1

Visión general de TSPi

El software industrial es desarrollado por equipos porque son más efectivos. Son capaces de llevar planes de proyecto, proceso común de trabajo y una efectiva comunicación que con llevan a productos de alta calidad.

1.1 ¿Qué es TSPi?

TPSi se define como una herramienta de trabajo o como un curso de alto nivel de licenciatura en la ingeniería de software de equipo. Se enfatiza en balancear el proceso, producto, trabajo en equipo y aprovecharse de la experiencia en la industria in la planeación y manejo de proyectos de software. TSPi guía al equipo paso a paso, y le muestra cómo aplicar la ingeniería de software y sus principales procesos en el ambiente de trabajo de equipo.

Por qué los equipos de ingeniería necesitan un proceso

Reunir un grupo de personas no produce automáticamente un equipo. Los pasos requeridos para construir un equipo no son evidentes; requieren de mecanismos. Los integrantes de un equipo deben definir cómo trabajar juntos y para ello definir una estrategia de trabajo. Ellos deben asignar las tareas entre los miembros del equipo, coordinada cada una de éstas y reportar los progresos.

1.2 Principios de TSPi

1. El aprendizaje es más efectivo cuando se sigue un proceso definido y se consigue una rápida retroalimentación. Los scripts y formularios de TSPi proveen una definición moderada y un marco de referencia repetible para el trabajo de ingeniería en equipo. TSPi provee un rápido rendimiento en la retroalimentación porque produce en forzosos ciclos cortos de desarrollo y resultados de evaluaciones por cada ciclo.
2. El productivo trabajo en equipo requiere una combinación de metas específicas, un en un ambiente de trabajo colaborativo.
3. Cuanto se ha luchado con los problemas de un actual proyecto y se ha seguido una guía efectiva de soluciones, se apreciarán los beneficios de las buenas prácticas de desarrollo.
4. La instrucción es más efectiva cuando se construye bajo un cuerpo de conocimiento previo. Se obtiene una experiencia más gratificante con equipos de software y cusos de equipos de software.

1.3 Diseño de TPSi

1. Provee un simple marco de referencia que se basa en PSP.

2. Desarrollo de productos en varios ciclos.
3. Establece medidas estándar para la calidad y el desempeño.
4. Provee medidas precisas para equipos y estudiantes.
5. Hace uso de roles y equipos de evaluación.
6. Requiere procesos de disciplina.
7. Provee una guía para los trabajos de equipo.

1.4 Flujo y estructura de TSPi

TSPi cuenta con varios ciclos. El primer ciclo inicia con el lanzamiento, en el cual el instructor describe los objetivos generales del producto. Posteriormente, el equipo sigue a través del TSPi en siete pasos: estrategia, planeación, requerimientos, diseño, implementación, pruebas y postmortem. En el ciclo 2 los ingenieros repiten los mismos pasos, esta vez orientados hacia la mejora del producto producido en el ciclo 1.

Estrategia de desarrollo cíclico

Cuando se inicia una estrategia de desarrollo cíclico, lo mejor es comenzar con la versión vital mínima de producto. Al decidir el tamaño y el contenido de cada ciclo, se deben considerar las siguientes limitaciones.

1. Cada ciclo debe producir una versión probable que es un subconjunto apropiado del producto final.
2. Cada ciclo debe ser lo suficientemente pequeño como para ser fácilmente desarrollo y probado en el tiempo disponible.
3. Cuando se combinan, los productos del ciclo deben producir el producto final deseado.

El TSPi comienza por tener equipos que producen una estrategia de desarrollo. Para empezar, recogiendo la base más pequeña razonable para desarrollar durante el primer ciclo. Luego estimar los tamaños de las funciones que se planean añadir en cada ciclo siguiente. Este enfoque garantiza que se completan subconjuntos del trabajo inicial del producto.

1.5 El proceso de TSPi

La secuencia de comandos de la tabla 1.1 muestra el flujo general de TSPi. Cada paso de guion es adoptado por una o más detalladas secuencias de comandos. Todos los guiones de TSPi se incluyen también en el apéndice D. Cada secuencia de comandos se inicia con una breve exposición de la finalidad general de la actividad (por ejemplo desarrollar un documento de requerimientos, producir un diseño o llevar a cabo una prueba). Cada script tiene también criterios de entrada y de salida.

1.6 La Estructura de libros de texto y de flujo

Es importante leer el libro y seguir los pasos especificados en el documento DEV de la tabla 1.1.

1.7 Resumen

Los principios básicos de TSPi son:

1. El aprendizaje es más eficaz cuando los estudiantes siguen los pasos definidos y repetibles y obtener una rápida retroalimentación sobre su trabajo.

2. el trabajo en equipo productivo requiere un objetivo definido el equipo, un entorno de trabajo eficaz y capaz de entrenamiento y liderazgo.

3. Cuando los estudiantes están expuestos a los problemas de los proyectos de desarrollo realistas y luego guiados a soluciones eficaces, que obtienen una mejor apreciación del valor de la ingeniería de sonido.

4. La instrucción es la masa efectiva cuando se basa en el cuerpo a disposición de la ingeniería previa, científica, y la experiencia pedagógica.

A partir de estos cuatro principios, el diseño TSPI implica siete opciones.

l. Proporcionar un marco simple que se basa en el fundamento de la PSP.

2. Desarrollo de productos en varios ciclos.

3. Establecer las medidas estándares de calidad y rendimiento.

4. Proporcionar las medidas precisas para que los equipos y los estudiantes.

S. Use papel y equipo de evaluaciones,

6. Requerir disciplina de proceso.

7. Proporcionar orientación sobre problemas de trabajo en equipo.

El proceso TSPi sigue una estrategia de desarrollo cíclico. Al comenzar un pequeño conjunto de funciones iniciales, el equipo puede completar rápidamente la primera versión del trabajo para posteriormente mejorar la planificación y desarrollo durante el segundo ciclo. Si hay tiempo, un tercer ciclo permite que el aprendizaje se refuerce aún más. La estrategia de desarrollo cíclico es muy similar a los procesos seguidos por los grupos de desarrollo de software a gran escala con éxito.

2

La lógica del proceso de trabajo en equipo

2.1 Por qué los proyectos fallan

Cuando los proyectos fallan, generalmente es porque los problemas de trabajo en equipo y no por cuestiones técnicas. PAG 41

2.2 Problemas comunes de equipo

Estudios demuestran que los más comunes problemas conciernen al liderazgo, la cooperación, participación, procastinación, calidad y evaluación.

2.3 ¿Qué es un equipo?

Un equipo consta de:

a. Al menos dos personas, que
b. Trabajan hacia un objetivo/meta/misión común, donde
c. A cada persona le ha sido asignada un rol específico o funciones a realizar, y donde
d. La finalización de la misión requiere de la dependencia entre los miembros del equipo.

Tamaño del equipo

Los equipos pueden ser de casi cualquier tamaño, partiendo de dos. Sin embargo, en situaciones prácticas, los equipos son más efectivos cuando desarrollan relaciones con casi todos los miembros. El tamaño del equipo lo suelen definir las industrias, es decir, la organización donde se construya; para proyectos grandes, se construyen equipos de hasta 20 personas, cada una de ellas trabajando bajo la supervisión de un gerente o supervisor.

En cuanto a los estudiantes, los equipos suelen tener entre cuatro y doce miembros, dependiendo la dificultad de los proyectos a desarrollar. La experiencia permite afirmar que los equipos entre cuatro y ocho miembros suelen ser más efectivos.

2.4 Construyendo equipos efectivos

Para construir equipos efectivos, se necesitan los tipos adecuados de tareas y condiciones de trabajo. El equipo debe tener un alto compromiso y un ambiente de trabajo que soporte dicho trabajo en equipo. Los miembros del equipo requieren de cohesión, metas, realimentación, y comúnmente, un marco de trabajo.

2.5 Cómo desarrollan los equipos

Algunos desarrollos de equipo pueden ocurrir por suerte, o pueden resultar de un conciso proceso de desarrollo en equipo. Este trabajo consiste en iniciar con tareas individuales, de tal modo que cada miembro tiene unas metas definidas. Como equipo, los integrantes aceptan las metas comunes con una especial significancia. La asignación de las metas individuales puede llevarse a cabo en forma aleatoria o de acuerdo a la propuesta del equipo mismo.

2.6 Cómo construir equipos TSPi

Grupos pequeños puede conllevar a equipos más efectivos. Las técnicas de ayuda a equipos permiten construir relaciones interpersonales que ayudan al trabajo conjunto y al soporte mutuo. Las guías de TSPi proporcionan os pasos de construcción de equipo para definir las metas, roles, establecer planes y mantener la comunicación entre los miembros.

2.7 Resumen

En este capítulo se presenta una visión general y la lógica de TSPi. Cubre lo que son los equipos, lo que se hace en el trabajo en equipo y cómo se resuelven problemas en equipo. El TSPi está diseñado para aprovechar las ventajas naturales de los equipos y reducir al mínimo la probabilidad o el impacto de las debilidades de los equipos.

Un problema significativo de las personas se refiere a la capacidad de manejar las presiones del cronograma. El TSPi ayuda a que los equipos manejen dichas presiones mediante el desarrollo de planes detallados y proporcionando un proceso de desarrollo estructurado.

El secreto de los equipos exitosos, es la construcción de una misión clara, distinta y con alto control de tareas. El entorno también apoya al buen desempeño del equipo; a animar a los miembros a planificar el trabajo y permitirles mantener una disciplina personal.

Cuando los equipos se unen, pueden conllevar a resultados extraordinarios. Definen sus propios procesos, establecen y trabajan bajo un plan común. Mantener la energía y el entusiasmo también ayuda a tener equipos verdaderamente excepcionales.

3

Lanzamiento de un proyecto de equipo

3.1 ¿Por qué dirigir un lanzamiento de equipo?

Los equipos necesitan establecer sus relaciones de trabajo, determinar los roles, y añadir sus metas. El lanzamiento es el primer paso, y se estima un tiempo de una hora para lograrlo.

El instructor es quien toma la decisión final en la asignación de roles.

3.2 Metas de equipo

1. Producir un producto de calidad

1.1 % defectos encontrados dentro de la primera compilación 80%

1.2 Número de defectos encontrados en la prueba de sistema 0

1.3 Funciones de requerimientos para cuando el proyecto esté completo 100%

2. Correr un proyecto productivo y bien manejado

2.1 Error en el estimado del tamaño del producto < 20%

2.2 Error en las horas estimadas de desarrollo < 20%

2.3 % Información registrada en el cuadero del proyecto 100%

3. Terminar a tiempo

3.1 Días antes o después de completar el desarrollo. < 4.

3.3 Metas de miembros-equipo

1. Ser un miembro efectivo y cooperativo

1.1 La evaluación PEER promedio del rol para ayuda y soporte debe ser mayor a 3.

1.2 La evaluación PEER promedio para la contribución general debe ser mayor a 3.

2. Hacer traabajo personal disciplinado de manera consitente.

2.1 % de información personal registrada y en el cuaderno del proyeto 100%

2.2 Porcentaje de semanas para completar WEEK 100%

3. Planear y rastrear todo el trabajo personal

3.1 Porcentaje de información peronal del proyecto registrada en SUMP y SUMQ 100%

3.2 Porcentaje de tareas de proyecto con información planeada y real completa en el formato TASK 100%

4. Producir productos de calidad (INS LOGTEST)

4.1 % promedio de defectos encontrados antes de la primera compilación > 70%

4.2 Densidad de defectos encontrada durante la compilación < 10KLOC

4.3 Densidad de defectos encontrada durante la prueba de unidad < 5KLOC

4.4 Densidad de defectos encontrada después de la prueba de unidad 0

3.4 Metas de rol

Líder de equipo

1. Construir y mantener un equipo efectivo

2. Motivar a todos los miembros del equipo a participar activamente en el proyecto

3. Resolver todos los problemas que presenten los integrantes del problema del equipo. Los que exponen ellos

4. Mantener al instructor informado acerca del progreso del equipo

5. Desempeñarse efectivamente como facilitador de las reuniones de equipo

Gerente de desarrollo (producir productos funcionales)

1. Producir producto superior

2. Utilizar por completo las habilidades de los integrantes

Gerente de planeación (Guiar al equipo en hacer un plan detallado)

1.Producir un plan completo, preciso y exacto, para el equipo y cada integrante del equipo

2. Reportar de manera exacta el estatus semanal del equipo

Gerente de calidad/proceso (asegurar que el equipo use TSP para producir un producto libre de defectos)

1. Todos los integrantes del equipo reportan de manera precisa y utilizan apropiadamente la información de TSPi

2. El equipo sigue fielmente TSPi y produce un producto de calidad.

3. Todas las inspecciones de equipo son apropiadamente reportadas y moderadas.

4. Todas las juntas de equipo se reportan de manera precisa y los reportes se almacenan en el cuaderno del proyecto.

Gerente de soporte (Asegurar eque el proyecto es apropiadamente controlado)

1. El equipo tiene herramientas adecuadas y métodos para soportar el trabajo.

2. No se hacen cambios no autorizados a los productos base

3. Todos los riesgos y problemas del equipo se registran en el lanzamiento de problemas y se reportan cada semana

4. El equipo cumple sus metas de reúso para el ciclo de desarrollo.

Responsabilidades	Líder de equipo	Gerente de desarrollo	Gerente de planeación	Gerente de proceso/calidad	Gerente de requerimientos/soporte
Construir y mantener un equipo eficaz	x				
Resolver los problemas entre los miembros del equipo	x				
Rastrear y reportar el progreso del equipo	x				
Actuar como facilitador de la reunión	x				
Interface con el instructor	x				
Mantener el cuaderno de proyectos	x				
Ayudar al equipo asignar tareas	x				
Conducir el trabajo de desarrollo		x			
Equipo de planificación de plomo y seguimiento de los progresos			x		
la planificación de entrega de calidad y seguimiento				x	
Proporcionar apoyo proceso de equipo				x	
Mantener los estándares del equipo y glosario				x	
Manejar el cumplimiento de la presentación de informes				x	
Alertar al equipo a problemas de calidad				x	
Obtener herramientas y el apoyo necesarios					x
gestión de la configuración mango					x
Dirigir la Junta de Control de Cambios					x
Actuar como defensor de equipo de reutilización					x
Manejo de emisión y seguimiento de riesgos					x
Mantener el sistema de glosario					x
Desarrollar el producto	x	x	x	x	x
Hacer planes personales	x	x	x	x	x
Realizar un seguimiento del trabajo personal	x	x	x	x	x
Producir productos de calidad	x	x	x	x	x
Siga las prácticas personales disciplinados	x	x	x	x	x

3.5 Lanzamiento de scripts TSPi

El formato de lanzamiento se refiere a LAU1 y LAUn.

Criterio de entrada

1. Los estudiantes han completado el curso PSP
2. Los estudiantes han leído el libro de texto capítulos 1, 2,3 y Apéndice A

En la revisión general del curso, el instructor revisa los objetivos TSPi del curso y discute lo que se espera lograr. Este es el momento de preguntar acerca de cómo se evaluará el trabajo y discutir las calificaciones individuales y en equipo. El objetivo general del curso es que los miembros del equipo trabajen cooperativamente para producir un efectivo resultado general.

Para ser más efectivo, se necesitan combinar los talentos y habilidades de los integrantes. Para proveer la información necesaria, se debe completar el formato INFO. Esta información se utiliza para asignar el rol y las tareas.

En el siguiente paso del proceso, el instructor describe el producto a construir y contesta las preguntas relacionadas al curso.

Durante la siguiente reunión, el instructor lleva a cabo las asignaciones de equipo y rol. Cuando es posible, el instructor mantiene al mismo equipo durante todo el curso.

Posteriormente, el instructor revisa el proceso de fijación de metas y las metas de TSPi.

Después de que los equipos han establecido metas y roles, es tiempo de llevar a cabo la primera reunión de equipo. Le provee una oportunidad al equipo para discutir y ponerse de acuerdo en los objetivos para el ciclo de desarrollo. Para llevar a cabo la junta, debe seguirse el formato WEEK. Lo más importante de la reunión semanal es reunir y analizar la información del equipo de la semana anterior y para el ciclo de desarrollo hasta la fecha. Todos los integrantes entregan su formato WEEK al gerente de planeación y éste elabora el WEEK del equipo.

Una parte importante de la reunión semanal es que los miembros se pongan de acuerdo acerca de la información que se le entregará al gerente de planeación y cuándo se le entregará.

En este punto, se ha completado el proceso de lanzamiento y el equipo está listo para comenzar el proyecto, siguiendo el formato STRAT.

3.6 Resumen

En este capítulo se describen los pasos en el proceso de lanzamiento TSPi. Durante el lanzamiento, el instructor lleva a los estudiantes a través del resumen del curso, la recopilación de información de los estudiantes, los objetivos del producto, trabajos en equipo, fijación de objetivos de equipo y la reunión de equipo. Luego los equipos mantienen su primera reunión semanal y están de acuerdo en cómo y cuándo van a proporcionar los datos semanales con el gerente de planificación. Tras el lanzamiento del equipo, los equipos siguen el proceso TSPi para iniciar sus proyectos.

El proceso de lanzamiento se utiliza para formar y construir equipos, determinar los roles de los miembros del equipo, establecer los objetivos del equipo, y ayudar a los equipos a que decidan sus prácticas de trabajo. Definir y acordar de qué se encargará cada función es un primer paso esencial en la formación de equipos. El instructor hace que las asignaciones iniciales de rol, que en posteriores ciclos se puede asignar de acuerdo a las preferencias de los estudiantes.

El establecimiento de objetivos se realiza al inicio de cada ciclo del proyecto TSPi. Los objetivos del equipo establecen el marco para la estrategia y el plan. Estos, a su vez, proporcionan la base para el equipo posteriormente. Los equipos deben saber cómo medir los objetivos, establecer las metas no agresivas y proveer los esfuerzos para lograr dichos objetivos. La única manera de aprender a establecer objetivos útiles, es trabajar en ellos para satisfacerlos.

Para TSPi, los tres objetivos básicos son: producir un producto de calidad, ejecutar un proyecto productivo y bien administrado, y acabar en el tiempo estimado. Las medidas de éxito son que cada equipo produzca un producto de calidad, con funciones previstas de acuerdo al calendario previsto.

4

Desarrollo de estrategia

4.1 Primera planeación

4.2 ¿Qué es una estrategia?

4.3 Diseño conceptual

4.4 Gestión de riesgos

4.5 Estrategia de reutilización

4.6 Estrategia de scripts

Criterio de Entrada

1. Los estudiantes han leído el Capítulo 4
2. El instructor ha revisado y discutido el proceso TSPi
3. El instructor ha descrito los objetivos generales del producto
4. Los equipos de desarrollo se han formado y se han asignado los roles
5. Los equipos se han puesto de acuerdo acerca de los objetivos de su trabajo

Establecer Criterios de Estrategia

Aunque pueden existir muchas estrategias viables para incluso los productos más simples, es importante primero establecer los criterios para evaluar las estrategias y luego utilizar estos criterios para revisar las propuestas. El principal objetivo de la estrategia es minimizar el riesgo de que se exceda el calendario de desarrollo disponible. Los criterios sugeridos para lograr este objetivo son los siguientes:

1. El producto del ciclo 1 provee un subconjunto de trabajo de función mínima del producto final.
2. El producto del ciclo 1 provee una base que puede mejorarse fácilmente.
3. Los productos del ciclo son de alta calidad y pueden probarse fácilmente.
4. El diseño del producto tiene estructura modular que permite a los miembros del equipo trabajar independientemente.

Producir el diseño conceptual

El gerente de desarrollo lidera al equipo en la producción del diseño conceptual. En el primer ciclo de desarrollo, este diseño debe incluir todas las funciones que se pretendan desarrollar en los tres ciclos de desarrollo. Como parte del proceso de estrategia, se va a decidir qué funciones a desarrollar en cada ciclo.

Seleccionar la Estrategia de Desarrollo

Con el criterio establecido, el siguiente paso es producir una estrategia de desarrollo propuesta. Una forma de hacer esto es examinar las funciones deseadas para el producto total y luego seleccionar un subconjunto mínimo de trabajo. El objetivo general es tener diversos elementos funcionales en desarrollo en cada ciclo.

Al desarrollar una estrategia, se debe reconocer que algunos elementos de producto muy probablemente serán más grandes que lo planeado. Entonces, si se pretende incluir mucha función en el primer ciclo, el trabajo puede tomar más de lo planeado y puede no tenerse el tiempo de completar los ciclos de desarrollo subsecuentes. La llave es producir un producto trabajable mínimo que se pueda probar y demostrar.

Aunque el equipo puede seleccionar cualquier método para desarrollar la estrategia, usualmente es una buena idea que el gerente de desarrollo proponga una estrategia inicial y el diseño conceptual. Luego debe revisarla el equipo. Si otros integrantes también desean hacer propuestas, deberían hacerlo. Luego hay que evaluar las propuestas realizadas y ya sea seleccionar una o combinarla y ajustarla para producir una que sea aceptable para el equipo.

Producir el Estimado Preliminar de Tamaño

Al producir la estrategia, se debe estimar cuántos LOCs va a requerir de manera probable cada función del diseño conceptual. Estos estimados son necesariamente hipotéticos pues aún no se ha decidido cómo construir el producto.

Después de postular las funciones y componentes probables, hay que considerar cada elemento del programa y juzgar cuántos LOCs nuevos y cambiados va a necesitar. Luego se deben registrar los estimados en el formato STRAT. Con cada ciclo subsecuente de desarrollo, hay que reevaluar esta estrategia y los estimados de tamaño y tiempo.

Producir el Estimado Preliminar de Tiempo

Usando el tamaño estimado del programa y la tasa de LOC por hora, se debe estimar cuánto tiempo tomará desarrollar cada función de manera probable.

Evaluar riesgos

Se deben evaluar los riesgos de acuerdo a su probabilidad de ocurrencia e impacto. Las clasificaciones son H (alto), M (medio) y L (bajo).

Documentar la estrategia

Al final del paso del desarrollo de la estrategia, hay que utilizar STRAT para documentarla.

Actualizar la Estrategia de Desarrollo

Hay que ingresar los nombres y tamaños de los componentes ya desarrollados y de las funciones que cubren. Luego hay que definir las funciones a añadir para el siguiente ciclo de desarrollo.

El proceso de manejo de configuración tiene diversas funciones clave. Debe registrar lo siguiente:

1. Copias de cada versión de cada elemento del producto
2. Un registro de todos los cambios hechos a cada línea de base
3. Quién hizo el cambio
4. Cuándo se hizo el cambio
5. Qué fue el cambio
6. Por qué se hizo el cambio

El gerente de soporte produce el plan de manejo de configuración y lo revisa con el equipo.

4.7 Resumen

5

Plan de desarrollo

5.1 La necesidad de planificación

5.2 Proceso de planeación TSPi

5.3 Herramienta de soporte TSPi

5.4 Desarrollo de scripts de plan

Criterio de Entrada

Hay que asegurarse de haber completado la estrategia de desarrollo y el diseño conceptual.

Planeación de Proyecto Paso 2.1

Enlistar los productos a producir en este ciclo de desarrollo y estimar sus tamaños. Aquí se nombran las partes del sistema y se estima cuán grandes serán. También se identifican otros productos y se estiman sus tamaños.

Planeación de Proyecto Paso 2.2

Registrar la información de producto y tamaño para este ciclo de desarrollo en el formato SUMS.

Planeación de Proyecto Paso 3.1

Producir una lista de tareas. Esta es la lista de los pasos principales en todos los scripts de desarrollo de TSPi.

Planeación de Proyecto Paso 3.2

Producir los estimados de tarea para el equipo y de manera individual. Estimar cuánto tiempo tomará cada tarea de manera probable. El equipo completo debe hacer estos estimados.

1. Para cada tarea, ingresar en “Fase” la fase del proceso en la que se llevará a cabo.

2. En “Parte” ingresar el nombre y/o número de parte producida por la tarea (esto utilizando como base la información contenida en el SRS).
3. Ingresar el número de ingenieros que estarán involucrados en la tarea en el espacio marcado como “#Ingenieros” en el formato TASK
4. Ingresar las horas estimadas para cada ingeniero en la columna para ese rol determinado.
5. Si los estimados de tiempo para algunas de estas tareas son mayores a 10 horas, romper las tareas en subtareas más pequeñas y estimar el tiempo requerido para cada una.
6. Ingresar el tiempo total en horas para cada tarea.

Planeación de Proyecto Paso 4.1

Ingresar las horas semanales en el formato SCHEDULE. Estimar cuántas horas les tomará a ti y a tus compañeros dedicarse al proyecto cada semana. Mientras se planea el ciclo 1, incluir horas para ocho o más semanas, al menos hasta el primer intento del plan.

Planeación de Proyecto Paso 4.

Producir los formatos de equipo TASK y SCHEDULE. Después de ingresar las horas semanales en el formato SCHECDULE para el equipo, se usará la herramienta TSPi para calcular el valor planeado y la fecha de compleción anticipada para cada tarea. Luego generar los formatos TASK y SCHEDULE completos.

Como paso final en la compleción del plan de equipo, hay que revisar que el plan incluya suficientes horas para producir el producto en el tiempo disponible. Seguir los siguientes pasos:

1. Totalizar las horas que los miembros planean invertir en el proyecto durante este ciclo.
2. Comparar el tiempo total desde el paso 1 con el tiempo total requerido para hacer todas las tareas del ciclo.
3. Si los tiempos de tarea exceden el tiempo de trabajo total planeado, no tiene punto realizar el siguiente paso.
4. Si el trabajo es muy grande, las alternativas son trabajar más o reducir el trabajo planeado. Si se deciden por más horas, ingresar las nuevas horas en el borrador de horario y ver si se ajusta al tiempo disponible.
5. Si se decide reducir la cantidad de trabajo, regresar al paso de estrategia y revisar las funciones a implementar durante este ciclo. Rehacer el plan.

No se deben reducir los estimados.

Después de haber completado los formatos TASK y SCHEDULE para el equipo, hay que completar las porciones de tamaño y tiempo en el formato SUMP.

Planeación de Proyecto Paso 5

Producir el plan de calidad. Para hacer el plan de calidad, se debe comenzar estimando los defectos a inyectar en cada fase. Se debe hacer esto para el producto completo. Luego

ingresar la meta de rendimiento para cada fase de remoción de defectos. Finalmente, se genera un plan de calidad de prueba para saber si cumple con estos criterios. Si no, hacer ajustes y corree otro plan de prueba. Seguir estos pasos:

1. Estimar cuántos defectos el equipo inyectará en cada fase. Basar este estimado en el tiempo invertido por fase multiplicado por la tasa de inyección de defectos.
2. Estimar el rendimiento que el equipo alcanzará para cada fase de remoción de defectos. Basar este número en los criterios de rendimiento en la tabla 5.8. Cuando se ingresa el rendimiento, la herramienta calcula los defectos en el producto en esa fase y deduce el porcentaje de rendimiento.
3. Al estimar el rendimiento para las fases de prueba, nota que el rendimiento de prueba depende de la densidad de defectos ingresada en las fases de prueba. Comenzar con las metas de rendimiento para estas fases de prueba y ver si los valores de defectos/KLOC vienen en rangos específicos. Si se encuentran muchos defectos, reducir los rendimientos de fase de prueba hasta que los rendimientos y defectos/KLOC comparan con los criterios.
4. Utilizar la herramienta TSPi para generar el plan de calidad de prueba.
5. Comparar los valores en este plan de calidad de prueba con la tabla 5.8. Si el total de defectos/KLOC no están dentro del rango sugerido para los ingenieros entrenados en PSP, hay que incrementar o disminuir los tiempos de desarrollo para el diseño y fases de código o ajustar los defectos inyectados en una o más fases. Continuar ajustando los tiempos de fase o tasas de inyección de defectos hasta que el total de defecto/KLOC cumpla con los criterios.
6. Si las tasas de remoción de defectos y tasas de revisión e inspección no están razonablemente cerca de los criterios, ajustar el tiempo invertido en las fases de remoción de defectos o cambiar los rendimientos de remoción de defectos hasta que estos valores estén más cerca de los criterios. Se ajusta el formato TASK.
7. Si las proporciones de defecto son bajas, reexaminar la información para asegurarse de que se cumplen los demás criterios.
8. Si los números de defectos/KLOC son muy altos en la compilación o prueba, revisar que nuevamente el valor total de defectos/KLOC esté dentro del rango.
9. Repetir pasos 4 a 8 hasta que el plan de calidad cumpla con los objetivos.
10. Utilizar herramienta TSPi para generar el formato SUMQ final.

Planeación de Proyecto Paso 6

Hacer planes individuales. Deben llenar los formatos TASK y SCHEDULE de manera personal.

Planeación de Proyecto Paso 7

Balancear la carga de trabajo. Esto con base en los formatos TASK y SCHEDULE. Si se cambia algo en estos formatos, también cambian los formatos SUMP y SUMQ.

Paso Final de Planeación

Producir y distribuir los planes. Cuando los planes individuales estén balanceados, el paso final es producir el plan de equipo consolidado: SUMP, SUMQ, TASK y SCHEDULE.

Criterio de Salida

Se requieren los formatos TASK y SCHEDULE terminados para el equipo y de manera individual. También se requieren los formatos SUMS, SUMP y SUMQ.

5.5 Rastreo de trabajo

5.6 Plan de calidad

5.7 Resumen

6

Definición de requerimientos

6.1 ¿Qué son los requerimientos?

Se utiliza el documento SRS (que no es fundamental, pero es muy útil) para hacer una descripción clara y precisa del producto a crear. Los requerimientos proveen retroalimentación al cliente para garantizar el cumplimiento de las necesidades; el instructor puede actuar como cliente para asegurar que se esté logrando lo esperado.

En el proceso de requerimientos se deben plantear preguntas de parte de todos los integrantes, y que, para todas, debe existir alguna respuesta. Las adivinanzas también se recomiendan para garantizar que se entiende el problema del cliente, es decir, se valida si realmente se sabe lo que se quiere hacer.

Los requerimientos responden a dos criterios: declaración inicial de las necesidades del cliente y, explicar para qué se quiere hacer el proyecto. El instructor es la última palabra en la construcción de los requerimientos y así, del cumplimiento de estos criterios; tener acceso al usuario final otorga más seguridad.

En el ciclo 1, esta fase tarda una semana, en el ciclo 2 y 3, esta fase tarda media semana.

6.2 Por qué necesitamos los requerimientos

Los requerimientos se necesitan para identificar claramente las necesidades de los clientes. Cada uno revisa las necesidades del cliente. Para lograr entender estas necesidades se deben hacer discusiones en torno a preguntas planteadas, a fin de identificar cuáles de ellas se entienden, y con ello añadir información fundamental al proyecto.

Alcanzar el entendimiento común de lo que se va a construir; es esencial la participación de todos.

6.3 Cambios de los requerimientos

Como, hasta que el usuario prueba el sistema, es que sabe cómo va a cambiar su ambiente de acuerdo al producto que le proporcione el equipo de desarrollo, se recomienda un alto contacto con el cliente, a fin de aprender a detalle y con claridad lo que necesita.

Con el documento SRS se puede hacer más eficiente el ponerse de acuerdo con los usuarios, haciendo que los ingenieros expongan las funcionalidades ofrecidas por el producto, a los usuarios. Aun así, todos los cambios cuestan dinero y tiempo.

Pasos principales pasos para la obtención de requerimientos:

1. Evaluar la factibilidad del sistema.
2. Entender los problemas de organización.
3. Identificar los accionistas (stakeholdes) del sistema.
4. Registrar las fuentes de requerimientos.
5. Definir el ambiente operativo de sistema.
6. Evaluar los problemas de negocio.
7. Definir las restricciones de dominio.
8. Registrar la justificación de requerimientos.
9. Prototipar los requerimientos pobremente obtenidos.
10. Definir los escenarios de uso.
11. Definición del proceso operacional.

6.4 Especificación de requerimientos de software

Consiste en describir con propias palabras lo que se va a desarrollar y cómo se va a desenvolver el producto. Producir los requerimientos para permanentemente hacer el diseño. Se debe evitar el cómo y buscar el qué se va a hacer.

Temas principales a tratar:

- Requerimientos funcionales: entradas, salidas, cálculos y casos de uso.

-Interfaz externa de requerimientos: usuario, hardware, software, comunicación.

- Restricciones de diseño: formatos de archivo, lenguajes, estándares, compatibilidad.

- Atributos: disponibilidad, seguridad, mantenibilidad, conversión.

- Otros requerimientos: base de datos, instalación

Formato SRS (no debe por qué ser largo, debe ser específico)

1. Tabla de contenido
2. Introducción.
   2.1 Objetivo de SRS.
   2.2 Declaración del problema.
   2.3 Información del proyecto de equipo.
3. Declaración de requerimientos funcionales.
   3.1 Descripción de los requerimientos de la función del sistema.
   3.2 Requerimientos ciclo 1.
   3.3 Requerimientos ciclo 2.
   3.4 Estrategia de arriba-abajo.
4. Definición de reglas utilizadas en los requerimientos.
5. Requerimientos de interfaz externa.
6. Restricciones de diseño/implementación
7. Requerimientos especiales de sistema.
8. Referencias y fuentes de información.

Para hacer la rastreabilidad de requerimientos se puede utilizar una numeración por párrafos o líneas sobre el documento SRS (ver apéndice A).

6.5 Scripts de requerimientos TSPi

Criterio de Entrada

- Se tiene una estrategia de desarrollo y un plan
- Se tiene el diseño conceptual.
- Todos los miembros del equipo han leído este capítulo y producido la declaración de necesidad del producto.

Revisión de declaración de necesidad

Se definen los requerimientos del producto. El gerente de desarrollo guía al equipo a lo largo de la revisión de la declaración de necesidad del producto y se asegura de que todas las preguntas e incertidumbres sean resueltas. En esta revisión, hay que concentrarse en las funciones se espera incluir durante el ciclo 1.

Aclaración de la Declaración de Necesidad

El gerente de desarrollo revisa la lista de preguntas del equipo para aclarar con el instructor.

Asignación de Tareas de Requerimientos

El gerente de desarrollo guía al equipo en la estructuración del SRS y los ayuda a identificar el trabajo requerido para producir el SRS. El equipo se compromete con fechas para completar el documento.

Documentación de Requerimientos

Al documentar el SRS, se hace una breve y clara declaración de lo que se pretende construir. Todos tienen que llegar a un acuerdo. Hay que asegurarse de registrar y rastrear la fuente de todas las declaraciones de requerimientos.

Plan de Prueba del Sistema

Se deben hacer pruebas bajo varias condiciones de error así como considerar usabilidad y problemas de recuperación.

Requerimientos e Inspección de Planeación de Prueba de Sistema

Inspección del SRS completo junto con el plan de prueba del sistema. En la inspección del SRS hay que identificar preguntas o desacuerdos y luego reunirse para la inspección, que debe ser liderada por el gerente de Calidad/Proceso que sigue el formato INS. El objetivo de la inspección es encontrar cualquier problema en el SRS así como inconsistencias antes de comenzar el trabajo de diseño.

Actualización de Requerimientos

Actualizar el SRS y el plan de prueba de sistema para corregir los problemas encontrados durante la inspección.

Revisión de usuario del SRS

El usuario final debe leer el SRS y estar de acuerdo en lo que describe.

Línea base de Requerimientos

El gerente de desarrollo hace un SRS oficial que ahora únicamente puede cambiarse utilizando el procedimiento de control de cambios (CCR).

Criterio de Salida

- SRS completo, actualizado e inspeccionado, plan de prueba del sistema bajo control de configuración.

- Toda la información de tiempo personal, defecto y de tamaño ingresada en formatos TSPi y sistema de soporte.

- INS completo a partir de la inspección de requerimientos.

- Copias de todos los formatos, SRS y plan de prueba del sistema en el cuaderno del proyecto.

6.6 Resumen

En la fase de requerimientos, se produce la especificación de requisitos de software. El propósito es describir en propias palabras, aquellas funciones del producto a desarrollar. El SRS también debe proporcionar criterios claros e inequívocos para la evaluación del producto terminado, así como la orientación sobre la verificación de lo que se supone que debe hacer. Aunque el documento SRS es útil, no es lo más importante. Lo que realmente se quiere, es un acuerdo común de equipo de qué construir; es por esto que los requisitos de un proceso bien estructurado son importantes.

A medida que los usuarios aprenden más sobre el producto, ellos van a querer cambiar los requerimientos. Estos cambios a menudo son problemas serios, ya que cuestan tiempo y dinero.

Garantizar la trazabilidad funcional, el número de párrafos y las secciones del SRS, pueden ayudar a identificar el origen de cada declaración. Es necesario trazar el código desde el diseño y desde el diseño hasta el SRS y el estado de necesidad. Se debe etiquetar todos los requisitos de declaración de SRS como se explica en el apéndice A. Cuanto más sea posible, mantener todos los estados funcionales en oraciones separadas, breves y precisas.

7

Diseño con equipos

7.1 Principios de diseño

Se describe el proceso de diseño detallado en los formatos IMP1 e IMPn. DES1 y DESn menciona el proceso de diseño detallado.

7.2 Diseño en equipos

Hay la necesidad de definir la estructura total del sistema, y la forma de lograrlo, es trabajando todos juntos. Un diseño de alto nivel contiene interfaces, estructura lógica, funciones unitarias y generalidades del programa.

Un buen estudio de diseño puede proporcionarse gracias al desarrollo de prototipos. Si algunos integrantes del equipo son capaces de llevar a cabo el diseño, puede asignárseles la tarea, y uno o dos integrantes, encargarse del desarrollo del prototipo para posteriormente probarlo con los usuarios típicos del sistema. Este prototipo se asocia principalmente a la interfaz.

7.3 Estándares de diseño

Convenciones. Se debe definir el glosario del sistema, módulos, componentes, variables, archivos, cambios de nombre.

Formatos de interfaz. Incluye parámetros de variables, códigos de error, u otras condiciones.

Mensajes de sistema y error. Establece un estándar de formatos y mensajes entendibles. Ahorra el tiempo de desarrollo.

Estándar de defectos de TSP.

[TABLA]

Hay que tener diseños precisos, y seguramente contar con los casos de uso nativos de PSP, buscando cómo es que el programa se va a utilizar, de modo que se identifique problemas de diseño y la usabilidad.

7.4 Diseño para la reutilización

Hacer un programa de reúso desde el principio del proyecto fomenta la producción de productos de alta calidad. Hay que definir funciones reutilizables autocontenidas, para así, contar con formas de comparar diseños con bajo acoplamiento.

Se debe garantizar el desarrollo de una buena documentación para evitar tener que mirar el código fuente. De igual manera, pruebas unitarias sobre código reusable permite visualizar la calidad de éstas.

Responder a las preguntas:

1. ¿El programa tiene las funciones correctas?
2. ¿La interfaz es adecuada?
3. ¿El rendimiento es el adecuado?
4. ¿El programa cumple con los estándares de calidad?

Si al responder a estas preguntas la mayoría de respuestas es "No", entonces el reuso llevará más trabajo.

[SOPORTE DE APLICACIÓN]

7.5 Diseño para la usabilidad

Consiste en la construcción de prototipos que son mostrados constantemente al cliente para identificar las interfaces.

7.6 Diseño para pruebas

Desarrollar código de pruebas permite probar componentes específicos del programa. Se realizan pruebas de caja negra y blanca. Las pruebas de caja blanca muestran los caminos que recorre la ejecución del programa, y las pruebas de caja negra, representa los caminos externos.

7.7 Revisión de diseño e inspecciones

[AYUDA]

7.8 Scripts de diseño TSPi

Criterio de Entrada

Haber completado la estrategia de desarrollo y el plan, haber documentado los estándares de diseño, y un SRS completo e inspeccionado.

Diseño de alto nivel

Se produce un diseño de alto nivel con los siguientes pasos: decidir la estructura general del producto, nombrar todos los componentes del producto, asignar funciones del producto a los componentes, producir las especificaciones externas del componente, asignar funciones de

casos de uso a estos componentes, e identificar las tareas de diseño a completar. El gerente de desarrollo es quien guía al equipo en estos pasos.

Estándares de Diseño

Se producen los estándares de diseño y el glosario de nombres con ayuda del gerente de calidad/proceso.

Estructura general del producto

Primero se produce un diseño conceptual de alto nivel. Esto identifica las partes del componente, nombra sus funciones generales y decide cómo relacionarla. Existen muchas formas de hacer el diseño: diagramas de flujo, casos de uso u otras metodologías.

Para asignar las funciones del producto a los diferentes componentes, utilizar una tabla de rastreabilidad. Utilizar el formato STRAT para enlistar cada componente con sus funciones. Agrupar las funciones por componente, junto con las referencias a los párrafos de SRS para cada función. Luego etiquetar los componentes. La ventaja de esta tabla es que provee una forma rápida de ver lo que hace cada componente e identificar funciones omitidas o redundantes.

Asignación de Tareas de Diseño

El gerente de desarrollo guía al equipo en la creación del documento SDS. El equipo asigna estas secciones a los diferentes integrantes y se comprometen con fechas de entrega.

Especificación de Diseño

Mientras se produce el SDS, hay que definir las interfaces externas y especificaciones funcionales para cada componente. Como parte del SDS, hay que generar un conjunto completo de escenarios que cubran las funciones externas de todos los componentes.

El paso final en el diseño de alto nivel es producir varios documentos de diseño que especifiquen la lógica y estructura del programa principal del sistema. Después de producir el SDS, hay que revisar el trabajo y arreglar los problemas que se encuentren.

Cuando se termine de revisar el SDS personal, se le da al gerente de desarrollo y lo incorpora en el SDS general final.

Plan de Prueba de Integración

Es importante producirlo mientras se hacen las especificaciones de diseño. La razón es que las pruebas de integración se supone deben revisar y verificar todas las interfaces a lo largo de los componentes del sistema.

Inspección de diseño

Todo el equipo debe participar en la inspección, encontrando el número máximo de defectos de diseño y asegurándose de que todos los integrantes entienden el diseño.

Actualización del diseño

Después de la inspección de diseño, se deben arreglar los errores identificados en tu parte del SDS y/o plan de prueba de integración y hacer que uno o dos ingenieros revisen las correcciones si se necesita. Luego se le da el SDS corregido y revisado al gerente de desarrollo, que produce y distribuye los documentos completos.

Línea de base del Diseño

El gerente de soporte hace el SDS oficial y le da una copia a cada integrante del grupo.

Criterio de Salida

1. SDS completo, inspeccionado y corregido.
2. Plan de prueba de integración completo, inspeccionado y corregido.
3. Estándares de diseño y glosario de nombre.
4. INS completo.
5. Formatos SUMP y SUMQ actualizados.
6. Información de tiempo, tamaño y defectos de los ingenieros ingresada a la herramienta.
7. Copias de los materiales de diseño en el cuaderno de trabajo.

7.9 Resumen

El objetivo principal del proceso de diseño es asegurar que los ingenieros producen minuciosos diseños, de alta calidad. En el diseño de los equipos, primero producir una estructura global de diseño y luego dividir el producto en sus componentes principales. Los miembros del equipo, posteriormente diseñan por separado y ofrecen sus diseños para el gerente de desarrollo, que las combina en una única especificación de diseño de sistema. También es importante para producir e inspeccionar el plan de pruebas de integración, al mismo tiempo que se produce e inspecciona el SDS.

El diseño es el proceso creativo de decidir cómo se va a construir un producto. Su objetivo es proporcionar una base precisa, completa y de alta calidad para su implementación. Un diseño completo también debe definir cómo las partes del producto interactúan, cómo van a ser ensambladas, para producir el sistema acabado y terminado. A continuación, el diseño de alto nivel se hace una tarea completa y precisa, los ingenieros deben implementar componentes sin más orientación que el diseño. Esto requiere que los diseñadores utilizan un diseño preciso y con notaciones no ambiguas.

El proceso de TPSi para producir el diseño de alto nivel, consta de cuatro pasos: decidir sobre la estructura general del producto, las funciones asignadas a los componentes del producto, la producción de las especificaciones externas de los componentes, y decidir cuáles componentes y funciones desarrollar en cada ciclo.

Uno de los problemas en el diseño de grandes productos de software, es que la estructura básica debe definirse antes de cualquier otra cosa especificar. Hasta que esta estructura se

resuelve, se vuelve difícil dividir el trabajo entre los miembros del equipo. Esto hace que sea difícil de utilizar todas las energías y capacidades plenas de los miembros del equipo. Esto proporciona una excelente oportunidad para llevar a cabo los estudios de diseño, producción de interfaces de prototipo, desarrollo de normas de equipo, o desarrollar planes y especificaciones de reutilización.

Se deben establecer normas que tratan de convenciones de nomenclatura, formatos de interfaz, mensajes de sistema y de error, y las normas de tamaño. Tal vez la norma de diseño más importante se refiere a la representación de diseño. La reutilización es una forma potencialmente poderosa para mejorar la productividad de equipo. Las cuestiones clave son la reutilización de interfaces estándar, convenciones de llamada, estándares de documentación, calidad de productos y soporte de aplicaciones.

8

Implementación de producto

8.1 Diseño de criterios de complexión

Repetir los scripts DES1 y DES desde que no se alcance la especificación de alto nivel. Se pueden implementar algunos elementos del producto, mientras se diseñan otros; aunque es un proceso es irregular.

Todas las especificaciones de alto nivel deben estar completas e inspeccionadas para proceder con la implementación, sin embargo, el dividir el proyecto en varios componentes permite trabajarlos en forma independiente de tal manera que, se puede implementar parte del producto total.

8.2 Estándares de implementación

No debe tomar gran cantidad de tiempo su definición. Primero se debe estar de acuerdo de los estándares a construir, asignando a uno o dos miembros la construcción de borradores, para que sean discutidos por todos los miembros del equipo. El gerente de calidad/proceso es quién dirige esto.

Revisión de estándares, nombres de interfaz, estándares de tamaño, estándares de codificación, estándares de mensaje, estándares de defectos y estándares de prevención de defectos. Si un estándar propuesto se visualiza útil, se recomienda trabajarlo, sin que esto implique que el estándar sea perfecto; trabajar con un estándar que se ajuste será más sencillo, y cuyas mejoras será mejor registrarlas para futuros proyectos.

Revisar el nombre, interfaz y estándares desarrollados durante la fase de diseño para asegurar que son apropiados. Revisar que la lista de rutinas reutilizables esté completa y que todos los miembros del equipo la estén utilizando. Hay que revisar el glosario de nombres para que todos los estén utilizando. Revisar que las variables compartidas, nombres de archivos y parámetros son consistentes.

Las inspecciones de código serán más fáciles y efectivas al usar todos, las mismas notaciones. Hacer también buenas prácticas en comentarios. Si se usan las mismas convenciones, seguramente serán elementos a reusar a futuro.

Cada equipo tiene un estándar de conteo (que se define en la fase de diseño), como los LOC, documentos SRS y SDS. Se recomienda el conteo de páginas para definir el tamaño de estos documentos. Documentos: Requerimientos SRS, Diseño de alto nive SDS, diseño detallado, pantallas y reportes, bases de datos, scripts de prueba, mensajes y materiales de prueba. Para requerimientos se puede usar el conteo de páginas de texto o párrafos. Para el diseño detallado se suele hacer conteo del número de líneas de texto, párrafos o número de casos de uso.

Estimar y rastrear el trabajo es lo que se está buscando. Para trabajos pequeños se recomienda sólo hacer un conteo de tiempo, no de tamaño. Asignar categorías de tamaño puede ayudar a determinar si un módulo conlleva más o menos tiempo.

Categorizar los defectos es una tarea difícil y requiere ser puntual para especificar claramente sin con ello tener una lista enorme de posibles defectos. Saber diferenciar entre tipos de defecto y causas de defecto (desconocimiento del lenguaje es una causa).

Categorías recomendadas para prevención de defectos:

1. Educación: conocimiento del lenguaje, ambiente o aplicación.
2. Comunicación: Arreglar el proceso.
3. Transcripción: Usar mejores herramientas.
4. Vista general: Refinar el proceso y utilizar mejores métodos.

Enfoque de prevención de defectos:

1. Elige los tipos de defectos que parezcan que causan el mayor problema.

2. Examina el número de defectos para identificar causas particulares de defectos y decidir a cuáles atender.
3. Cuando veas un defecto y veas que puedes prevenirlo, haz un cambio en el proceso para prevenirlo.
4. Comienza a ver por la siguiente categoría de defecto para manejarlo de la misma manera.

Buscar formas de cambiar el proceso para prevenir el defecto es la clave. Rastrear el rendimiento para ver si ha dado frutos, y notar si ha mejorado, sino, tratar de descubrir qué fue y registrarlo.

8.3 Estrategia de implementación

Debe estar conforme a la estrategia de diseño. En el diseño se trabaja de lo grande a lo pequeño (o específico), en la implementación es al revés. Se crean las funciones para asegurar que se hace lo que el programa espera. Se hacen los objetos atómicos para ir a un siguiente nivel, estando seguro de que cumplen con las especificaciones.

Utilizar los encabezados de comentarios para cada programa fuente, para que usuarios potenciales puedan observar rápidamente lo que hace el programa. Adicionalmente se recomienda colocar advertencias, con variables en mayúscula.

Cuando se hagan reuniones, o a diario informar, se debe preguntar qué y quién tiene elementos reutilizables, de modo que el líder de soporte los añada a la lista de elementos de reúso.

Antes de probar hay que revisar que el código cumple con las especificaciones.

8.4 Revisiones e inspecciones

Muchos defectos son aleatorios especialmente asociados al problema de tecleo. Una o dos semanas se utilizan para la detección de defectos. Es útil tener una lista de inspecciones para verificar y actualizar la lista con base en lo obtenido. El apéndice C trata éste tema.

8.5 Scripts de implementación

Se utiliza IMP1 e IMPn.

Criterio de entrada

1. Una estrategia y plan de desarrollo completos.
2. Especificaciones de SRS y SDS completas, revisadas y actualizadas.
3. Un estándar de codificación bien definido y documentado.
4. Copias disponibles de la lista de especificaciones de las rutinas funcionales, glosarios y todos los estándares adoptados por el equipo.

Implementación de la planeación

Como hay ingenieros mejores que otros, se reitera el enfoque de: algunos orientados al diseño y otros orientados a la implementación. El primer paso es revisar los trabajos a realizar y que dichas tareas estén distribuidas en varios miembros del equipo. Un enfoque es tener algunos ingenieros especializados en diseño y otros en implementación.

Después de asignar las tareas, cada ingeniero planea su trabajo de implementación; para estas tareas de pocas horas, un tiempo estimado es adecuado. Deben hacer un plan con los formatos SUMP y SUMQ, y para tareas no sustanciales utilizar el formato SUMTASK. Asumiendo que se tenga un estimado de LOG para cada módulo y has obtenido información de productividad personal, el trabajo de planeación no debería tomar mucho tiempo.

El siguiente paso es conducir una revisión personal de diseño que consta de revisar los Loops con tablas de rastreabilidad o máquinas de estado.

Después de fijar los problemas encontrados en la revisión hay que desarrollar cualquier código especial de prueba de unidad. Como el mayor error se encuentra en problemas de diseño, es necesario hacer el desarrollo de pruebas antes de la inspección de diseño detallado. Los planes de prueba revisan los caminos lógicos, Loops, variables, parámetros, límites y aseguramiento de la buena ejecución del programa.

Después del desarrollo de prueba, se lideran a uno o dos integrantes para el diseño detallado. Se ingresa esta información de inspección en el formato INS y los defectos más importantes en el LOGD. Hay que dar copia del INS al líder de calidad/proceso y al líder de equipo.

Después de la inspección de diseño se debe programar el módulo. Antes de que personalmente revises, mirar la historia de defectos y deducir cuántos defectos se puedan inyectar durante la programación. Con las propias tasas de defectos inyectados (mínimos y máximos) se puede hacer un estimado bueno. Es buena idea fijar un objetivo de tiempo; 50% del 78% que se gasta. Una simple lectura del programa fuente puede tomar más del tiempo de crear el código. Hay que revisar la consistencia, igualdades, punteros, secuencias de llamado, etc; como hay mucho que ver, hay que garantizar el checklist aun cuando se tarde más del tiempo estimado. Otra buena técnica es que otro ingeniero revise tu programa (en caso de que uno no pueda encontrar más defectos) para que así pueda compilar el programa.

Después de la compilación compara el diseño, revisión de diseño, código y tiempos de revisión de código con el plan de calidad del equipo. Hay que revisar los niveles de defecto y tasas de defecto, si la información indica que hay problemas hay que revisar los resultados con el líder de calidad/proceso. Una forma de decir que se está cumpliendo (tabla 5.8), tiempo invertido en el diseño debe ser mayor que el tiempo de codificación; el tiempo invertido en la revisión de diseño debe ser mayor al 50% del tiempo de diseño,… (lista)

Si la calidad del programa es pobre y si el rendimiento ha sido menor al 70%, el líder de calidad proceso que lleve a cabo la inspección de código. Entre más pobre sea la calidad del programa, y menor el rendimiento, mayores revisores se van a necesitar (apéndice C). Hay que utilizar clientes que no conocen el programa sin mencionar los defectos y de acuerdo a lo que ellos identificaron tomar acciones descritas en el apéndice C.

El siguiente paso son las pruebas de unidad. Se utilizan los materiales y el plan de prueba.

Siguiendo la inspección del código y las pruebas de unidad, el líder de calidad/proceso debe revisar la información del componente, para saber si es suficiente para incluirlo en el sistema base. El criterio para ello se usa el plan de calidad (formato SUMQ tabla 8.3 muestra que se cubren niveles de defectos, proporciones de defecto, proporciones de tiempo), el equipo entra en discusión y decidir qué hacer. Las opciones son poner el componente en el sistema base de todas formas para hacer más diseño, revisiones de código e inspecciones, o poner a un ingeniero retrabajando el componente antes de la integración; esto ahorra al equipo muchos días.

Se debe hacer una nueva revisión sobre el programa completo, reescribiendo cualquier código cuestionable o torpe. Hacer análisis de máquina de estado. También se pueden hacer mismos checkeos en forma individual para que cada uno descubra de forma independiente los defectos.

Finalmente hay que darle al líder de soporte la revisión de calidad.

Criterios de salida

1. Componentes completos, revisados e inspeccionados.
2. Los componentes ingresados en el sistema de configuración de manejo.
3. Formatos INS LOG completos para las revisiones de diseño, código e inspecciones.
4. Formatos SUMP, SUMQ y SUMS actualizados para el sistema y todos sus componentes.
5. Planes de prueba de unidad, soporte. Tamaño, tiempo e información de defectos.
6. Cuaderno actualizado.

8.6 Resumen

Los principales pasos en el proceso de implementación es la planeación de la implementación, diseño detallado, especificación detallada, revisión de código, prueba de unidad, revisión de calidad de componentes y liberación de componentes. Los estándares de implementación también se desarrollan para añadir y extender los estándares definidos en la fase de diseño.

La estrategia de implementación debe ser consistente con la estrategia de diseño, al hacer esto hay que considerar las pruebas, reúso e inspecciones. Para hacer las pruebas e inspecciones más eficientes la implementación debe comenzar con los detalles hasta lo global. Primero módulos más bajos y llegar a estructuras de alto nivel.

Tanto en diseño como en implementación, hay que considerar el reúso. En la paso de implementación, el líder del equipo guía al equipo en la asignación de tareas de implementación, luego cada ingeniero producen el diseño detallado para cada módulo y conducen una revisión de diseño personal, también desarrollan cualquier material de prueba requerido. Luego el líder de calidad/proceso lidera a los ingenieros a través de las inspecciones de diseño detallado, en la que cada ingeniero debe por lo menos completar una tabla de rastreo, tabla de ejecución o análisis de máquina de estado de cada loop o estructura

de máquina de estado. Siguiendo la inspección de diseño, los ingenieros codifican los módulos y personalmente los revisan.

Ahora los ingenieros revisan la información de calidad con el líder de calidad/proceso, y utilizan esta información para guiar la inspección de código. Entre más pobre sea la calidad del programa se necesitarán más revisores. Hay que seguir el script INS para la inspección de código. En estas inspecciones varios defectos serán encontrados por uno o varios ingenieros; entre menos ingenieros los encuentren, serán menos los defectos reales que se encuentren en código. Siguiendo INS, el equipo estima el número probable de defectos que quedan y el rendimiento de inspección. Con el rendimiento de inspección uno o más ingenieros deberían hacer la inspección. Finalmente se hace la prueba de unidad, luego el líder de calidad/proceso valida que el componente vaya o no en el sistema base. Entonces los criterios se hacen por el plan de calidad, si el componente tiene problemas de calidad, las alternativas son ponerla en el base o poner otros ingenieros a retrabajarlo antes de la integración.

Después de pasar la revisión de calidad, los componentes completos se dan al líder de soporte para ingresarlo en el sistema base. Al final de esta fase, el equipo debe haber completado, revisado e inspeccionado los componentes que están bajo el manejo de configuración. Deben haber completado los formatos INS y LOGD para las inspecciones de diseño, código y reinspecciones. Formatos SUMP, SUMS Y SUMQ actualizados para el sistema y todos sus componentes. La información de proceso debe estar ingresada en la herramienta y el cuaderno de proyecto actualizado.

9

Integración y prueba de sistema

9.1 Principios para hacer pruebas

Evaluar el producto, no arreglarlo. Generalmente un producto de baja calidad demuestra problemas de calidad en las pruebas.

9.2 La estrategia para hacer pruebas en TSP

Durante las pruebas se verifica que los productos sean de alta calidad.

1. Utilizando partes desarrolladas y probadas por unidad hay que construir el sistema
2. Hacer una prueba de integración de sistema para verificar que todo está presente y que funcionan juntas.

3. Hacer pruebas al sistema de producto para validar que hace lo que los requerimientos requieren. Identificar módulos baja calidad y regresarlos al gerente de calidad/proceso para evaluación y limpieza.
4. Identificar componentes de baja calidad que sean problemáticos.

Construir, hacer prueba de integración y prueba de sistema.

Junta varias partes del sistema para probarlo (construcción de sistema). La prueba de integración identifica si las interfaces son compatibles trabajan juntos. Se verifican las funciones del sistema contra los requerimientos. Se solicita también pruebas de regresión.

9.3 Estrategia de construcción e integración

Asegurarse de que todas las partes están presentes para complementar un sistema de trabajo y proveerlo a la prueba de integración y de sistema. Se define el enfoque de la prueba de integración, que básicamente revisa que todos los componentes estén presentes y todos sus llamados funcionen. No se deben probar las funciones de componentes, eso se hace en la prueba de sistema. La estrategia de integración depende de la estrategia de desarrollo y de todo el trabajo.

Estrategia del Big Bang: poner todas las partes juntas haber si funcionan juntas o no. Requiere menor tiempo de pruebas pero pocas veces es efectiva, aunque sirve para sistemas de baja calidad.

Uno a la vez. Añadir partes a la vez. Te guía a las causas de los problemas, pero posiblemente requiera mayor trabajo de desarrollo de prueba.

Cluster. Añadir partes en cluster. No se puede sólo tomar las partes que sea, únicamente tomar y unir componentes relacionados.

Sistema plano. Integrar todas las partes de más alto nivel, y luego ir ahondando de manera sucesiva en las capas del sistema en paralelo. Se puede añadir un componente a la vez o los que uno quiera. Puede detectar problemas del sistema de manera temprana. Provee la flexibilidad de construir rápidamente un esqueleto total de todo el sistema. Su problema principal, es que requiere grandes números de stock o proveer números falsos.

El mejor enfoque es considerar todas las opciones y aplicar el que mejor se adapte al proyecto en particular.

9.4 Estrategia de prueba de sistema

Prueba de sistema:

¿El sistema desempeña apropiadamente las funciones que debe desempeñar?

¿El sistema cumple sus metas de calidad?

¿El sistema opera apropiadamente bajo condiciones normales?

¿El sistema opera de manera apropiada bajo condiciones adversas?

Hay que revisar el número de áreas clave. Por ejemplo, puede el sistema ser instalado? Inicia de manera correcta? Desempeña todas las funciones establecidas en los requerimientos? Se recuperará de fallas de hardware o software? Su rendimiento es adecuado? Evaluar el tiempo de respuesta sería para este caso.

Identificar los objetivos principales del sistema porque rara vez es suficiente alcanzar a recubrir todas las pruebas; un plan de prueba de sistema. Luego de ello divisar una estrategia para asegurar que todos los objetivos se cumplan.

Estrategias alternativas. Puedes probar los objetivos en serie (con un plan que captura lo más importante del proyecto), probando cada una de las funciones, operaciones bajo condiciones de estrés luego la usabilidad y luego el rendimiento. Este enfoque deja poco o casi nada de tiempo para pruebas reales de sistema.

Pruebas comprensivas. Con un plan de prueba cuidadosamente diseñado, se puede evaluar diversas características de producto.

Otra estrategia es de áreas funcionales. Cubre todos los aspectos de un área antes de pasar a otra. Elimina la duplicación de pruebas. El problema es que no se enfoca al comportamiento general del sistema.

Tercera estrategia: combinación de las anteriores. Funciones de bajo nivel para evaluar el comportamiento normal y anormal, luego se sube un nivel, y se prueban agregados funcionales para asegurar que trabajan juntos, y nuevamente se verifica el comportamiento normal y anormal, se sube de nivel y así sucesivamente. Su desventaja es el tiempo que requiere tomar de forma progresiva las combinaciones funcionales para un sistema grande.

Cuarta estrategia: enfoque de reversa. Se parte con funciones de más alto nivel y se trabaja hacia abajo, haciendo pruebas de estrés y normales. Puedes hacer una prueba extensiva de varias combinaciones funcionales. Cubre problemas de sistema rápido, sólo funciona con productos de calidad (?). Con cada identificación de problemas de calidad, regresar, hablar con el gerente de calidad para que lo reemplace o repare.

9.5 Planeación de pruebas

Se hace en diversos lugares. En TSPi, al hacer la prueba conceptual, se necesitan planes de pruebas para construcción e integración. Plan de prueba que deseas utilizar, te guía mejor.

Se debería tener a este punto (al final) una vista de todos los pasos de prueba a desarrollar, los materiales requeridos para cada prueba, los resultados que las pruebas producen, un estimado para el tiempo de corrida sin defectos, los defectos a encontrar, el tiempo para cada prueba, un estimado del trabajo requerido para desarrollar cada ítem en un plan de prueba. Se va a necesitar una lista de materiales de soporte requeridos y qué pruebas van a soportar, los objetivos para cada prueba, cuán grandes se espera que sean los materiales, quién va a

desarrollar cada uno de los ítems de soporte de prueba y cuándo las pruebas de desarrollo se van a completar.

9.6 Rastreo y prueba de medida

Efectividad de prueba: cuántos defectos cada prueba descubre como una función de su tiempo de corrida. Se puede utilizar el número de defectos/hora como un modo de mérito en la selección de pruebas a incluir.

Además de LOGD y LOGT

¿Cuánto te tomó correr esta prueba?

¿Cuántos defectos encontraste?

¿Requiere intervención manual?

Registro de prueba!

Fecha de corrida de prueba, el nombre de la persona que la ejecuta, nombre y/o número de corrida de prueba, producto y configuración a probar, tiempo de inicio y fin de prueba, el número de defectos encontrados con las referencias LOGD y resultados de prueba. La forma más simple es el registro cronológico.

Módulos propensos a defecto. Defectos encontrados por clientes y por ingenieros. A mayores defectos que encuentres en prueba, mayor serán los que no encuentres.

Esto sugiere que se puede utilizar la información de prueba para evaluar el riesgo que el sistema pueda tener. Hay que sortear la información de defecto de módulo para encontrar qué módulos tienen los mayores defectos en cada prueba. Aún después de la prueba tienen defectos, el hecho de registrar, permite identificar más fácil el problema.

Hay dos formas de examinar la información dentro del módulo. SUMP da los valores de defectos removidos en cada fase, o SUMQ para las medidas de calidad de un módulo individual.

Rastreo de defectos. ¿Por qué y cómo hacer que estos defectos no vuelvan a pasar? ¿Cómo se pueden corregir estos defectos ahora?

9.7 Documentación

Parte del equipo construye la documentación de usuario mientras la otra parte se encarga de las pruebas. Con TSPi se centra en la documentación básica de usuario. Todo esto depende de la calidad y contenido funcional del producto. La cantidad de gente para cada uno de los dos grupos depende el grado de calidad y contenido funcional del producto. Conforme pasan ciclos se recomienda aumentar el número de personas para desarrollar la documentación.

En muchas formas la documentación es más importante que el código del programa. Si se hace primero la documentación, se tendrá que hacer muchos cambios y al revés, el trabajo

toma más tiempo. Como parte de la prueba de sistema, es buena idea revisar la documentación de usuario.

La guía de usuario debe tener una tabla de contenido y organizado y orientado a las necesidades del lector y no al producto. Debe incluir el "cómo comenzar", hacerlo fácil, explicando lo que el usuario puede hacer con su producto.

Hacer un glosario para palabras no estándar, presentar procedimientos de solución de problemas, incluir excepciones de mensajes de error, y procedimientos de recuperación, índice de claves. La documentación debe comenzar por un bosquejo de alto nivel.

Al revisar el documento hay que checkar:

1. Distribución
2. Terminología
3. Contenido del documento ¿Cubre todo el material requerido?
4. Decisión. Los métodos y procedimientos realmente funcionan?
5. El documento es fácil de leer?
6. Comprensión. Los lectores pueden entender lo que está escrito?

9.8 Scripts de prueba TSPi

Criterio de entrada:

Se han completado e inspeccionado los SRS y SDS.

Se han implementado e inspeccionado y probado los componentes de la unidad bajo control de configuración

Se tiene la versión anterior del producto configurada y controlada

Desarrollo de pruebas:

1. Prueba de construcción total: Primer paso de la prueba de integración revisa que el sistema se haya construido y que todos los componentes planeados están incluidos. Es una revisión completa que verifica que cada componente esté presente.
2. Procedimientos de integración para probar interfaces. A partir de la prueba de completitud se conocen los componentes están presentes y ahora se asegura que se ejecuta, se empatan apropiadamente y se pueden llamar entre interfaces para suplir funciones apropiadamente.
3. Materiales de prueba de integración. Prueba de interfaces manuales y automáticas.
4. Materiales de prueba de sistema. Materiales que prueban el software mediante condiciones normales. Se visualiza cómo se va a utilizar el sistema. Deben considerar usabilidad, rendimiento y corregir cálculos matemáticos.
5. Declaración clara de los resultados esperados en cada prueba

Construcción

1. Revisar todos los componentes para asegurarse que están a la mano y cumplen con los requerimientos de dependencia (como bases de datos).
2. Revisar los ítems abastecidos, garantizando que no falte ninguna pieza.
3. Revisar la construcción propuesta para ver si hay consistencia y totalidad.
4. Compilar y enlazar componentes del sistema.

Llenar LOGD y LOGT para cada defecto encontrado y tarea realizada respectivamente. Dependiendo de los defectos, las acciones a tomar las decide el gerente de calidad.

Integración:

1. Revisar la totalidad del producto construido. Correr una prueba de totalidad
2. Correr pruebas de integración.
3. Registrar el plan de prueba en el LOGTEST. Nombre, fecha, tiempo, sistema a prueba, etc.
4. Si se encuentran defectos, el dueño del producto registra cada defecto. El gerente de calidad determina si se continúan con las pruebas. Hay que solucionar los problemas en forma paralela.

Prueba de sistema. Registro de defectos. Hay que hacer que el gerente de calidad proceso determine si toca continuar el proceso o cancelar.

Pruebas de regresión: hacer corridas sobre los ciclos de desarrollo.

Criterio de salida:

1. Una versión de producto completa, íntegra, con pruebas de sistema y pruebas de regresión
2. Registro de prueba completo en el registro de prueba
3. Completar las entradas LOGD y un análisis de defecto para cada defecto encontrado.
4. Documentación de usuario completa realizada
5. Formato SUMP y SUMQ actualizados por el sistema
6. Copia para la carpeta del proyecto

9.9 Resumen

Hay que identificar qué partes del sistema son propensas a defecto. Estos módulos deben ser sacados de las pruebas. Prueba e integración ensamblan el sistema, asegurando que las interfaces se unen correctamente y se encuentren todos los módulos. Deben garantizar que el sistema:

1. El sistema desempeña las funciones que debe desempeñar
2. El sistema cumple sus metas de calidad
3. El sistema operará apropiadamente bajo condiciones normales
4. El sistema operará apropiadamente bajo condiciones adversas
5. El sistema comienza de manera apropiada
6. El sistema desempeña todas las funciones dadas en los requerimientos

7. El sistema se recuperará de fallas de hardware o energía
8. El rendimiento del sistema es el adecuado
9. El sistema es utilizable

En la planeación de prueba se ordenan las pruebas y se ejecutan las pruebas planeadas, registrando los resultados y el tamaño de los módulos evaluados. Las necesidades del usuario es el enfoque de la documentación, a cargo de parte del equipo.

10

Postmortem

10.1 ¿Por qué se necesita postmortem?

El software es el núcleo de casi todos los productos y servicios de las industrias; es importante casi para cualquier industria.

10.2 Lo que un postmortem puede hacer por ti

Forma estructurada de aprender y mejorar. El trabajar más no es sinónimo de una mejora consistente; deben hacerse cambios de trabajo. En esta fase se visualizan los cambios de un ciclo a otro, determinar problemas, causas y divisas procesos de prevención. Hay que decidir dónde y cómo mejorar los cambios en los procesos personales.

10.3 Propuesta de mejora de proceso

La llave es enfocarse en cambios pequeños. Cada mejora ayuda un poco, con el tiempo se verá un cambio total significativo y mejorar el proceso personal. Cualquier idea de mejora es mejor anotarla en PIP para que no se olvide.

10.4 Scripts de postmortem

PM1 y PMn se muestran en las tablas 10.1 y 10.2; el 1 necesita al instructor. Se utiliza para curbir cualquier problema o lección para futuros postmortem, se evalúa la efectividad en discusiones de anteriores postmortem para decidir qué dejar y qué cambiar. Hay que concentrarse en temas relevantes o hecho de información para los reportes.

Criterio de entrada:

1. El equipo ha completado y probado el producto
2. Los ingenieros han reunido toda la información y formatos requeridos.

Revisión de los formatos del ciclo: SUMP y SUMQ. Se revisa la información de lo que los integrantes y equipo hicieron, identificar qué funcionó y qué no, comparar el rendimiento de metas y planes, identificar las áreas de problema y necesidades de mejorar, divisar mejoras de proceso (PIPs).

Revisión de calidad evaluando. Si tuvo uno o más defectos de sistema, evaluar qué mejorar en el proceso:

Hay que responder:

¿Cómo se compara el rendimiento real con el plan?

¿Qué lecciones puedes aprender de esta experiencia?

¿Deberías usar criterios personales y de equipo en el futuro?

¿En dónde ves oportunidades de mejora y por qué?

¿Dónde tuviste problemas que pueden corregirse la siguiente ocasión?

PM, MEJORAS:

¿En dónde es que el equipo hizo las cosas bien?

¿Cómo es que su rendimiento se compara con otros equipos?

¿Qué metas definir para futuros ciclos/proyectos?

¿Dónde deberían modificar los procesos que utilizaron?

Preparar todos los PIPs para sugerencias de mejora, evaluando el rendimiento, checando las metas, identificar áreas de prioridad para la mejora.

10.5 Evaluaciones de equipo

El líder del equipo examina cada rol.

1. ¿Dónde funcionó?

2. ¿En dónde están los problemas?

3. ¿Qué metas a mejorar para el desarrollo?

Facultad del instructor:

¿En qué podría ser más útil?

¿Problemas de instalación o herramientas?

Concentrarse en experiencias constructivas.

10.6 Preparar reporte de ciclo

Describe lo que se produjo, los proceso que se utilizaron y los roles llevados a cabo. Lo que se trabajó, lo que funcionó, lo que no, responsabilidades del equipo, como rol. Reporte breve, actual y enfatizar las lecciones aprendidas. Comparar el rendimiento con el ciclo anterior y destacar cualquier tendencia.

Formato del reporte de ciclo:

Tabla de contenido

Resumen

Reportes de rol

Líder de equipo

Desarrollo

Planeación

Proceso

Calidad

Soporte

Reportes de ingeniero

10.1 Reportes de rol

¿Cómo evaluar tu rendimiento de rol? ¿Cómo se desempeñó el equipo con el rol respectivo? Lo que funcionó y no y cómo mejorarse el rol para la siguiente ocasión. Es una guía para que la siguiente persona desempeñe mejor el rol.

Líder de equipo: Rendimiento del equipo desde el liderazgo, elementos de compromiso, áreas donde hubiera sido útil un instructor, prácticas que funcionaron y no.

Genere de desarrollo. Compara el contenido del procuto con los requerimientos y evalúa la efectividad, qué elementos fueron efectivos y que no, qué estrategia tomar a futuro. Describe los pasos de de diseño e implementación que se tomaron para la usabilidad, seguridad, rendimiento, compatbilidad o instalación.

Gerente de planeación. Describe cómo se comparó el rendimiento del equipo con el plan, las tendencias de valor ganado, valor ganado del equipo y reporte. Se comparan los ciclos anteriores y con otros proyectos.

Gerente de calidad/proceso. Utiliza información de calidad real para describir el rendimiento del equipo en términos de calidad. Ciclos de desarrollo completos. Se mide la disciplina.

Se debe incluir un resumen de todos los PIPs y cómo se manejaron. Se requiere una evaluación del rendimiento de inspección del equipo y de cada ingeniero.

El gerente de soporte describe las instalaciones de soporte y anota cualquier problema a mejor. Se comenta sobre la configuración, procedimientos de control de cambios, cómo se trabajó. La información debe incluir cambios de información de actividad, cambios tardíos, información cómo debo manejar mejor los cambios.

Gerente de soporte -> estrategia de reúso, porcentaje de reuso. Sugiere cómo mejorar las estrategias de reuso.

10.1 Reportes de ingeniero

Se debe reportar rendimiento personal incluida la cantidad de trabajo, cómo trabajar mejor a futuro.

10.1 Traduciendo el reporte

Definición de compromisos. El líder de equipo distribuye, el gerente de calidad proceso dirige el reporte borrador, luego los ingenieros hacen correcciones.

10.1 Evaluaciones de rol

Cada miembro del equipo prepara una copia personal de PEER, dando el punto de vista del rendimiento del equipo (porcentual).

10.1 Sugerencias de evaluación de rol

Actuar como si los comentarios fueran a la persona que llevaba el rol. Cada evaluación se hará pública. Tener cuidado de que las evaluaciones no avergüencen a nadie. Las evaluaciones son para ayuidar a la siguiente persona con el rol.

Criterio de salida:

1. El equipo ha producido un producto de alta calidad con la documentación requerida
2. El producto completo está bajo control de configuración
3. La información de proceso ha sido evaluada y los PIPs han sido sometidos y completos.
4. Las evaluaciones de rol están completas y sometidas.
5. El reporte de ciclo de desarrollo está completo y con copia para todos
6. Todos los formatos TSPi se han completado.
7. Cuaderno de proyecto actualizado con todo

10.1 Resumen

El postmortem provee una forma estructurada de mejorar el proceso personal y de equipo. Se empieza con el líder para especificar lo que funcionó, lo que no y se compara lo planeado y lo real. Se identifica lo que se trabajó, qué problemas iba. Se revisa del líder a los demás. Concentrarse en áreas de mejora.

Mantener el reporte breve y actual. Hay que comparar el rendimiento con anteriores ciclos de desarrollo y marcar tendencias. Se tiene la opción de hacer comentarios adicionales del tema pero que sean de forma constructiva