitos estratégicos. Las medidas del proyecto de soft son tácticas. Las métricas de proyectos y lo sindicadores derivados de ellos lo sutilizan un gestor de proyectos y un equipo de soft para adaptar el flujo de trabajo del proyecto y las actividades técnicas.

**Tienen 2 aspectos fundamentales**:

* Se utilizan para minimizar la planificación de desarrollo, haciendo los ajustes necesarios que eviten retrasos y reduzcan problemas y riesgos potenciales.
* Se utilizan para evaluar la calidad de los productos en el momento actual y cuando sea necesario, modificando el enfoque técnico que mejore la calidad

**Como debemos utilizar las métricas durante el proyecto?**

Todos los proyectos deberían medir:

* **Entradas**: la dimensión de los recursos, que se requieran para realizar el trabajo
* **Salidas**: medidas de las entregas o productos creados durante el proceso de ingeniería del soft.
* **Resultados**: medidas que indican la efectividad de las entregas.

**Mediciones del software:**

Medidas Directas: se incluye el coste y el esfuerzo aplicado. Entre las medidas directas del producto se incluyen las líneas de código (LDC) producidas, velocidades de ejecución, tamaño de memoria, y los defectos informados durante un periodo de tiempo establecido.

Medidas Indirectas: se incluyen la funcionalidad, calidad, complejidad, eficiencia, fiabilidad, facilidad de mantenimiento y muchas otras.

El dominio de las métricas del software se dividen en métricas de:

* Proceso
* Proyecto
* Producto

**Métricas Orientadas al Tamaño:**

Provienen de la normalización de las medidas de calidad y/o productividad considerando el tamaño del soft que se haya producido.

**Las métricas son**:

* Errores por KLDC
* Defectos por KLDC
* E por LDC
* Paginas de documentación por KLDC
* Errores por persona-mes
* LDC por persona-mes
* f por pagina de documentación.

**Métricas Orientadas a la Función:**

Utilizan una medida del a funcionalidad entregada por la aplicación como un valor de normalización. Se determinan 5 características de dominios de información. Los valores del os dominios de información se definen de la siguiente forma:

* Numero de entradas del usuario
* Números de salidas de usuarios
* Numero de peticiones de usuario
* Numero de archivos
* Numero de interfaces externas

**Métricas ampliadas de punto de Función:**

La extensión del os puntos de función se utiliza en la ingeniería, en las aplicaciones de tiempo real y en las aplicaciones orientada sal control.

Las características del as 3 dimensiones del software (datos, funcionales y control), se cuentan, cuantifican y transforman en una medida que proporciona una indicación del a funcionalidad entregada por el soft, llamada punto de función.

**Métricas para la calidad del soft:**

**Medida de la calidad:**

Corrección: la más común es defectos por KLDC

* Facilidad de mantenimiento: se deben utilizar medidas indirectas. Una simple métrica orientada al tiempo ese l tiempo medio de cambio (TMC)
* Integridad: para medirla se tienen que definir 2 atributos adicionales. Amenaza y Seguridad
* Facilidad de uso: se puede medir con 4 características:
  + - Habilidad intelectual y/o física requerida para aprender el sistema.
    - Tiempo requerido para llegar a ser moderadamente eficiente ene l uso del sistema
    - Aumento neto en productividad, medida cuando alguien utiliza el sistema moderadamente y eficientemente
    - Valoración subjetiva del a disposición del os usuarios hacia el sistema.

**Eficiencia del a eliminación de defectos**

EED, e suna medida del a habilidad de filtrar las actividades del a garantía de calidad y de control al aplicarse a todas las actividades del marco de trabajo del proceso.

EED = E/ (E+D)

E= numero de errores encontrados antes del a entrega del software al usuario final.

D= es el numero de defectos encontrados después de al entrega.

**Que es la eficiencia de eliminación de defectos?**

El valor ideal del EED=1, no se han encontrado defectos en el soft.

De forma realista, D será mayor que 0, pero el valor de EED todavía se puede aproximar a 1

**Integración del as métricas dentro del proceso de ingeniería de software**

Cuando en la revisión del análisis no se detectan errores, se pasan a la tarea de diseño. Cuando se utilizan en este contexto, EED se vuelve a definir como:

EED i = Ei / (Ei + Ei+1 )

**Argumentos para las métricas del software:**

***Porque es tan importante medir el proceso de ingeniería del software y el producto que producen?.***

Si no se miden no hay forma real de determinar si se esta mejorando. Y sin o se esta mejorando, se esta perdido.

Por lo tanto la medición se utiliza para establecer un alinea base del proceso desde donde se pueden evaluar las mejoras.

**Establecimiento de una línea base:**

Estableciendo una línea base de métricas se pueden obtener beneficios a nivel de proceso, proyecto y producto.

Los datos de la línea base deben tener los siguientes atributos:

1. Los datos deben ser razonablemente exactos.
2. Los datos deben reunirse del mayor numero de proyectos que sea posible
3. Las medidas deben ser consistentes
4. Las aplicaciones deben ser semejantes para trabajar en la estimación

**El desarrollo del a métrica y del a OPM**

Método de desarrollo de métrica mas ampliamente aplicado y mejor conocido.

Para que una organización tenga un programa de medida exacto es necesario que tenga constancia de 3 componentes:

* Proceso donde pudieran articularse metas u objetivos para sus proyectos
* Proceso donde estas metas pudieran ser traducidas al os datos del proyecto que exactamente reflejasen dichas metas u objetivos en términos de soft.
* Un proceso que interpretara los datos del proyecto con el fin de entender los objetivos.

Los objetivos de OPM pueden articularse por medio de 3 plantillas que cubren:

El propósito: se utiliza para articular o comparar l oque esta siendo analizado y el propósito de dicha parte del proceso.

la perspectiva: pone su atención en los factores que son importantes dentro del propio proceso o producto que esta siendo evaluado

El entorno: contexto dentro del cual el método OPM sea plica e implica el examen del personal, la propia empresa y los entornos de recursos en los que el análisis se esta llevando a cabo

**Variación de la gestión: control de procesos estadísticos**

Técnica llamada grafico de control: permite que los individuos o las personas interesadas en la mejora del procesos de software determine si la dispersión, y la localización o métrica de procesos que es estable o inestable.

2 gráficos de control distintos se usan en la evaluación de los datos métricos:

* Grafico de control de rango móvil
* Grafico de control individual

**Como podemos estar seguros de que las métricas que hemos recopilado son estadísticamente validas?**

1. Calcular los rangos móviles
2. Calcular la medida del os rangos móviles
3. Multiplicar la medida por 3.268

**Métricas para organizaciones pequeñas:**

una organización pequeña puede seleccionar el siguiente conjunto de medidas fácilmente recolectables:

* Tiempo (horas o días)
* Esfuerzo (horas- persona)
* Tiempo requerido (horas o días)
* Errores descubiertos durante el trabajo para realizar el cambio.
* Defectos descubiertos después de que el cambio se haya desviado a la base del cliente.

**Establecimiento de un programa de métricas de software**

Según el IIS, para establecer un programa de medición de soft dirigido hacia objetivos, ha desarrollado una guía extensa

1. Identificar los objetivos del negocio.
2. Identificar lo que se desea saber o aprender.
3. Identificar los subojetivos.
4. Identificar las entidades y atributos relativos a esos subobjetivos.
5. Formalizar los objetivos del a medición,
6. Identificar preguntas que puedan cuantificarse y los indicadores relacionados que se vana usar para ayudar a conseguir los objetivos de medición
7. Identificar los elementos de datos que se vana recoger para construir los indicadores que ayuden a responder a las preguntas planteadas.
8. Definir las medidas a usar y hacer que estas definiciones sean operativas.
9. Identificar las acciones que serán tomadas para mejorar las medidas indicadas.
10. Preparar un plan para implementar estas medidas.

**Capitulo 24. Métricas Técnicas Para Sistemas Orientados a Objetos**

5 características que regulan las métricas especializadas:

* **Localización**: manera en que la información se concentra en un programa. En el contexto OO, la información se concentra por la encapsulacion de datos y procesos dentro del os limites de una clase u objeto. Las métricas deben aplicarse a la clase (objeto) como en una entidad completa
* **Encapsulacion**: empaquetamiento de una colección de elementos. Para los sistemas OO, la encapsulacion engloba las responsabilidades de una clase, incluyendo sus atributos y operaciones, y los estados del a clase, definidos por valores de atributos específicos.
* **Ocultamiento de información:** suprime los detalles operaciones de un componente de programa. Solo se proporciona la información necesaria para acceder al componente aquellos otros componentes que deseen acceder. Las métricas que proporcionan una indicación del grado de ocultación logrado suministran un indicio de la calidad del diseño OO
* **Herencia:** es un mecanismo que habilita las responsabilidades de un objeto, para propagarse a otros objetos. La herencia ocurre a través de todos los niveles de una jerarquía de clase.
* **Técnicas de abstracción de objetos**: la abstracción es un mecanismo que permite al diseñador concentrarse en los detalles esenciales de un componente de programa, prestando poca atención a los detalles de bajo nivel. Las métricas OO representan abstracciones en términos de mediciones de una clase.

**Métricas para el modelo de diseño OO:**

***Que características pueden medirse cuando se evalúa un diseño OO?***

9 características distintas y medibles de un diseño de OO:

* **Tamaño**: se define en 4 enfoques: población, volumen, longitud, y funcionalidad
* **Complejidad**: como se interrelacionan las clases de un diseño OO con otras.
* **Acoplamiento:** las conexiones físicas entre los elementos del diseño OO, representan el acoplamiento dentro de un sistema OO
* **Suficiencia:** grado en que una abstracción posee los rasgos mínimos necesarios, o el grado en que una componente de diseño posee características en su abstracción, desde el punto de vista de la aplicación actual.
* **Integridad:** la única diferencia entre integridad y suficiencia ese l conjunto de características, contra las que se comparan la abstracción o componente de diseño.
* **Cohesión:** la cohesión de una clase se determina examinando el grado en que el conjunto de propiedades que posee sea parte del diseño o dominio del problema
* **Originalidad:** es el grado en que una operación es atómica; esto quiere decir que la operación no puede ser construid afuera de una secuencia de otras operaciones contenidas dentro de una clase.
* **Similitud:** grado en que dos o mas clases son similares en términos de estructura, comportamiento, función o propósito
* **Volatilidad:** mil a probabilidad de que un cambio ocurra.

Estas métricas técnicas para sistemas OO pueden aplicarse no solo al modelo de diseño sino también al modelo de análisis.

**La serie de métricas CK:**

6 Métricas basadas en clases para sistemas OO

* **Métodos ponderados por clase (MPC):** asumen que n métodos de complejidad c1,c2…cn, se definen para la clase C
* **Arbol de profundidad de herencia (APH):** esta métrica se define como la máxima longitud del nodo a la raíz del árbol
* **Numero de descendientes(NDD):** las subclases inmediatamente subordinadas a una clase en la jerarquía de clases se denominan sus descendientes
* **Acoplamiento entre clases objeto (ACO):** el modelo CRC debe utilizarse para determinar el valor de ACO. Aco, es el numero de colaboraciones listadas para una clase, en la tarjeta CRC
* **Respuesta para una clase (RPC):** El conjunto de respuesta de una clase e suna serie de métodos que pueden potencialmente ser ejecutados, en respuesta a un mensaje recibido por un objeto, en la clase. RPC se define como el numero de métodos ene l conjunto de respuesta.
* **Carencia de cohesión en los métodos (CCM):** cada método dentro de una clase, C, accede a uno o mas atributos. CCM es el numero de métodos que accede a uno o mas de los mismos atributos

**Métricas propuestas por Lorenz y Kidd:**

Separan las métricas basadas en clases en 4 amplias categorías:

* Tamaño: se centran ene l recuento de atributos y operaciones para cada clase individual y los valores promedio para el sistema OO
* Herencia: se centran en la forma en que las operaciones se reutilizan en la jerarquía de clases
* Valores internos: examinan la cohesión y los aspectos orientados al código.
* Valores externos: examinan el acoplamiento y la reutilización

**Tamaños de clase (TC):** el tamaño general de una clase puede medirse determinando las siguientes medidas:

* El total de operaciones que se encapsulan dentro del a clase.
* El numero de atributos encapsulados por la clase.

Valores grandes para TC indican que la clase debe tener bastante responsabilidad.

Esto reducirá la reutilización del a clase y complicara la implementación y las pruebas.

**Numero de operaciones redefinidas para una subclase (NOR):**

Los valores grandes para el NOR, generalmente indican un problema de diseño. Si el NOR es grande, el diseñador ha violado la abstracción representada por la superclase. Esto provoca una débil jerarquía de clases y un software 00, que puede ser difícil de probar y modificar.

**Numero de operaciones añadidas por una subclase (NOA):**

La medida que el valor NO Ase incrementa, la subclases e alejad e la abstracción representada por la superclase

**Indice de especialización (IES):**

Proporciona una indicación aproximada del grado de especialización, para cada una del as subclases en un sistema OO. La especialización se puede alcanza añadiendo o eliminando operaciones, pero también redefiniendo.

**La colección de métricas MDOO:**

Proporcionan indicadores cuantitativos para el diseño de características OO.

***Métricas MDOO:***

* **Factor de herencia de métodos (FHM)** : El grado que en la arquitectura de clases de un sistema OO hace uso del a herencia tanto para métodos como atributos.
* **Factor de Acoplamiento (FA):** indicador de conexiones entre los elementos del diseño. El valor del FA crece, la complejidad del soft 00 también crece y la comprensión, el mantenimiento ye l potencial de reutilización , pueden resentirse como resultado.
* **Factor de polimorfismo (FP):** numero de métodos que redefinen métodos heredados, dividida por el máximo numero de posibles situaciones polimorficas distintas. FP es una medida indirecta del a cantidad relativa de ligadura dinámica en un sistema.

**Métricas orientadas a operaciones:**

**3 métricas**:

* **Tamaño medio de operación(TOmedio):** el número de mensajes enviados por la operación proporciona una alternativa para el tamaño de operación. A medida que el número de mensajes enviados por una sola operación se incrementan, es más probable que las responsabilidades no han sido correctamente asignadas dentro de la clase.
* **Complejidad de operación (CO):** Puede ser calculada usando cualquiera de las métricas de complejidad propuestas para el software convencional. El diseñador debería esforzarse por mantener la CO tan baja como sea posible.
* **Numero de parámetros de media por operación (NPmedia):** tan largo como sea el numero de parámetros de operación, mas compleja será la colaboración entre objetos. En general, NPmedia debe mantenerse tan baja como sea posible.

**Métricas para pruebas orientadas a objetos**

Las métricas se organizan en categorías, que reflejan características de diseño importantes.

**Encapsulacion:**

* **Carencia de Cohesión en métodos (CCM):** Cuanto mas alto sea el valor CCM será necesario probar mas estados para asegurar que los métodos no generan efectos colaterales
* **Porcentaje público y protegido (PPP):** Losa tributos públicos que se heredan de otras clases son visibles para esas clases. Los protegidos son una especialización y son privados a subclases específicas. Esta métrica indica el porcentaje de atributos de una clase que son públicos. Valores altos para PPP incrementan la probabilidad de efectos colaterales entre clases.
* **Acceso Publico a datos miembros** (APD): esta métrica indica el numero de clases que pueden acceder al os atributos de otras clases, una violación de encapsulacion. Valores altos para APD producen potencialmente efectos colaterales entre clases. Las pruebas deben diseñarse para estar seguros de que ese tipo de efectos colaterales serán descubiertos

**Herencia:**

* **Numero de Clases raíz (NCR):** esta métrica es un recuento del as distintas jerarquías de clases, que sed escriben ene l modelo de diseño. A medida que el NCR se incrementa, el esfuerzo de comprobación también se incrementa
* **Numero de Padres Directos (NPD):** cuando es utilizado ene l contexto OO, el NP Des una indicación de herencia múltiple. NPD>1 indica que la clase hereda sus atributos y operaciones de mas de una clase raíz. Se debe evitar que NPD>1 tanto como sea posible.
* **Numero de descendientes (NDD) y árbol de profundidad de herencia(APH):** las métricas definidas para la complejidad de clase, incluyen 3 métricas CK:
  + - **Métodos ponderados por clase (MPC)**
    - **Acoplamiento entre clases de objetos (ACO)**
    - **Respuesta para una clase (RPC)**

**Métricas para proyectos orientados a objetos:**

Aspecto clave al que debe hacer frente un jefe de proyecto durante la planificación, e suna estimación del tamaño de implementación del software. Las siguientes métricas pueden proporcionar una visión sobre el tamaño del soft:

* **Numero de escenario (NE):**  El numero de escenarios o casos de uso, es directamente proporcional al numero de clases requeridas para cubrir los requisitos, el numero de estados para cada clase, el numero de métodos , atributos y colaboraciones, el NE es un importante indicador del tamaño del programa
* **Numero de clases clave (NCC):** Una clase clave se centra directamente en el dominio del negocio para el problema, y tendrá una menor probabilidad de ser implementada por medio del a reutilización. Valores altos para NCC indican gran trabajo de desarrollo substancial
* **Numero de subsistemas (NSUB):**  El numero de subsistemas proporciona una visión sobre la asignación de recursos, la planificación ye l esfuerzo de integración global.

Las métricas NE, NCC, NSUB pueden recolectarse sobre proyectos 00 pasados, y están relacionados con el esfuerzo invertido ene l proyecto como un todo, y en actividades de procesos individuales. Colectivamente estas métricas pueden usarse para estimar el esfuerzo, duración, personal y otra información de proyecto para el proyecto actual.

**Capitulo 25. Gestión del Riesgo**

**Estrategias del Riesgo Rectivas y Proactivas**

La mayoría de los equipos de soft se apoya exclusivamente en las estrategias de riesgo reactivas. El equipo de soft no hace nada acerca de los riesgos hasta que algo sale mal.

Una estrategia considerablemente mas inteligente para la gestión del riesgo es ser proactivo. (comienza mucho antes de que comience el trabajo técnico. e identifican los riesgos potenciales, se valoran su probabilidad e impacto, y se les clasifica según su importancia.

El objetivo principal es evitar el riesgo. El riesgo siempre involucra 2 características:

* **Incertidumbre**: el riesgo puede o no ocurrir
* **Perdida**: si el riesgos e convierte en realidad, ocurrirán consecuencias o perdidas indeseables

**Riesgos del Proyecto**

Los riesgos del proyecto amenazan el plan del proyecto. Los riesgos del proyecto identifican potenciales problemas en presupuestos , calendarizacion, personal, recursos, participantes y requisitos, y su impactos obre un proyecto de software.

**Riesgos técnicos**

Amenazan la calidad y actualizada del software que se producirá. Los riesgos técnicos identifican potenciales problema sen diseño, implementación, interfaz, verificación y mantenimiento. También son factores de riesgo técnico la ambigüedad del a especificación, la incertidumbre técnica, la obsolescencia técnica y la tecnología de punta. Los riegos técnicos ocurren porque el problema es mas difícil de resolver del o que en un principio se pensó que serian.

**Riesgos de negocio**

Amenazan la viabilidad del software que se construirá. Estos riesgos con frecuencia ponen en peligro el proyecto o el producto.

Los candidatos para los 5 mayores riesgos de negocios son:

* Construcción de un producto o sistema excelente que en realidad nadie quiere
* La construcción de un producto que y ano encaja en la estrategia comercial global de la compañía
* La construcción de un producto que la fuerza ventas no sabe como vender
* La perdida dela poyo del os altos ejecutivos debido aun cambio ene l enfoque o ene l personal
* La perdida presupuestaria o del personal asignado

**Los riesgos predecibles se extrapolan de la experiencia con proyectos previos**

**Los riesgos impredecibles son el comodín del a baraja., pueden y de hecho ocurren pero son extremadamente difíciles de identificar con antelación**

7 principios del a gestión de riesgos

1. Mantenimiento de una perspectiva global
2. Tener una visión previsora
3. Alentar la comunicación abierta
4. Integración
5. Enfatizar un proceso continuo
6. Desarrollo de una visión conjunta del producto
7. Alentar el trabajo en equipo

**Identificación de Riesgos**

Existen 2 tipos distintos de riesgos para cada una el as categorías que se han presentado anteriormente

* **Riesgos genéricos**: amenaza potencial pata todo el proyecto de soft.
* **Riesgos específicos**: del producto los pueden identificar solo aquellos con un claro conocimiento del a tecnología, personal y el entorno especifico de soft que se construirá.

Los riesgos se pueden identificar mediante un método que consiste en una lista de verificación de riesgos, y pueden enfocarse en algún subconjunto de riesgos conocidos y predecible sen las siguientes subcategorías genéricas:

* **Tamaño del producto**
* **Impacto en e l negocio**: riesgos asociados con las restricciones que impone la gerencia o mercado
* **Características del cliente**
* **Definición del proceso:** riesgos asociados con el grado ene l que se ha definido el proceso de soft y en que le da seguimiento la organización que lo desarrolla.
* **Entorno de desarrollo**: riesgos asociados con la disponibilidad y calidad del as herramientas que se usaran en la construcción del producto
* **Tecnología que construir**: asociados con la complejidad del sistema que se construirá
* **Tamaño y experiencia del a planilla de personal:** asociados con la experiencia global técnica ye l proyecto del os ingenieros de soft que se harán el trabajo

**Evolución del riesgo global del proyecto**

Algunas preguntas que se basan en los datos de riesgo obtenido al entrevistar en distintas partes del mundo a gestores de proyecto de soft experimentados.

* *Los usuarios finales están comprometidos con el proyecto ye l sistema/producto que se construirá?*
* *Los usuarios finales tienen expectativas realistas?*
* *El ámbito del proyecto es estable?*
* *Los requisitos del proyecto son estables?*

Si las respuestas a algunas de estas y otras preguntas es negativa, se deben instituir sin demoras los pasos de :

* Reducción
* Supervisión
* Gestión

**Componentes y controladores del riesgo**

* **Riesgo de desempeño**: grado de incertidumbre de que el producto satisfaga los requisitos y se ajuste a luso que se pretende darle
* **Riesgo de costo:** grado de incertidumbre de que se mantenga el presupuesto del proyecto
* **Riesgo de soporte**: grado de incertidumbre de que el soft resultante será facil de corregir, adaptar y mejorar:
* **Riesgo de calendarizacion**: grado de incertidumbre de que se mantenga la calendarizacion del proyecto y de que el producto se entregue a tiempo

El impacto de cada controlador de riesgo sobre el componente de riesgo se divide en 4 categorías de impacto: Despreciable, marginal, critico o catastrófico.

**Evaluación del impacto:**

**Categorías:**

* **Catastrófico**: el fracaso en la satisfacción del os requisitos resultaría en un fracaso del a misión
* **Critico:** el fracaso para satisfacer los requisitos resultaría en un desempeño degradado del sistema hasta un punto donde el éxito del a misión es cuestionable
* **Marginal:** el fracaso para satisfacer los requisitos resultaría en degradación del a misión secundaria.
* **Despreciable:** el fracaso al satisfacer los requisitos crearía inconvenientes o impactos no operativos

\*Esto hace referencia: a la consecuencia potencial de errores o fallas de soft no detectados\*

**Componentes:**

* **Desempeño.**

Categoría-> **Catastrófico**: cierta reducción en el desempeño técnico

Categoría -> **Critica**: cierta reducción en el desempeño técnico

Categoría-> **Marginal:** Mínima o pequeña reducción en el desempeño técnico

Categoría**> Despreciable:** ninguna reducción en el desempeño técnico

* **Soporte.**

Categoría-> **Catastrófico**: Software que no responde o no se puede soportar

Categoría -> **Critica**: Demora menores en las modificaciones del software

Categoría-> **Marginal:** Respuesta de soporte de software

Categoría**> Despreciable:** Software que fácilmente se le da soporte

* **Costo**

Categoría-> **Catastrófico**: Recortes financieros significativos, probable superación del presupuesto

Categoría -> **Critica**: Cierto recorte de recursos financieros, posibles excesos

Categoría-> **Marginal:** Suficientes recursos financieros

Categoría**> Despreciable:** Posible superávit presupuestal

* **Calendarización**

Categoría-> **Catastrófico**: COI inalcanzable

Categoría -> **Critica**: Posible deslizamiento ene l COI

Categoría-> **Marginal:** Calendarización alcanzable y realista

Categoría**> Despreciable:** COI fácilmente alcanzable

**Proyección del riesgo**

La proyección del riesgo intenta clasificare l riesgo en 2 formas:

* La posibilidad o probabilidad de que el riesgo sea real
* Las consecuencias del os problemas asociados con el riesgo, en caso de que ocurra.

El planificador del proyecto, junto con otros gestores y personal técnico, realizan 4 pasos en la proyección del riesgo:

1. establecimiento de una escala que refleje la posibilidad percibida de un riesgo.
2. Delineado del as consecuencias del riesgo
3. Estimación del impacto del riesgo ene l proyecto y el producto.
4. Tomar nota del a precisión global del a proyección del riesgo de modo que no haya malas interpretaciones.

**Evaluación del impacto del riesgo:**

3 factores afectan las consecuencias que son probables si un riesgo ocurre:

* **Su naturaleza**: problemas que son probables si ocurre
* **Su ámbito**: combina la severidad con su distribución global
* **Tiempo:** el tiempo de un riesgo considera cuando y durante que periodo se sentirá el impacto

la exposición al riesgo global, ER se determina mediante la siguiente relación

ER= PXC, donde pe s la probabilidad de que ocurra un riesgo y C el costo al proyecto en caso de que ocurra el riesgo.

Compárese la ER de todos los riesgos con la estimación de costo para el proyecto, si la E Res mayor que 50 por ciento del costo del proyecto, la viabilidad del proyecto debe reevaluarse.

**Refinamiento del riesgo**

Cual es una buena formad e describir un riesgo?

Representar el riesgo en formato condición-transición-consecuencial. Es decir el riesgos e establece en la siguiente forma:

Dado que <condición>entonces existe una preocupación de que (posiblemente)<consecuencia>

**Reducción, Supervisión y Gestión del Riesgo:**

Una estrategia eficaz debe considerar tres temas:

Evitar el riesgo, supervisare l riesgo y gestionar el riesgo y los planes de contingencia.

**Que se puede hacer para reducir un riesgo?**

* Reunirse con el personal actual para determinar las causas de la movilidad.
* Reducir aquellas causas que se controlan antes de que comience el proyecto.
* Una vez iniciado el proyecto, suponer que la movilidad ocurrir ay entonces desarrollar técnicas que aseguren la continuidad cuando la gente se aleje.
* Organizar equipos de proyecto de modo que la información acerca de cada actividad de desarrollo se disperse con amplitud
* Definir estándares de documentación y establecer mecanismos que aseguren que los documentos se desarrollen en una forma oportuna.
* Llevar a cabo revisiones por pares de todo el trabajo
* Asignar un miembro de personal de respaldo por cada categoría critica

**En caso del a elevada tasad e movilidad del personal, se pueden supervisar los siguientes factores:**

* Actitud general del os miembros dele quipo con base en las presiones del proyecto
* El grado en el cual el equipo esta cuajado.
* Relaciones interpersonales entre los miembros del equipo
* Potenciales problemas con las compensaciones y los beneficios
* La disponibilidad de empleo dentro y fuera del a compañía

Si la ER par aun riesgo especifico es menor que el costo del a reducción del riesgo, no se intenta reducir el riesgo sino continuar supervisándolo

La experiencia indica que un 80 por ciento del riesgo del proyecto global puede explicarse solo con 20 por ciento del os riesgos identificados.

**Análisis de seguridad y peligros de software:** son actividades de aseguramiento de la calidad del soft, que se enfocan en la identificación y evaluación de los peligros potenciales que pedieran afectar al soft negativamente y provocar una falla en todo el sistema.

**El Plan RSDR**

Plan de reduccion, supervisión, y gestión del riesgo. Este plan documenta todo el trabajo realizado como parte del análisis del riesgo y gestor del proyecto lo emplea como parte del plan global del proyecto.

**La supervisión del riesgo es una actividad de seguimiento del proyecto con 3 objetivos:**

* Valorar si los riesgos predichos de hecho ocurren
* Asegurar que los pasos para evitar el riesgo definidos para este se están aplicando con propiedad
* Recopilar información que pueda usarse en futuros análisis de riesgo

**Capitulo 25. Gestión de la calidad**

La gestión del a calidad del soft e suna actividad protectora o de sombrilla que incorpora tanto control como aseguramiento del a calidad

La gestión de la calidad abarca:

* Un proceso de garantía de la calidad del soft
* Tareas especificas de aseguramiento y control de la calidad
* Practicas efectivas de ingeniería de software
* Control de todos los productos de trabajo de soft y los cambios que generan
* Un procedimiento para garantizar la concordancia con los estándares de desarrollo del software
* Mecanismos de medición e informe.

El control de la variación es la clave para un producto de alta calidad. En el contexto del software se lucha por controlar la variación en el proceso genérico que se aplica y el énfasis de calidad que pernea el trabajo de ingeniería de software

**Calidad:**

* **De diseño**: características que los diseñadores especifican para un elemento. En el desarrollo de software esta calidad incluye: requisitos, especificaciones y el diseño del sistema
* **De concordancia**: calidad de concordancia es el grado en el que las especificaciones de diseño se aplican durante la fabricación. La calidad de concordancia es un tema enfocado principalmente en la implementación.

**Control de calidad**

Este involucra la serie de inspecciones, revisiones y pruebas empleadas a lo largo del proceso de software para garantizar que cada producto de trabajo satisfaga los requisitos que se le han asignado.

**Garantía de calidad**

Consiste en un conjunto de funciones de auditoria e información que evalúan la efectividad y que tan completas son las actividades de control de calidad. La meta del aseguramiento de la calidad es brindarle al gestor los datos necesarios para que este informado acerca de la calidad del producto.

**Costo de la calidad**

El costo de la calidad incluye todos los costos que genera la búsqueda de calidad o demanda el desarrollo de las actividades relacionadas con la calidad

**Componentes del costo de calidad:**

* **Prevención**: incluyen planificación de la calidad, revisiones técnicas formales, equipo de pruebas y entrenamiento.
* **Evaluación:** incluyen actividades para comprender mejor la condición del producto la primera vez a través de cada proceso.
* **Fallas**: son aquellos que desaparecerían sin o aparecieran defectos antes de enviar un producto al os clientes.

**Se subdividen en:**

* **costos de fallas internas** : cuando se detecta un defecto ene l producto antes del envió., incluyen: reelaboración, reparación y análisis en modo de falla
* **Costos de fallas externas**: se asocian con defectos detectados después de que el producto ha sido enviado al cliente

**Garantía del a calidad del software (SQA)**

*Como se define la calidad del software?*

1. los requisitos del software son la base del as medidas del a calidad. La faltad e concordancia con los requisitos e suna faltad e calidad
2. los estándares especificados definen un conjunto de criterios de software que guían la forma en que el software se elabora. Si no se siguen los criterios casi seguramente resultara una falta de calidad.
3. Con frecuencia no se menciona un conjunto de requisitos implícito. Si el software concuerda con sus requisitos explícitos pero facrasa al satisfacer los requisitos implícitos, su calidad esta en duda.

**Actividades de SQA**

Grupo SQA: tiene la responsabilidad de planificar, supervisar, guardar registros, analizar y reportar la garantía de la calidad.

La misión del SQA es auxiliar al equipo de software a conseguir un producto final de alta calidad.

*Cual es el papel de un grupo de SQA?*

* Preparar un plan de SQA para un proyecto.
* Participar en el desarrollo de la descripción del proceso de software del proyecto
* Revistar las actividades de ingeniería del software para verificar que se ajuste al proceso de soft definido.
* Audita productos de trabajo de software seleccionados para verificar que se ajusten con los definidos como parte del proceso de software.
* Garantiza que las desviaciones en el trabajo de soft y en los productos de trabajo estén documentados y se manejen de acuerdo con el procedimiento establecido.
* Registra cualquier falta de ajuste y lo informa al gestor ejecutivo.

**Revisiones del software**

Las revisiones del soft son un filtro para el proceso de software, sirven para descubrir errores y defectos que luego pueden eliminarse. Las revisiones del software purifican las actividades de ingeniería del soft que se han denominado análisis, diseño y codificación

Revisión técnica formal: es el filtro mas efectivo desde el punto de vista de aseguramiento del a calidad. Dirigida por los ingenieros de software. La rtf es un medio efectivo para descubrir errores y mejorar la calidad del software.

**Revisión técnicas formales**

*El objetivo principal de una* RTF es encontrar los errores antes de que pasen a otra actividad de ingeniería de soft o sean liberado al usuario final

Una RTF se enfoca en una porción relativamente pequeña de un producto de trabajo

En algunas situaciones es buena idea hacer que alguien distinto al productor recorra el producto que experimenta revisión, esto conduce a una interpretación literal del producto de trabajo y aun mejor reconocimiento de los errores.

**Informe de la revisión y conservación de registros**

Se genera una lista de problemas de revision (cumple 2 propósitos: identificar áreas con problema en el producto, funcionar como lista de verificación de elementos de acción que guían al productor conforme se hacen las correcciones). Se llena un informe resumen del a revisión técnica formal. Un informe resumen de la revisión responde 3 preguntas

* Que se reviso?
* Quién lo reviso?
* Cuales fueron los hallazgos y conclusiones?

**Directrices para las revisiones técnicas formales:**

1. revisar el producto, no al productor.
2. Establecer una agenda y respetarla
3. Limitar el debate y la impugnación
4. Enunciar áreas de problemas, pero no se intente resolver todos los que se hayan señalado.
5. Tomar notas.
6. Limitar el número de participantes e insistir en la preparación anticipada.
7. Desarrollar una lista de verificación para cada producto que tenga probabilidad de ser revisado.
8. Asignar recursos y programar las RTF
9. Realizar un entrenamiento significativo de todos los revisores
10. Analizar las revisiones previas

**Revisiones basadas en muestras**

Las revisiones toman tiempo, pero es tiempo bien empleado, sin embargo, si el tiempo es corto y no se tiene otra opción, no se dispensen las revisiones. En su lugar utilícense revisiones basada sen muestras. Para ser eficaz el proceso basados en muestras debe intentar cuantificar aquellos productos de trabajo que sean objetivos principales para las RTF completas.

**Garantía del a calidad estática del software**

Para el soft, la garantía del a calidad estadística implica los siguientes pasos:

1. Información acerca de los defectos de software se recopila y clasifica
2. Se intenta determinar la causa subyacente de cada defecto
3. Mediante el principio de pareto(80 por ciento del os defectos se encuentran por ciento de todas las causas posibles) se aísla un 20 por ciento(los vitales)
4. Una vez que las causas vitales han sido identificadas, se corrigen los problemas que han provocado los defectos.

**Seis sigma para ingeniería del software**

Seis sigma es la estrategia mas ampliamente empleada en la actualidad para aseguramiento de calidad estadístico en la industria. Es una metodología rigurosa y disciplinada que utiliza análisis de datos y estadístico para medir y mejorare l desempeño operativo de una compañía al identificar y eliminar los defectos en la fabricación y los procesos relacionados con el servicio.

La metodología seis sigma define 3 pasos centrales:

1. Definir los requi8sitos del cliente, entregables y metas del proyecto por medio de métodos bien definidos de comunicación con el cliente.
2. Medir el proceso existente y su salida para determinar el desempeño de calidad actual
3. Analizar las métricas de defecto y determinar las causas poco vitales.

Si el proceso de soft existente esta en marcha, pero se requiere mejorarlo 6 sigma sugiere dos pasos adicionales:

1. mejorar el proceso eliminando las causas originales del os defectos
2. controlar el proceso para agilizar que el trabajo futuro no vuelva a introducir las causas de defectos.

**Método DMACM**: definir, medir, analizar, mejorar y controlar.

Si una organización esta desarrollando un proceso de soft, (en lugar de mejorar un existente), los pasos centrales aumentan del a siguiente manera:

Diseñar el proceso para:

1. evitar las causas originales del os defectos
2. satisfacer los requisitos del cliente

verificar que el modelo de proceso, de hecho, evitara los defectos y satisfará los requisitos del cliente

a esta versiones a veces se la llama **método DMADV**(definir, medir, analizar, diseñar y verificar)

**Fiabilidad del software**

Se define en términos estadísticos como la probabilidad del a operación libre de fallas de un programa de computadora en un entorno especifico durante un tiempo especifico.

***Que significa el término falla?***

La falla es la falta de concordancia con los requisitos del software. Las fallas solo pueden ser molestas y catastróficas

**Medidas de fiabilidad y disponibilidad**

Los problemas de confiabilidad software casi siempre pueden seguirse a defectos en diseño o implementación-

Una simple medida de fiabilidad ese l tiempo medio entre fallas (TMEF), donde:

TMEF= TMDF + TMDR

Tmd y tmdr: tiempo medio de falla y tiempo medio de reparación, respectivamente

**Seguridad del software**

Es una actividad de aseguramiento del a calidad del software que se enfoca en la identificación y evaluación del os peligros potenciales que pueden afectar negativamente al soft y provocar una fallad e todo el sistema

La confiabilidad del soft utiliza análisis estadístico para determinar la probabilidad de que ocurriera una falla del soft, sin embargo el hecho de que ocurra una falla, no necesariamente resulta en un peligro o percance. La seguridad del soft examina las formas en las cuales las fallas resultan en condiciones que pueden conducir aun percance.

**Los estándares de calidad ISO 9000**

Los sistemas de garantía de calidad fueron creados para ayudar a las organizaciones a garantizar que sus productos y servicios satifacen las expectativas del os cliente sal cumplir sus especificaciones.

ISO 9000 describe l oque se debe hacer para ser manejable, pero no describe como se debe hacer.

**El plan de SQA**

Este plan proporciona un mapa para instituir la garantía de la calidad del soft. Desarrollado por el grupo de SQA, el plan funciona como plantilla para las actividades de SQA que se instituyan para cada proyecto de soft.

En el IEE Ese publico un estándar par aplanes de SQA, que recomienda una estructura que identifica:

1. el propósito y el ámbito del plan
2. una descripción de todos los productos de trabajo de ingeniería de soft
3. todos los estándares y practicas aplicables que se aprovechan durante el proceso de soft
4. acciones y tareas de SQA
5. herramientas y métodos que soportan las acciones y atareas de SQA
6. procedimientos de gestión de configuración de soft para gestionar el cambio
7. métodos para ensamblar, salvaguardar y mantener todos los registros realacionados con el SQA
8. papeles y responsabilidades en la organización relativas a la calidad del producto.

**A modo de resumen:**

El aseguramiento del a calidad del software ese l mapeo (corrección) del os preceptos gerenciales y las disciplinas de diseño del a garantía de calidad en el espacio gerencial y tecnológico aplicable del a ingeniería del software

**Capitulo 27. Gestión del Cambio**

la gestión de la configuración del software (GCS O GC), es una actividad protectora (sombrilla) que se aplica a los largo del proceso de software. Las actividades GCS se desarrollan para

1. identificar el cambio
2. controlar el cambio
3. garantizar que el cambio se implementara de manera adecuada
4. reportar los cambios a otros que pudieran estar intereados

es importante distinguir entre soporte de software y gestión de la configuración del software.

El soporte es un conjunto de actividades de ingeniería de soft que ocurren después de que este se ha entregado al cliente y que fue puesto en operación.

La gestión de la configuración del software es un conjunto de actividades de seguimiento y control que se inician cuando comienza un proyecto de ingeniería de software y terminan solo cuando este se retira de operación.

Una meta primordial de la ingeniería del software es mejorar la facilidad con la que los cambios se pueden acomodar y reducir el esfuerzo cuando los cambios se deben realizar.

**Gestión de la configuración del software**

Los elementos que comprenden la información producida como parte del proceso de software se denomina colectivamente configuración del software

**Cual es el origen de los cambios que se requieren para el software?**

* Nuevas condiciones en el negocio o mercado dictan los cambios en los requisitos del producto o las reglas del negocio.
* Nuevas necesidades del cliente demandan la modificación de los datos que producen los sistemas de información, de la funcionalidad que entregan los productos o los servicios que entrega un sistema basado en computadora.
* La reorganización o el crecimiento o reducción del negocio provocan cambios en las prioridades del proyecto o en la estructura del equipo de ingeniería del software
* Restricciones presupuestales o de calendarización inducen una redefinición del sistema o producto.

La GCS considera como una actividad de aseguramiento de la calidad del software que se aplica a lo largo del proceso respectivo.

**Un Escenario GCS**

Debe haber un mecanismo para garantizar que los cambios simultaneos al mismo componente son seguidos, gestionados y ejecutados adecuadamente.

Un sistema de GC utilizado en este escenario apoyaría todas estas funciones y tareas, esto es, las funciones determinan la funcionalidad requerida de un sistema GC. El gestor del proyecto ve una GC como un mecanismo de auditoria. El gestor de configuración como un mecanismo de creación de control, seguimiento y políticas, el ingeniero de software como un mecanismo de control del cambio, la construcción y el acceso; y el usuario como un mecanismo de garantía de la calidad

**Elementos de un sistema de gestión de la configuración**

4 importantes elementos que deben estar presentes cuando se desarrolla un sistema de gestión de configuración

* Elementos componentes
* Elementos de proceso
* Elementos de construcción
* Elementos humanos

**Linea base**

Una línea base es un concepto de gestión de la configuración del software que ayuda a controlar el cambio sin impedir seriamente el cambio justificable

Una vez establecida una línea base, los cambios se pueden realizar, pero se debe aplicar un procedimiento especifico formal para evaluar y verificar cada uno.

En el contexto de la ingeniera del software, una línea base es un hito en el desarrollo del software. Se marca una línea base para la entrega de uno o mas elementos de configuración del software que se han aprobado como consecuencia de una revisión tecnica formal.

Un producto de trabajo de ingeniería del soft se convierte en una línea base solo después de que se ha revisado y aprobado.

**Elementos de configuración de software**

Un elemento de configuración de software (ECS) es informacion que se crea como parte del proceso de ingeniería del software. De manera mas realista un ECS es un documento, un conjunto completo de casos de prueba o un componente de un programa dado.

Los ECS están organizados para formar objetos de configuración susceptibles de catalogar en la base de datos del proyecto con un solo nombre.

Un objeto de configuración tiene un nombre, atributo y esta conectado con otros objetos por medio de relaciones. Flecha curva, indica realacion de composición. Flecha recta con doble punta indica una interrelacion.

**El deposito de ECS**

En la actualidad los ECS se conservan en una base de datos o deposito del proyecto. El diccionario webster define la palabra deposito como cualquier cosa o persona que se considera como centro de acumulación o almacenamiento.

Hoy el deposito es una base de datos que actua como el centro tanto de la acumulación como de almacenamiento de la informacion de ingeniería del software. El papel de persona es interactuar con el deposito mediante las herramientas que tiene integrada.

**El papel del deposito**

El deposito de de ECS es el conjunto de mecanismos y estructuras de datos que permite que un equipo de soft maneje el cambio en una forma eficaz. El deposito proporciona las funciones obvias de un sistema de gestión de base de datos, pero además el deposito realiza o impulsa las siguientes funciones:

* Integridad de los datos
* El compartir informacion
* Integración de herramientas
* El fortalecimiento de la metodología
* Estandarización de los documentos

**Contenido del depósito**

* **Contenido del negocio:** reglas del negocio, funciones del negocio, estructura de la organización, arquitectura de información
* **Contenido de gestión del proyecto:** estimaciones del proyecto, calendarización del proyecto, requisitos de GCS, requisitos de SQA, informes de proyecto, de auditoria, métricas del proyecto.
* **Contenido de modelo:** casos de uso, modelo de análisis, modelo de diseño, métricas técnicas
* **Contenido de construcción:** código fuente, código de objeto, instrucciones de construcción del sistema
* **Contenido V y V:** casos de prueba, guiones de prueba, resultados de prueba, métricas de calidad
* **Documentos:** plan del proyecto, plan GCS/SQA, especificaciones del sistema, especificaciones de requisitos, plan y procedimiento de prueba, documentos de soporte, manual del usuario.

**Características de la GCS**

El apoyo a la GCS requiere que el almacen o el deposito tenga un conjunto de herramientas que ofrezca soporte para las siguientes características

* Versiones
* Gestión del seguimiento de la dependencia y del cambio
* Seguimiento de requisitos
* Gestión de la configuración
* Rutas de auditoria

**El proceso de gestión de la configuración del software define una serie de tareas que tiene 4 objetivos principales:**

* Identificar todos los elementos que colectivamente definen la configuración del software
* Gestionar los cambios a uno o mas de dichos elementos
* Facilitar la construcción de diferentes versiones de una aplica
* Garantizar que la calidad del soft se conserva conforme la configuración evoluciona a lo largo del tiempo.

**5 tareas de la GCS (capas del proceso):**

Identificación

Control de versión

Control del cambio

Auditoria de la configuración

Informe

**Identificación de objetos en la configuración del software**

Es posible identificar 2 tipos de objetos

* **Básicos:** unidad de información creada por un ingeniero de software durante el análisis, el diseño, el código o las pruebas.
* **Agregado:** es una colección de objetos de objetos básicos y otros objetos agregados.

las interrelaciones establecidas para los objetos de configuración permiten que un ingeniero de software evalue el impacto del cambio.

**Control de la versión**

Un sistema de control de la versión implementa 4 grandes capacidades:

1. Una base de datos del proyecto(deposito) que guarda todos los objetos de configuración relevantes
2. Una capacidad de gestión de versión que almacena todas las versiones de un objeto de configuración
3. Una facilidad de hechura que permita al ingeniero de software recopilar todos los objetos de configuración relevantes y construir una versión especifica del software.

(la 4ta no la encontré je)

**El modelo de sistema contiene:**

1. una planilla que incluye una jerarquía de componentes y un orden de construcción para los componentes que describe como se debe construir el sistema.
2. Reglas de construcción
3. Reglas de verificación

**El proceso de control de cambio**

Se reconoce la necesidad del cambio -> solicitud de cambio por parte del usuario -> evaluación del desarrollador-> se genera informe de cambio-> la autoridad de control del cambio decide:

Solicitud se pone en fila para la acción se genera OCI->asignación de individuos para los objetos de configuración-> salida de objetos de configuración-> se hace el cambio-> entrada de los elementos de configuración que han cambiado -> establecimiento de un alinea base para pruebas -> realización de actividades de garantía de calidad y pruebas -> promoción del os cambios para incluirlos en la siguiente liberación -> reconstrucción del a versión apropiada del software -> revisión del cambio para todos los elementos de configuración -> inclusión de los cambios en la nueva versión -> distribución del a nueva versión

Se nieva la solicitud de cambio-> se informa al user

**Auditoria del a configuración**

La identificación , el control del a versión ye l control del cambio ayudan al desarrollador del soft a mantener el ordene n l oque de otro modo seria una situación caótica e inestable

Como se puede garantizar que el cambios e ha implementado con propiedad?

1. revisión técnicas formales
2. auditoria del a configuración del software

**Informe de Estado**

El informe de estado del a configuración e suna tarea de GCS que responde las siguientes preguntas .

1. que ocurrió?
2. quien lo hizo?
3. Cuando ocurrió?
4. Que otra cosa será afectada?

**Gestión de la configuración para la ingeniería web**

Se deben considerar 4 temas cuando se desarrollan tácticas para la gestión del a configuración del a webapp:

* Contenido
* Personal
* Escalabilidad
* Políticas

**Objetivos de configurar WEBAPP**

la webapp abarca una amplia gama de objetos de configuración: objetos de contenido, componentes funcionales y objetivos de interfaz.

**Gestión del contenido**

Se relaciona con la gestión de la configuración en el sentido en que un sistema de gestión del contenido SGC establece un proceso que adquiere contenido existente, los estructura en una forma que permite presentarlos aun usuario final y los ofrece al entorno del lado del cliente para su despliegue.

Un SGC configura el contenido para el usuario final al innovar 3 subsistemas integrados: de colección, de gestión y de publicación

El subsistema de colección abarca todas las acciones que se requieren para crear, adquirir o convertir el contenido en una forma que se pueda presentare n el lado del cliente

El subsistema de gestión implemente un deposito para todo el contenido. La gestión del a configuración se lleva a cabo dentro de este subsistema.

Implemente este subsistema un deposito que abarca los siguientes elementos:

* Base de datos contenido
* Capacidades de las bases de datos
* Funciones de gestión del a configuración

El subsistema de publicación obtiene contenido del deposito y lo entrega a los navegadores en el lado del cliente.

Este subsistema logra sus tareas mediante una serie de plantillas, cada plantilla es una función que construye una publicación empleando uno de 3 componentes distintos:

* Elementos estáticos
* Servicios de publicación
* Servicios externos

**Gestión del cambio**

Una gestión de cambio eficaz requiere modificar el proceso de control de cambios convencionales. Cada cambio se debe clasificare n una de cuatro clases

Clase 1: cambio de contenido o función que corrija un error o mejore el contenido o funcionalidad locales

Clase 2: una cambio de contenido o función que tenga impactos obre otros objetos de contenido o componentes funcionales

Clase 3: un cambio de contenido o función que tenga amplio impacto a través de una webapp

Clase 4: un gran cambio de diseño

**Control de la versión**

Para no sobrescribir por completo el trabajo de otro desarrollador , se debe establecer un proceso de control del a versión

1. se debe establecer un deposito central para el proyecto webapp
2. cada ingeniero web crea su propia carpeta de trabajo
3. los relojes en las estaciones de trabajo de todos los desarrolladores deben estar sincronizados
4. conforme se desarrollan nuevos objetos de configuración o se cambian los objetos existentes se importan al deposito central
5. conforme con los objetos se importan al o exportan del deposito se elabora un mensaje automático de registro cronometrado