UCI

UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

Facultad 9

Estrategia del Proceso de Software en Equipo para mejorar la calidad de los productos en el Polo PetroSoft.

Polo Productivo
Soluciones Informáticas para la Industria Petrolera
PetroSoft

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

AUTORA: Yadnira Ramírez Rodríguez

TUTOR: Ing. Enrique Pérez Rodríguez

CO_TUTORA: Ing. Saily Porta García

ASESORA: Karolay Rodríguez Alpajón

Ciudad de la Habana, Julio de 2008
"Año 50 de la revolución"

Frase

"La cooperación es la convicción plena de que nadie puede llegar a la meta si no llegan todos."

Virginia Burden

Dedicatoria

A mis padres y mis abuelos por ser mi principal inspiración de querer llegar a convertirme en una persona preparada, digna y trabajadora. Por todo el esfuerzo, entrega, amor y apoyo incondicional que me han dado siempre. A ustedes especialmente va dedicado todo mi esfuerzo.

Agradecimientos

A mami y a papi por su apoyo en todos estos años, por llenarme de amor y fuerzas cada día y guiarme en todos mis pasos.

A mis abuelos por darme su amor, dedicación, por sus deseos de verme graduada y sentirse siempre tan orgullosos de las cosas buenas que hago.

A mi novio por estar siempre a mi lado en los buenos y malos momentos apoyándome, dándome fuerzas y amor para seguir adelante.

A mi tutor Enrique Pérez Rodríguez por todo su apoyo.

A mi cotutora Saily Porta García por estar ahí cuando más la necesité.

A Fidel y a la Revolución por darme la oportunidad de hacer realidad mi sueño.

A todas aquellas personas que me han apoyado a lo largo de mi vida y que han estado junto a mí en los buenos y malos momentos. A todos ustedes: Muchas gracias, no saben cuanto ha significado que hayan estado ahí para mí.

Resumen

Las sociedades se enfrentan a una transformación global de sus planteamientos, consecuencias del desarrollo de las tecnologías de la información y las comunicaciones (TIC), estos avances han contribuido a que el conocimiento sea considerado como un nuevo recurso, generador de importantes ventajas competitivas.

La Universidad de las Ciencias Informáticas (UCI) fue creada con el objetivo de vincular la docencia y la producción, en ella existe un alto porcentaje de polos productivos donde se encuentran estudiantes y profesores trabajando en conjunto para desarrollar productos de software, uno de estos es el Polo PetroSoft, que cuenta con varios proyectos los cuales no tienen un modelo para procesos de software que guíen el trabajo en equipo lo que trae consigo un conjunto de deficiencias que afectan la calidad de los productos. Teniendo en cuenta los problemas detectados se decidió que se le diera solución a través de una estrategia del Proceso de Software en Equipo (TSP).

Este trabajo está estructurado en tres Capítulos. El primero, contiene la fundamentación teórica de esta investigación, en la cual son expuestos los principales conceptos y argumentos que esclarecen el objeto de estudio. En el segundo se describe una estrategia del TSP para mejorar la calidad de los productos en el Polo PetroSoft y en el tercer Capítulo se realiza la validación de la estrategia a partir del método de expertos.

Palabras Claves

Software, Producción, Estrategia, Calidad, Procesos

Índice

Índice de Tablas y Figuras.

Introducción.

En un mundo de cambios constantes y competencia global, las organizaciones de desarrollo de software son presionadas para alcanzar mayor eficiencia con menores costos. Para poder lograr este objetivo, es necesario adoptar una forma de trabajo que permita entender, controlar, comunicar, mejorar, predecir y certificar el trabajo realizado. Actualmente existe una gran diversidad de opciones relacionadas con procesos de desarrollo. Constantemente se escuchan diferentes acrónimos como CMM, CMMI, RUP, ISO, PSP, TSP, que causan confusión, principalmente debido a la mala interpretación de los mismos.

Hasta hace poco tiempo, la producción de software era realizada con un enfoque artístico y no industrial. Ante la constante presencia de proyectos fallidos, y con el objetivo de mejorar la calidad de los productos, en los últimos años las organizaciones introdujeron los métodos de ingeniería de software. A partir de estos, se formalizó el enfoque de ingeniería de producto para desarrollar software. Factores como la globalización han obligado a las organizaciones a contar con marcos de trabajo que las ayuden hacer las cosas de la manera más eficiente.

Siempre que para alcanzar algún fin deseado se necesite ejecutar una serie de acciones, y estas tengan cierto orden, dependencias, roles responsables, resultados, tiempos de ejecución y herramientas de apoyo, se estaría hablando de procesos, que pueden ser predefinidos y personalizados. Un proceso de software efectivo habilita a la organización a incrementar su productividad al desarrollar software: Permite estandarizar esfuerzos, promover reuso, repetición y consistencia entre proyectos. Proporciona la oportunidad de introducir mejores prácticas de la industria. Permite entender que las herramientas deben ser utilizadas para soportar un proceso y establece la base para una mayor consistencia y mejoras futuras.

Actualmente existe una gran variedad de modelos para procesos de software. Se pueden entender más fácilmente si se clasifican en dos tipos: genéricos y específicos. Los genéricos abarcan todo los procesos relacionados con el desarrollo de software. Se deben usar como referencia para definir procesos en una organización y para autoevaluación. Son medios para evaluar que tan bien o mal está la organización.

Dentro de estos se encuentran:
CMM (Capability Maturity Model) - Modelo de Madurez de Capacidades.
CMMI (Capability Maturity Model Integration) - Modelo de Madurez de Capacidades Integrado.
ISO (International Standards Organization) - Organización Internacional de Estándares.

ISO/IEC 15504 (Organización Internacional de Estándares).
MoProSoft.

Los modelos específicos son enfocados a la ingeniería de productos de software. Se dice cómo se deben hacer las cosas y se usan como guía para ejecutar proyectos. Estos son:
RUP (Rational Unified Process) – Proceso Unificado de Desarrollo.
PSP (Personal Software Process) – Proceso de Software Personal.
TSP (Team Software Process) – Proceso de Software en Equipo.

¿Qué procedimiento se debe usar para llevar a cabo el desarrollo de un software con mejor calidad? Todo desarrollo de software es riesgoso y difícil de controlar, por lo que es de vital importancia la aplicación de procedimientos para lograr una total satisfacción de clientes y desarrolladores.

Sin embargo, muchas veces no se toma en cuenta el utilizar un procedimiento adecuado, sobre todo cuando se trata de proyectos pequeños de dos o tres meses. Lo que se hace con este tipo de proyectos es separar rápidamente el aplicativo en procesos, cada proceso en funciones, y por cada función determinar un tiempo aproximado de desarrollo.

Cuando los proyectos que se van a desarrollar son de mayor envergadura, ahí sí toma sentido el basarse en un procedimiento de desarrollo, y se empieza a buscar cuál sería el más apropiado para ese caso. Lo cierto es que muchas veces no se encuentra el más adecuado y se termina por hacer o diseñar un procedimiento propio, algo que por supuesto no está mal, siempre y cuando cumpla con el objetivo. Los proyectos en problemas son los que salen del presupuesto, tienen importantes retrasos, o simplemente no cumplen con las expectativas del cliente.

La Universidad de las Ciencias Informáticas, es una institución que está actualmente en desarrollo y por ello no cuenta con todo el personal capacitado para la producción por lo que han aparecido diversas dificultades en el desarrollo de los productos del software. Esto no solamente viene ligado a la falta de experiencia por parte del personal sino también a la falta de procedimientos para guiar al personal que se encuentra involucrado en el desarrollo de los mismos. Debido a esta falta de procedimientos es que se decide investigar una estrategia para aplicar Procesos de Software en Equipo al Polo PetroSoft.

El uso de planes y procedimientos trae orden y eficacia a cualquier trabajo y les permite a los obreros concentrarse para producir un producto superior. Un esfuerzo disciplinado quita la pérdida, error, e ineficacia, liberando los recursos financieros para mejores usos. Debido a que generalmente los ingenieros de software no son instruidos en planificación, seguimiento o medición de calidad, ellos

normalmente no realizan un seguimiento de su trabajo, y la calidad del software es raramente moderada.[1]

TSP es una metodología para dirigir el trabajo de mejora y desarrollo de software además de establecer un entorno donde el trabajo efectivo de equipo sea normal y natural. Para poder aplicar el Proceso de software en Equipo hay que tener una previa capacitación en PSP.

La UCI hoy día produce una gran cantidad de software y es por eso que necesita de procedimientos que guíen el trabajo del personal, ya que la no utilización de los mismos conlleva a que en muchos casos los tiempos de desarrollos que se emplean sean mayores a los planificados, lo que trae consigo que la calidad del producto no sea la esperada y aumenten los costos.

Las condiciones en las cuales se desenvuelve la producción de software en el Polo PetroSoft conllevan a que surja el siguiente problema científico: **¿Cómo generar una estrategia del Proceso de Software en Equipo (TSP) que resuelva los problemas en los proyectos productivos del Polo Petrosoft?**

Para darle solución a este problema se hizo necesario que el objetivo general estuviera encaminado a **plantear una estrategia del TSP que permita mejorar la calidad de los productos en los proyectos productivos del Polo PetroSoft.**

Para darle cumplimiento a dicho objetivo fue necesario centrar la investigación en **los procedimientos de desarrollo que plantea el Proceso de Software de Equipo (TSP),** lo cual constituye el objeto de estudio.

Las estrategias que serán planteadas como solución de esta investigación tendrán su incidencia final en **los equipos de desarrollo de los proyectos productivos del Polo PetroSoft,** lo cual determina el campo de acción de la misma.

Para darle cumplimiento al objetivo trazado se determinó que las **tareas a realizar en esta investigación** estarían dirigidas a:

1. Localizar la existencia del problema en el uso de las etapas de TSP en el desarrollo de software.
2. Investigar sobre los métodos de TSP utilizados en el desarrollo de software en los proyectos del Polo PetroSoft.
3. Investigar sobre el estado del arte y la aplicación de TSP en el desarrollo de software.
4. Hacer entrevistas a líderes y personal involucrados en el Polo PetroSoft.
5. Elaborar una estrategia del Proceso de Software en Equipo para el Polo PetroSoft.

El desarrollo exitoso de las tareas expuestas anteriormente contribuirá al cumplimiento de la **Hipótesis** de esta investigación:

Si se cuenta con una estrategia del Proceso de Software en Equipo que permita mejorar la calidad de los productos de manera eficiente entonces los productos desarrollados en el polo productivo de Petrosoft tendrán la calidad requerida.

Para el desarrollo de la investigación se utilizaron métodos teóricos y empíricos. Dentro de los teóricos están los de análisis y síntesis, el histórico-lógico y el hipotético-deductivo y dentro de los empíricos la entrevista. A continuación se especifica el por qué de la selección de los mismos.

Métodos teóricos:

Histórico-lógico: En la primera fase de esta investigación se desarrolla un estudio del estado del arte de la problemática analizada; revisando de forma crítica cada uno de los documentos para poder resaltar la importancia de las diferentes estrategias de desarrollo de software que se llevan a cabo en la actualidad, así como la eficiencia y/o deficiencias de cada una de estas.

Hipotético deductivo: Permite a partir del problema concreto plantear los objetivos específicos e hipótesis que en el transcurso de la investigación son resueltos siguiendo métodos científicamente bien fundamentados.

Analítico Sistémico: Nos plantea el problema como un todo, estudio de metodologías, esquemas del uso de las mismas en el desarrollo de software y cada uno de los métodos y herramientas que se utilizan.

Métodos Empíricos:

Entrevista: La entrevista se aplicó a conocedores del tema, personal capacitado en la muestra, o sea en un proyecto del polo productivo Petrosoft en la facultad, para recopilar información sumamente importante para la investigación.

Con el objetivo de lograr resultados que le dieran credibilidad a esta investigación fue necesario definir una población, de la cual se extrajo una muestra que fue analizada. De los resultados obtenidos a partir de este análisis se pudo inferir cómo se iba a comportar dicha población. A continuación se presentan:

Población: Los proyectos que integran el Polo PetroSoft.

Muestra: El proyecto de Sistemas Automáticos de Control de Gestión de Indicadores de Refinación (SACGIR) del Polo PetroSoft.

Unidad de estudio: Procedimientos de desarrollo que plantea el proceso de software en equipo.

La selección de esta muestra se llevó a cabo haciendo uso de un muestreo no probabilístico, en este caso: el muestro intencional, ya que este tipo de muestreo posibilita escoger los integrantes de la muestra, por lo que nos permite seleccionar explícitamente los elementos que son representativos o con posibilidad de brindar mayor información.

Capítulo 1

Fundamentación Teórica

1.1 Introducción

"La calidad de un software está dada por la calidad de sus procesos usados para desarrollarlo y mantenerlo". [2]

La meta de la ingeniería de software es construir productos de software, o mejorar los existentes; en ingeniería de procesos, la meta es desarrollar o mejorar procesos. Un proceso de desarrollo de software es un conjunto de personas, estructuras de organización, reglas, políticas, actividades y sus procedimientos, componentes de software, metodologías, y herramientas utilizadas o creadas específicamente para definir, desarrollar, ofrecer un servicio, innovar y extender un producto de software.

Para producir software es necesario tener en cuenta una serie de procedimientos que sin duda ayudan a que los productos sean terminados en tiempo y con una mayor calidad.

Son precisamente los procedimientos de desarrollo de software el objetivo esencial sobre el cual gira la investigación realizada para el desarrollo del Capítulo 1, pues constituye la base para el análisis y entendimiento del objeto de estudio que rige la misma, el cual se centra en los procedimientos de desarrollo que plantea el TSP.

1.2 Conceptos asociados al dominio del problema

A continuación se describen detalladamente un grupo de conceptos asociados al dominio del problema, con el objetivo de proporcionarle al lector la vía más apropiada para una mayor comprensión de los temas que serán abordados en este Capítulo 1.

1.2.1 Calidad de Software

La calidad de Software "es la concordancia con los requerimientos funcionales y de rendimiento explícitamente establecidos, con los estándares de desarrollo documentados y con las características implícitas que se esperan de todo software desarrollado profesionalmente". [1]

1.2.2 Estándar de calidad

Los estándares de calidad se definen como un "conjunto de criterios de desarrollo que guían la forma en que se aplica la ingeniería del software. Los estándares suministran los medios para que todos los procesos se realicen de la misma forma y son una guía para lograr la productividad y la calidad". [16]

Se puede decir que los requisitos del software constituyen la base para medir la calidad y que los estándares o metodologías constituyen la guía para ello ya que tienen como único fin producir software de alta calidad.

A continuación se describirán brevemente algunos estándares a nivel de proceso y de producto:

Estándares de calidad a nivel de proceso:

IEEE/EIA 12207:
Este estándar está formado por ISO/IEC 12207, que proporciona un ambiente de trabajo común para el desarrollo y la administración del software y por IEEE/EIA 12207.0. Éste último contiene conceptos y guías para propiciar un mejor entendimiento y aplicación del mismo, así como también establece las bases para la producción de software que serán útiles posteriormente.

ISO/IEC 20000:
Es el primer estándar internacional creado para auditar, certificar y administrar los servicios de la tecnología de la información (TI). Su implementación asegurará prácticas de trabajo proactivas capaces de entregar altos niveles de servicio al cliente que satisfagan las necesidades del negocio. ISO/IEC 20000:

- Está integrado a los estándares ISO de ingeniería de sistemas y de software.
- Ofrece una certificación organizacional.
- Muestra cómo administrar y mejorar la TI.
- Establece un criterio de auditoría.
- Suministra a los auditores un documento estándar, el cual es usado para medir la conformidad de la misma.

Este estándar está dividido en 2 partes, la primera es la Especificación de la Gestión del Servicio de TI y ayuda a iniciar, implementar o mantener la gestión del servicio de TI en la organización. Define los procesos y suministra el criterio de evaluación y recomendaciones para el responsable de la Gestión de Servicios de TI. También promueve la adopción de un proceso integrado que entrega servicios que cumplen con los requerimientos del negocio y del cliente y por último define los requerimientos para un proveedor de servicios administrados.

La segunda parte es el código de práctica y describe las mejores prácticas de los procesos de Gestión de Servicios. Esta parte representa un consenso de la industria sobre la guía para los auditores y provee mejoras en el servicio de planificación.

ISO 90003:2004:

Provee una guía para las organizaciones respecto de la aplicación de ISO/IEC 9001:2000 en la adquisición, suministro, desarrollo, operación y mantenimiento de software y servicios de soporte, pero no agrega o cambia los requerimientos de dicho estándar. Sus guías no tienen el propósito de ser utilizadas como criterio de evaluación en una certificación de SGC (Sistema de Gestión de la Calidad).

Se recomienda la aplicación de ISO 90003:2004 para un software que:

- Forme parte de un contrato comercial con otra organización.
- Sea un producto disponible para un sector del mercado.
- Sea usado para procesos de una organización.
- Esté relacionado a servicios de software.

ISO 90003:2004 está formado por 5 capítulos:

- Sistema de Gestión de la Calidad.
- Responsabilidad de la Dirección.
- Gestión de los Recursos.
- Realización del Producto.
- Medida, Análisis y Mejora. Los cuales especifican las actividades que deben ser consideradas cuando se implemente el SGC. Es independiente de la tecnología, modelos del ciclo de vida, procesos de desarrollo, secuencia de actividades y estructura organizacional utilizada en la organización.

Estándares de calidad a nivel de producto:

ISO/IEC 9126-1:2001 – Modelo de Calidad:

El Modelo de Calidad se encarga de describir el modelo de calidad del producto de software. Este estándar presenta tres niveles:

- Características.
- Subcaracterísticas.
- Métricas.

Las características de calidad interna y externa que presenta son aplicables a todo tipo de software estas son las siguientes:

- Funcionalidad.
- Confiabilidad.
- Facilidad de uso.
- Eficiencia.
- Facilidad de mantenimiento
- Portabilidad.

La calidad externa e interna son las que se encargan de evaluar el software según las necesidades del usuario basándose en el comportamiento del producto y de evaluar el total de atributos que un software debe satisfacer basándose en las condiciones internas, respectivamente.

ISO/IEC 9126-1:2001 permite especificar y evaluar la calidad del software según la adquisición, requerimientos, desarrollo, uso, evaluación, soporte, mantenimiento, aseguramiento de la calidad, y auditoría del software.

SQuaRE (Software Quality Requirements and Evaluation):

No es más que una nueva serie de normas basadas en ISO 9126 y en ISO 14598 (Evaluación del software) cuyo objetivo es coordinar y armonizar el contenido de ISO 9126 y de ISO 15939:2002 (Measurement Information Model). El Measurement Information Model presenta un modelo de información que ayuda a determinar qué se debe especificar durante la planificación y evaluación de la medición mediante la recopilación, preparación y análisis de los datos.

SQuaRE incluye un estándar de requerimientos de calidad. Está compuesto por 14 documentos agrupados en 5 tópicos:

- Administración de la Calidad.
- Modelo de Calidad.
- Medidas de Calidad.
- Requerimientos de Calidad.
- Evaluación de la Calidad.

1.2.3 Proceso de Software Personal (PSP)

"Es un proceso de auto mejoramiento diseñado para ayudar a controlar, administrar y mejorar la forma en que se trabaja individualmente. Está estructurado por formularios, guías y procedimientos para desarrollar software. Si es usado apropiadamente, brinda los datos históricos necesarios para trabajar mejor y lograr que los elementos rutinarios del trabajo sean más predecibles y eficientes."[3]

1.2.4 Proceso de Software en Equipo (TSP)

"Es un proceso que al igual que el PSP, está basado en el modelo CMM, TSP está diseñado para ayudar a controlar, administrar y mejorar la forma en que trabaja un equipo de software. Al igual que PSP, está estructurado por formularios, guías y procedimientos para desarrollar software. [2]

1.2.5 Proceso de Software en Equipo (TSPi)

Es una versión a escala reducida del TSP, es una guía paso a paso para lograr un proyecto de software en equipo. Enseña como aplicar conocimientos de ingeniería de software y procesos en un ambiente de trabajo en equipo. Define claramente los roles que cada miembro debe desempeñar, así como sus responsabilidades. Muestra qué se debe hacer, cómo hacerlo y cuándo hacerlo. Permite practicar y desarrollar una buena actitud de equipo de trabajo.[2]

1.2.6 Modelo de Madurez de las Capacidades (CMM)

"El CMM es un ambiente de trabajo (un framework) de madurez de procesos de software" [3]

Es una forma ordenada para las organizaciones de determinar las capacidades de sus procesos actuales y establecer prioridades en su mejora a través del establecimiento de 5 niveles de prioridad progresivos.

Los 5 niveles que presenta son:

- Inicial.
- Repetible.
- Definido.
- Gestionado.
- Optimizado.

1.3 Objeto de estudio

1.3.1 Descripción General

La introducción de esta investigación refleja que el objeto de estudio sobre el cual versa este trabajo de diploma está encaminado en los procedimientos de desarrollo que plantea el Proceso de Software en Equipo (TSP), enmarcado específicamente en el Polo PetroSoft.

TSP fue creado en 1999 por Watts Humphrey con el fin de proporcionarles a los estudiantes de ingeniería de software una visión total del ciclo de vida del software.

Para un mejor entendimiento del mismo se hace necesario que se comiencen planteando un conjunto de conceptos que definen al proceso de software de forma general y al proceso de software en equipo de forma particular, pues estos constituyen la base de los posteriores avances que se han ido realizando en función de la organización y desarrollo de los mismos.

Se parte de que un proceso de software es "el conjunto completo de actividades necesarias para convertir los requisitos de usuario en un conjunto consistente de artefactos que conforman un producto de software, y para convertir los cambios sobre esos requisitos en un nuevo conjunto consistente de artefactos"[4]

Para poder llevar a cabo este conjunto de actividades para mantener un producto de software se necesita del proceso de software en equipo pero antes se debe de tener una previa capacitación en el proceso personal de software del cual Humphrey plantea "El Proceso Software Personal (PSP) muestra cómo aplicar métodos avanzados de ingeniería a las tareas diarias de cada individuo, proporciona métodos detallados de planificación y estimación, muestra a los implicados en el proceso cómo controlar su rendimiento y explica cómo los procesos definidos guían su trabajo."[3]

PSP ofrece la administración y control del proceso al Ingeniero de Software. Con PSP los ingenieros desarrollan software usando una propuesta estructurada y disciplinada.

Los ingenieros se ocupan de:

- Seguir un proceso definido.
- Planificar, medir y seguir su trabajo.
- Administrar la calidad del producto.
- Aplicar aspectos cuantitativos para mejorar los procesos de trabajo personales.

TSP es un "proceso para formar y guiar equipos de ingenieros que desarrollan Software. Conjunto de Procesos definidos y estructurados que enfatizan el balance entre procesos, productos y trabajo en equipo. Indica qué hacer en cada fase del ciclo de desarrollo del proyecto y muestra cómo aplicar prácticas de ingeniería de software conocidas, en un ambiente de trabajo en equipo."[2]

Como bien se plantea anteriormente el TSP está formado por procesos, productos y trabajo en equipo. Para un mayor entendimiento de lo que plantea TSP se darán un conjunto de conceptos asociados a él.

Se parte de que un proceso es "un conjunto de actividades o acciones interrelacionadas entre sí que, a partir de una o varias entradas de información, materiales o de salidas de otros procesos, dan lugar a una o varias salidas también de materiales (productos) o información con un valor añadido". [5]
De igual forma se plantea que un proceso es "un conjunto de actuaciones, decisiones, actividades y tareas que se encadenan de forma secuencial y ordenada para conseguir un resultado que satisfaga plenamente los requerimientos del cliente al que va dirigido".[5]

Los productos son un conjunto de atribuciones tangibles e intangibles que incluye el empaque, color, precio, prestigio del fabricante, prestigio del detallista y servicios que prestan este y el fabricante.[6]
Un equipo consiste en al menos dos personas que trabajan juntos hacia una meta / objetivo / misión común, donde cada persona tiene asignados roles o funciones específicas y además completar la misión requiere alguna forma de dependencia entre los miembros del grupo".[2]

En los equipos de desarrollo de software, aunque las personas que lo conforman suelen tener distintas especialidades es necesario que todos trabajen en forma cooperativa en busca de un objetivo en común. Para realizar trabajo en equipo se necesita desarrollar habilidades específicas que permitan una correcta cohesión entre sus integrantes.[2]

Entre estas habilidades se encuentra la existencia de una meta definida y realista, la obtención de recursos adecuados para el trabajo, integrantes expertos y comprometidos con el logro del objetivo, trabajo disciplinado y constante. El desarrollo de estas habilidades es entre otras cosas lo que TSP ofrece al mercado del software.

TSP es un proceso muy importante ya que junto con PSP y CMM conforman las herramientas ideales para ayudar a las organizaciones en la formación de sistemas de software de alta calidad. TSP es un proceso intermedio entre los dos anteriores ya que CMM mejora el proceso de toda la organización y se enfoca en la forma de administrar los proyectos mientras que PSP mejora las habilidades personales, crea un hábito de desarrollo y se enfoca a las personas. TSP conecta las dos anteriores ya que mejora el desempeño de los equipos y se enfoca en los productos y en los equipos.

Una característica de gran importancia es que mientras PSP y CMM proveen una lista de las habilidades necesarias para llevar a cabo un proyecto de software efectivo, TSP es una guía para su realización. TSP está basado en un modelo secuencial incremental que divide el proceso de software en un conjunto de ciclos de desarrollo donde cada ciclo incluye la producción de software que cumple algunas características de los requerimientos de software. El ciclo final reúne la integración y las pruebas del sistema completo.

TSP es un método de establecimiento y mejora del trabajo en equipo para procesos software. Proporciona directrices para ayudar a un equipo a establecer sus objetivos, a planificar sus procesos y a revisar su trabajo con el fin de que la organización pueda establecer prácticas avanzadas de ingeniería y así obtener productos eficientes, fiables y de calidad.

TSP se basa en los siguientes principios:

- Los técnicos conocen muchas cosas sobre su trabajo y pueden realizar las mejores planificaciones. Cuando son ellos quienes planifican su propio trabajo, se encuentran comprometidos con el plan.
- Un seguimiento preciso de un proyecto requiere planes bien detallados y datos precisos. Únicamente el personal que realiza el trabajo es capaz de recoger con precisión dichos datos.
- Para minimizar el tiempo del proyecto, los ingenieros deben equilibrar su carga de trabajo.
- Para maximizar la productividad, el primer foco de atención debe ser la calidad.

1.3.2 Roles del Equipo:

Otro elemento importante del TSP es que para crear un equipo de trabajo efectivo se necesita la determinación y definición clara de roles para cada uno de los integrante del mismo. Estos roles a grandes rasgos son los siguientes:

Líder del proyecto:

El líder del proyecto guía al equipo y se asegura de que los ingenieros reporten las estadísticas de avance y que se complete el trabajo en la forma que fue planeado. En general es el responsable de que el equipo funcione adecuadamente.

Encargado de Desarrollo:

El encargado de desarrollo dirige al equipo en los asuntos de diseño y desarrollo del producto. Es el principal involucrado con la producción y funcionalidad de un producto de alta calidad de desarrollo.

Encargado de Planeación:

Guía al equipo en la planeación y seguimiento del producto. Su objetivo principal es guiar al equipo en la producción de un plan detallado y guiar el progreso del producto de acuerdo al plan.

Encargado de Procesos y Calidad:

Ayuda al equipo en la definición de los procesos necesarios para la realización del sistema y en el establecimiento y administración de planes de calidad que den eficiencia al proyecto.

Encargado de Soporte:

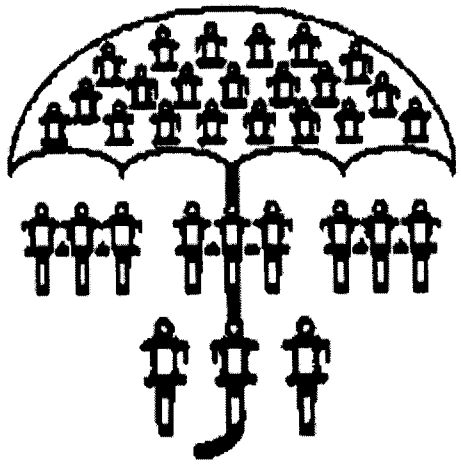
Ayuda al equipo a determinar, obtener y administrar las herramientas necesarias para cubrir las necesidades de tecnología y soporte administrativo. Es decir, que el proyecto esté propiamente soportado y controlado.

La formación de equipos requiere que los miembros entiendan qué y cómo hacer el trabajo; y que sus planes sean alcanzables. Para hacer un trabajo disciplinado, los ingenieros necesitan procesos operacionales que definan cómo es realizado el trabajo. El proceso operacional es semejante a un script y es diseñado para ser usado por los miembros del equipo. El TSP provee un proceso operacional definido que guía a los ingenieros y directores en los pasos para la construcción de un equipo. Con un proceso definido y un plan que sigue ese proceso, los ingenieros son eficientes. TSP provee los procesos operacionales necesarios para formar los equipos de ingenieros, establecer un ambiente de trabajo efectivo y guiar a los equipos en la realización del trabajo.

La conformación de equipos es uno de los requisitos de la mayor parte de los proyectos de ingeniería. Aunque ciertos proyectos pequeños de software pueden ser realizados en forma individual, la complejidad de los sistemas actuales y la demanda de cortos tiempos de entrega es tal, que ya no es práctico para una sola persona encargarse de proyectos de software.

TSP es una serie de métodos que pueden ayudar a los equipos de ingenieros a desarrollar sistemas. CMM provee la estructura de mejoramiento necesaria para el trabajo de Ingeniería. PSP provee la

disciplina de Ingeniería que los Ingenieros necesitan para utilizar un proceso definido, planificado y medido.



CMM	Mejora la capacidad de la organización y el enfoque de la dirección.
TSP	Mejora el rendimiento del equipo, genera productos de calidad a tiempo y dentro del presupuesto.
PSP	Genera habilidades individuales y disciplina

Figura1: Relación CMM, TSP y PSP [2]

El entrenamiento en PSP es requerido para suministrar a los ingenieros del conocimiento necesario para utilizar TSP. El entrenamiento en PSP incluye:

- Aprender cómo realizar un planeamiento detallado.
- Recopilar y utilizar los datos del proceso.
- Desarrollar planes valuados.
- Medir y administrar la calidad del producto.
- Definir y utilizar los procesos operacionales.

1.3.3 Estrategias de TSP

TSP proporciona una serie de estrategias para facilitar el trabajo en equipo:

1. Brinda un marco de trabajo simple basado en PSP
2. Usa problemas modestos, bien definidos.
3. Desarrolla productos en muchos ciclos.
4. Establece estándares de métricas para calidad y desempeño.
5. Brinda una definición de roles detallada.
6. Usa evaluación de roles y de equipo.
7. Es un proceso disciplinado.
8. Brinda guías en la solución de problemas de los equipos.

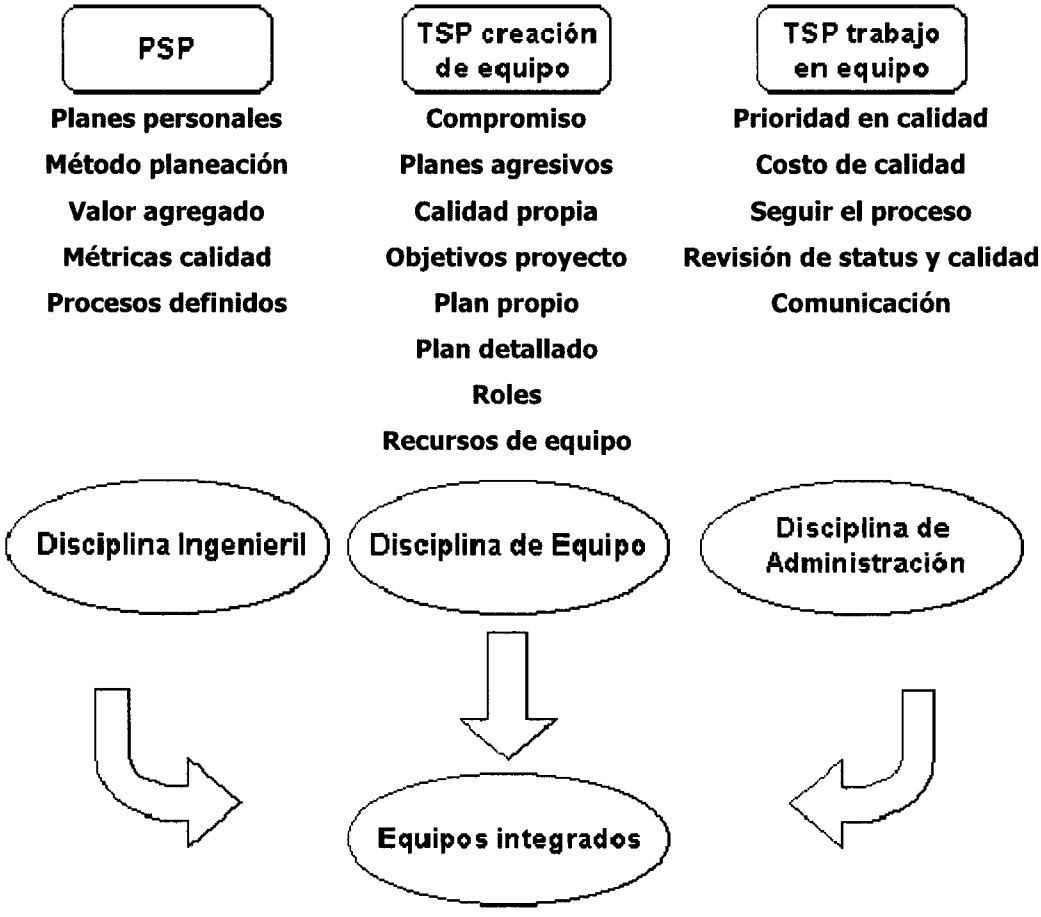


Figura 2. Estructura del TSP [2]

"Cuando un proyecto de software falla, se debe principalmente a problemas de trabajo en equipo y no a problemas técnicos".[2]

1.3.4 Problemas en el equipo.

Liderazgo inefectivo.

Sin un líder efectivo, los equipos tienen problemas para acometer sus planes y mantener su disciplina personal. Pero pocas personas son por naturaleza líderes, la mayoría necesita desarrollar las habilidades y practicar su uso. Los ingenieros necesitan ver un liderazgo efectivo en acción para conocer qué habilidades poner en práctica.

Falta de compromiso o cooperación:

Uno o varios miembros del equipo pueden no estar dispuestos a trabajar cooperadamente. Los equipos necesitan tratar este problema. La presión colectiva contribuye en gran medida, pero si la persona continúa intratable se debe discutir con el instructor.

Falta de participación:

Los miembros tienen diferentes conocimientos, habilidades y motivaciones, por lo que cada uno tiene una contribución diferente en el funcionamiento del equipo. Si alguien no está haciendo un esfuerzo serio, el espíritu del equipo sufre.

Demora:

Algunos equipos:

- No establecen fechas límites o metas.
- Las establecen pero no las cumplen.
- No chequean la ejecución y fallan por no tomar decisiones en tiempo y forma.

Causas principales:

- Liderazgo inexperto.
- Falta de metas claras.
- Falta de proceso definido y de un plan.

Baja Calidad:

Esto se debe principalmente a:

- Pruebas muy extensas.
- Planes retardados.
- Horarios prolongados.
- Productos finales no satisfactorios.

Cambios sutiles de funciones:

Durante el diseño y la implementación, con frecuencia los ingenieros encuentran formas de mejorar sus productos. Este problema es complejo porque no existe una clara división entre las funciones que provienen de las interpretaciones de los requerimientos y las que son adiciones verdaderas a estos.

Evaluaciones colectivas no efectivas:

Las evaluaciones colectivas pueden ser invaluables para los equipos de estudiantes.

Desventajas: Los estudiantes no les gustan evaluar a sus compañeros y raramente lo hacen con la calidad requerida.

Consecuencias: Los estudiantes sienten que las calificaciones en los cursos de equipo no son enteramente justas, particularmente los estudiantes altamente motivados. Esto puede:

- Provocar competencia entre los miembros del equipo
- Reducir su disposición a colaborar completamente.

Fases del Ciclo de Vida de TSP	Actividades
Lanzamiento	-Revisión de objetivos a perseguir. -Asignación de equipos y roles al personal. -Se describen las necesidades del cliente. -Se establece las metas individuales y del equipo.
Estrategia	-Crear un diseño conceptual para el producto. -Se establece la estrategia de desarrollo: se decide que será producido en cada ciclo. -Se hacen estimaciones iniciales de esfuerzos y tamaño. -Se establece un plan de administración de la configuración. -Se reutiliza el plan anterior. -Se establecen riesgos de administración.
Planeamiento	-Estima el tamaño de cada artefacto a ser desarrollado. -Se identifican las tareas: se estima el tiempo para completar cada tarea; se asignan tareas a los miembros del equipo. - Hacer un cronograma semanal para tareas terminadas. -Hacer un plan de calidad.
Requerimientos	-Se analizan las necesidades del cliente y se entrevistan. -Se especifican los requerimientos.

	-Se hace inspección de los requerimientos. -Se diseña un plan de pruebas del sistema.
Diseño	-Se crea un diseño de alto nivel. -Se especifica el diseño. -Se inspecciona el diseño. -Se desarrolla un plan de pruebas de integración.
Implementación	-Se usa PSP para implementar módulos y unidades. - Se crea el diseño detallado de los módulos y unidades. - Se revisa el diseño. - Se convierte el diseño al código. -Se inspecciona el código -Se compilan y prueban los módulos y unidades. -Se analiza la calidad de los módulos/unidades.
Pruebas	-Se construye e integra el sistema. -Se llevan a cabo las pruebas del sistema. -Se produce la documentación de usuario.
Después de la muerte	-Análisis de resultados. -Se escribe el reporte del ciclo. -Se producen evaluaciones de pares y equipo.

Tabla 1: Fases del ciclo de vida de TSP

TSPi usa los ciclos de desarrollo múltiples para elaborar el producto final.

- Iniciación: el instructor describe los objetivos generales del producto. El equipo sigue los pasos del proceso: estrategia, planificación, requerimientos, diseño, implementación, prueba y postmortem.
- Los ingenieros repiten los pasos y amplían el producto base obtenido. Si hay tiempo, se puede continuar ampliando el producto en ciclos subsiguientes.

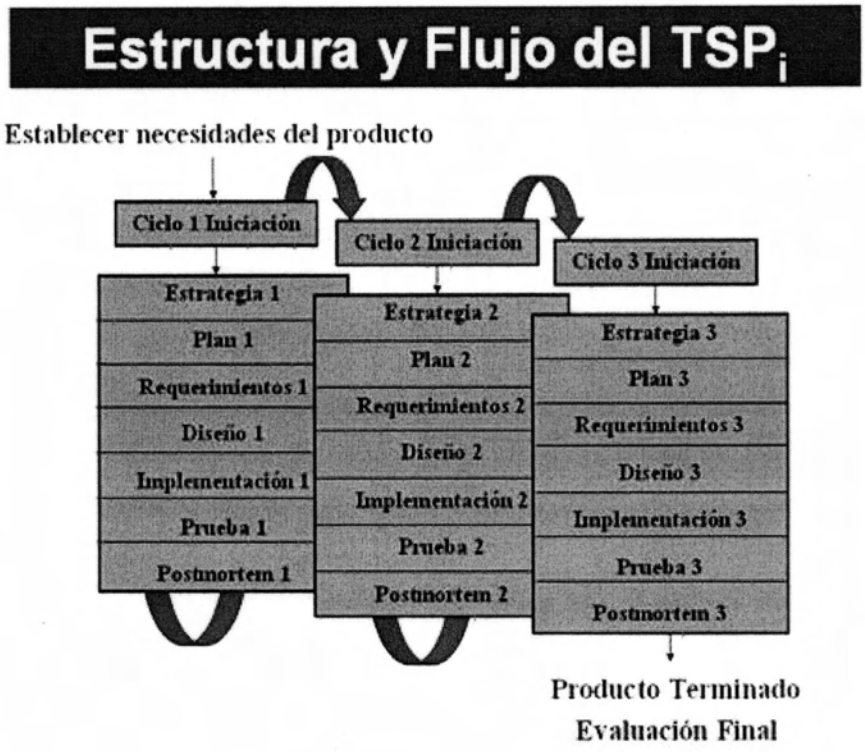


Figura 3. Estructura y flujo del TSPi [2]

1.3.5 Fases del ciclo de vida de TSP

El ciclo de vida de TSP consta de ocho fases. Aquí se abordará en qué consiste cada una y cuáles son sus objetivos.

Fase de Lanzamiento.

En un lanzamiento, todos los miembros del equipo desarrollan una estrategia, realizan un proceso y planifican su proyecto. Después de completar el lanzamiento, el equipo sigue con su proceso definido para hacer el trabajo.

Los equipos de TSP son relanzados de manera periódica. Debido a que el proceso de TSP sigue una estrategia de desarrollo iterativa, los relanzamientos periódicos son necesarios, ya que cada etapa o ciclo puede ser planeado con el conocimiento obtenido del ciclo anterior. En el lanzamiento de TSP, los equipos realizan un plan general y un plan detallado para los próximos 3 ó 4 meses. Una vez que los miembros del equipo han sido entrenados y el equipo ha sido formado, el equipo entero participa del lanzamiento del equipo TSP. El proceso de lanzamiento se grafica de la siguiente forma:

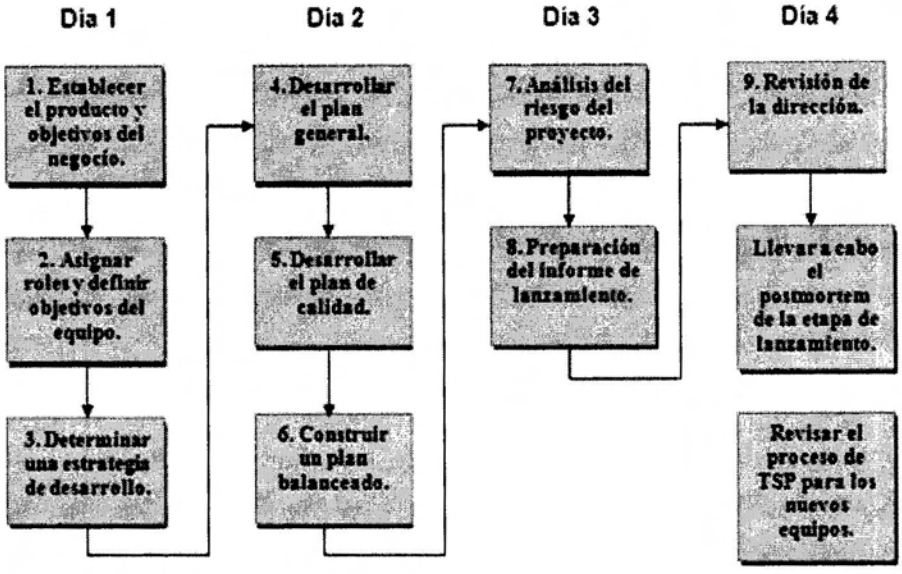


Figura 4. Proceso de Lanzamiento [2]

Paso	Actividad	Descripción
1	Establecer producto y objetivos del negocio.	-Revisar el proceso de lanzamiento e incorporar los miembros del equipo. -Tratar los objetivos del proyecto con la dirección y responder preguntas.
2	Asignar roles y definir objetivos del equipo.	-Seleccionar los roles del equipo. -Definir y documentar los objetivos del equipo
3	Determinar una estrategia de desarrollo.	-Producir un diseño conceptual del sistema. -Determinar la estrategia de desarrollo y los productos a realizar. -Definir el proceso de desarrollo a utilizar. Producir el proceso y soportar los planes.
4	Desarrollar el plan general.	Desarrollar las estimaciones del tamaño y el plan general.
5	Desarrollar el plan de calidad.	Desarrollar el plan de calidad.
6	Construir un plan balanceado.	-Asignación de trabajo a los miembros del equipo. -Planear las próximas etapas para cada miembro

		del equipo. -Armar un plan balanceado para el equipo y para cada miembro del equipo.
7	Análisis del riesgo del proyecto.	-Identificar y evaluar los riesgos del proyecto. -Definir las responsabilidades y puntos de control de la evaluación del riesgo.
8	Preparación del informe de Lanzamiento.	-Preparar un informe de lanzamiento para la dirección
9	Revisión de la dirección	Revisar las actividades de lanzamiento y los Planeamientos del proyecto con la dirección. Discutir los riesgos del proyecto, responsabilidades y acciones planeadas.

Tabla 2: Pasos y Actividades del TSP.

Una vez que el equipo de TSP es lanzado, se necesita asegurar que todos los miembros del equipo sigan el plan. Para ello, se deben tener en cuenta los siguientes tópicos:

- Liderar el equipo.
- Establecer una disciplina.
- Comunicación.
- Informar a la dirección.
- Mantener el plan.
- Estimar la terminación del proyecto.
- Re-balancear la carga de trabajo del equipo.

- Re-lanzar el proyecto.
- Manejo de la calidad en TSP.

Fase de Estrategias.

El objetivo es crear una estrategia para realizar el trabajo. Crear una gráfica de dependencias del producto. Hacer estimaciones preliminares sobre el tamaño del producto y el tiempo de desarrollo. Administrar los riesgos y el reuso. Finalmente, documentar la estrategia.

La estrategia del desarrollo se produce antes de iniciar el proyecto porque las actividades de estrategia y planeación están completamente relacionadas. El desarrollar un proceso comienza con un primer ciclo en el cual se diseña, implementa y se prueba el producto con una versión base produciendo un producto de calidad, con las funciones requeridas en el tiempo estimado.

Fase de Planeación.

El objetivo es realizar el plan de desarrollo del proyecto y el plan de calidad. Aplicar el criterio de estándares de calidad para generar un producto de alta calidad. Un plan está balanceado cuando todos los ingenieros completan sus tareas planeadas en el tiempo acordado. Ayuda a dirigir y a administrar el trabajo, además se analizan los requerimientos, se revisan e inspeccionan los productos, y se documenta la información. También se puede desarrollar un plan de prueba, ejecutar las pruebas y reportar cada trabajo.

Fase de requerimientos.

El objetivo es describir el proceso de requerimientos en TSPi. Explicar qué son, porqué son necesarios y discutir las características más importantes. En esta fase el equipo genera la Especificación de Requerimientos de Softwaree (ERS). En tal documento se hace una descripción clara de lo que será el producto; deberá incluir el criterio preciso para evaluarlo cuando esté terminado y asegurar que las funcionalidades sean las correctas. También proporciona retroalimentación al cliente acerca de lo que se pretende construir.

Fase de diseño.

El objetivo es proporcionar un diseño completo y de alta calidad que se utilice como base para la fase de implementación. El diseño es el principio creativo mediante el cual se decide cómo construir un producto, se debe contener una especificación completa y precisa de la construcción del producto. En un diseño completo se definen las partes principales de un producto, se describe como esas partes interactúan y se especifica cómo unirlas para producir el producto final. Cuando el diseño de alto nivel es completo y preciso, los ingenieros rápidamente pueden producir los diseños detallados de los

componentes; para eso, necesitan conocer las especificaciones funcionales completas de cada componente, sus interfaces y comportamientos de estado.

Fase de implementación.

El objetivo es describir el proceso de implementación a través del criterio de diseño, estándares de implementación, estrategias de implementación, revisiones e inspecciones.

Fase de pruebas.

El objetivo es integrar las pruebas y su documentación. Indicar los objetivos, las estrategias y planeación de las pruebas.

Fase de Postmortem.

En el Postmortem se revisa el trabajo del equipo para asegurar que se han completado todas las tareas necesitadas y todos los datos requeridos y registrados. También se re-examina lo hecho en el ciclo, qué fue lo correcto y ver como hacer el trabajo mejor para el siguiente ciclo. El postmortem nos da un camino ordenado para identificar las áreas que necesitan ser mejoradas y crear los cambios necesarios.

1.3.6 Métodos de estimación del Software

En principio, las estimaciones son hechas comparando el trabajo planeado con trabajos desarrollados anteriormente. Descomponiendo los productos en piezas más pequeñas y comparando cada parte con datos sobre las partes similares de productos anteriores, se puede determinar el tamaño del nuevo producto. Esta estrategia funciona bien para estimar casi cualquier tipo de trabajo de desarrollo. Sin embargo, requiere datos sobre los productos que se han desarrollado antes y el trabajo requerido para desarrollarlos. También se necesita un método que utilice datos históricos para hacer las estimaciones.

El Método de estimación basado en Proxy (PROBE)

El método PROBE defiende el uso de medidas seleccionadas personalmente y modelos de regresión basadas en datos personales para estimar el tamaño de un producto software. Un proxy es una medida de tamaño que puede ser utilizada para estimar la longitud de un proyecto medida en líneas de código. Por ejemplo PROBE utiliza una cuenta de objetos como un proxy de medida de tamaño. La estimación de líneas de código se utiliza para predecir el esfuerzo individual para desarrollar el esfuerzo. Los límites superior e inferior del intervalo de predicción deben ser también estimados. Este método ha sido automatizado en la herramienta Team Process Dashboard.

1.4 Descripción General del dominio del problema.

En la investigación fue necesario profundizar sobre el funcionamiento de los proyectos del polo productivo PetroSoft en la facultad, con el objetivo de conocer el estado actual y los problemas existentes en los mismos. El polo productivo PetroSoft está dividido en tres proyectos fundamentales, estos son el de Conceptualización de Soluciones, el Ministerio de Energía y Petróleo (MEMPET) y el Sistema Automatizado para el Control de la Gestión de Indicadores de las Refinerías (SACGIR). En este proyecto no se aplica correctamente el sistema de planificación, para la gestión del software y existen problemas en la estimación de riesgos. Se trabaja sobre estimaciones ficticias.

Es preocupante el descontento general del equipo de desarrollo, problema fehaciente que se evidencia también cuando se realizan los cortes de proyecto verificando gran parte de incumplimiento de tareas específicas, causando de esta manera principalmente el atraso general del proyecto. Tomando como referencia el proyecto Sistema Automatizado para el Control de la Gestión de Indicadores de las Refinerías (SACGIR) del polo productivo PetroSoft, se demuestra la necesidad de un proceso de control del trabajo individual y en equipo para aplicar entre los desarrolladores del mismo, así como de una selección más acertada de las herramientas y metodologías a utilizar para el desarrollo y puesta en práctica del software. Teniendo en cuenta los resultados de este proyecto se reflejan de forma resumida, cerca de dos meses de atraso. Existe un descontento generalizado entre los desarrolladores y jefes del proyecto a causa de la labor intensa poco orientada a necesidades y normas reales tanto del proyecto como de los mismos miembros del equipo de desarrollo, y en general una labor muy pobre de los sistemas de monitoreo, control, auditores, probadores y referente al personal en general vinculado con la calidad de este proceso productivo.

El proceso de gestión de los recursos, describe mediante organigramas la distribución tanto de las máquinas, como los roles de los miembros del proyecto, almacenando los datos relacionados con el personal, en un documento llamado SACGIR Capital Humano y recursos materiales almacena los datos relacionados con las máquinas asignadas a los polos, el nombre del polo al que se hace referencia, la cantidad de máquinas según el tipo de memoria, se almacena los datos relacionados con los recursos asignados al proyecto, la cantidad de máquinas portátiles y la cantidad de dispositivos externos. El líder del proyecto cuenta con los conocimientos necesarios para llevar a cabo la realización del proyecto. La visión del proyecto es discutida con todos lo miembros del equipo y es registrada a través de informes de reuniones del consejo de guerra en el cual es analizada la situación del proyecto, dándole cumplimiento al chequeo de acuerdos asignando responsables en cada unos de

los puntos a debatir y soluciones propuestas, aunque existe un atraso, ya que el último informe registrado de reunión del Consejo de Guerra fue el día 7 de enero de 2008.

Los miembros del equipo reciben una adecuada superación a través de cursos de superación y cursos optativos tales como Ingeniería de Requisitos, Modelado de Negocio y Cake PHP. Existen problemas en la planificación, esta no se realiza del todo y existe mucho atraso en cuanto a la actualización de documentos. Se planifica por fases y por iteraciones del ciclo de vida del proyecto, aunque no todas las fases o procesos del ciclo de vida están reflejados en la planificación, como es el caso del proceso de pruebas del software que no están registradas en los cronogramas, no se establecen puntos de chequeo que permitan verificar el estado del proceso.

Los objetivos son medibles y claros por fases e iteraciones. No existe un documento en el que se refleje el cumplimiento de las actividades actualizado, el último informe de actividades registrado es del día 24 de enero de 2008. No se realiza una planificación basada en los datos obtenidos del control de tiempo y esfuerzo de los miembros del equipo.

No se registra el tiempo de trabajo personal de los miembros del equipo, se hace una distribución del tiempo por miembros en cuanto a uso de las PC's, asignación de tiempo de máquina y se verifica su cumplimiento pero no se registra el tiempo de trabajo personal ni del equipo en ningún documento, por tanto no se registran los defectos encontrados, no se lleva un control en equipo de los compromisos documentados , pero si se realizan controles con el objetivo del cumplimiento de las tareas asignadas , se registran a través de documentos el control de la asistencia aunque existe atraso en cuanto a esto , el último informe sobre el control de asistencia registrado es del día 22 de noviembre de 2007 y se lleva a cabo un proceso de evaluación de los miembros del proyecto, y se procederá a realizar un análisis disciplinario con los estudiantes evaluados de Mal y de Bajo rendimiento, con sus respectivas sanciones. No existe por parte de los miembros del equipo conocimientos sobre los modelos de calidad, pero si esta registrado y bien definido el plan de aseguramiento de la calidad, nombrado SACGIR Plan de Aseguramiento de la Calidad v_1.0 fecha 28 de febrero de 2008.

1.5 Situación Problemática

La Universidad de las Ciencias Informáticas es una universidad concebida bajo un concepto de vinculación de la docencia con la producción de software. El objetivo de esta universidad es el de incrementar la economía del país mediante el desarrollo de la industria del software, cuyo destino está en las manos de estudiantes y profesores pertenecientes a la organización que se vinculan tempranamente a la producción.

La universidad está compuesta por diez facultades, las cuales participan en los proyectos que conforman los diferentes polos productivos de la misma, donde se busca que los productos que se desarrollen tengan una buena calidad. Para lograr que se cumpla este objetivo, el grupo de Calidad UCI está trabajando sobre la base del establecimiento de plantillas y métricas para controlar, de alguna forma, el proceso de desarrollo de software. Estas plantillas le han dado solución de manera temporal a los problemas de calidad que existen en la producción de software, pero desafortunadamente no le ofrecen al personal involucrado en cada uno de los proyectos, una guía que determine quién es el encargado de llenar los formularios que le corresponden según su rol además de que llenar algunas de estas plantillas resulta engorroso y de que necesitan otros formularios de gran importancia para hacer un buen seguimiento del proceso de desarrollo y aprovechar al máximo el tiempo y los recursos.

En el proyecto SACGIR, del Polo PetroSoft hay deficiencias de este tipo y lo que se busca es adaptar un modelo como TSP a las necesidades del proyecto para agilizar el trabajo y aprovechar al máximo el tiempo y los recursos en aras de obtener un producto de alta calidad.

Las entrevistas realizadas a los jefes de proyecto demuestran que no se utiliza TSP en este polo Acerca de este proceso se conoce muy poco dentro del personal involucrado en el proyecto.

Debido a todos los problemas se hace necesario determinar una estrategia del TSP para mejorar la calidad de los productos en estos proyectos para disciplinar y registrar el trabajo en equipo y para obtener datos históricos en qué basarnos a la hora de definir estas actividades.

1.6 Conclusiones parciales

A medida que las compañías utilizan TSP, los equipos van incrementando sus conocimientos y adquieren mayor experiencia en su uso, estos van mejorando en estimaciones, disminuyendo la densidad de los defectos, incrementando la productividad y por consiguiente mejorando la calidad del producto final. Actualmente la información que demuestre concretamente lo inefectivo de TSP es completamente nula.

La necesidad de procedimientos, estrategias y guías de procesos de software en equipo que rijan el desarrollo de productos de software ha conducido a la elaboración del Capítulo que acaba de ser presentado.

CAPÍTULO 2

Propuesta del Proceso de Software en Equipo

2.1 Introducción.

A lo largo de la investigación realizada, en el Polo PetroSoft de la Universidad de las Ciencias Informáticas, para poder desarrollar este trabajo de tesis, se pudo observar que el principal problema que afecta la obtención de productos con la calidad requerida, en el tiempo determinado, con los costos acordados y la completa satisfacción del cliente para el cual se trabaja, se debe principalmente a la no utilización del Proceso de Software en Equipo. Después de un análisis detallado del TSP, se realizará una estrategia como propuesta de solución para eliminar los problemas mencionados anteriormente.

En el presente capítulo se efectúa la propuesta de solución para aplicar Proceso de Software en equipo al Polo PetroSoft, se explica detalladamente la estrategia a aplicar, la cual está dividida por fases y cada fase contiene una serie de pasos y formularios que se deben cumplir para lograr que los productos finales cumplan con las expectativas de los clientes.

2.2 Etapas del ciclo de vida del TSP

2.2.1 Etapa de Lanzamiento:

El objetivo es formar y construir el equipo, determinar los roles del equipo a través del formulario **INFO** (Ver Anexo 3), establecer sus objetivos, ayudar a decidir sobre la práctica de éstos y cómo establecer los objetivos del equipo. Los objetivos deben ser medibles, dado que establecen la estructura para la estrategia y el plan.

Se deben definir los objetivos del proyecto, especificar cómo medirlos, proporcionarle una copia de los objetivos a cada integrante del equipo y al administrador de configuración para que los anexe en la carpeta del proyecto.

Para establecer los objetivos del proyecto en primer lugar se debe considerar que el producto cumpla con las expectativas del cliente. Por ello TSPi considera los siguientes objetivos básicos:

Objetivos del proyecto:

Objetivo 1: Generar un producto de calidad. TSPi propone los siguientes objetivos para desarrollar un producto de calidad.

- Porcentaje de defectos encontrados en la primera compilación: 80%.
- Número de defectos encontrados en las pruebas del sistema: 0.
- Requerimientos de funciones incluidas en el proyecto: 100%.

Objetivo 2: Realizar un proyecto bien administrado y que sea productivo.

- Error en el tamaño del producto estimado menor al 20%.
- Error en las horas de desarrollo estimadas menor al 20%.
- Porcentaje de datos registrados e incluidos en la carpeta del proyecto: 100%.

Objetivo 3: Finalizar a tiempo.

- Días antes o después en completar el ciclo de desarrollo a lo más 4.

Objetivo común.

El objetivo común consiste en ser un integrante de equipo eficaz y cooperativo. Todos tienen el objetivo principal de trabajar en cooperación con el equipo entero para generar un producto de alta calidad de acuerdo al calendario establecido para el proyecto. El éxito del equipo depende de que todos sus integrantes contribuyan con su mejor esfuerzo personal, apoyando a los demás y trabajando conjuntamente para resolver problemas y desacuerdos.

Objetivos de los miembros del equipo.

Se espera que cada integrante del equipo conozca los objetivos del proyecto y que además cumpla los siguientes objetivos como integrante y sepa cómo medirlos.

Objetivo 1: Ser un miembro del equipo efectivo y cooperativo.

Objetivo 2: Hacer un trabajo disciplinado.

- Porcentaje de datos personales registrados en la carpeta del proyecto: 100%.
- Porcentaje de las semanas en las cuales se realizaron y finalizaron las actividades registradas en cada forma personal SEMANA: 100%.

Objetivo 3: Planear y mantener un seguimiento de trabajo personal.

- Porcentaje de datos del proyecto registrados en los formatos REGT: 100%.

Objetivo 4: Generar productos de calidad.

- Promedio de porcentaje de defectos encontrados antes de la primera compilación mayor al 70%.
- Densidad de defectos encontrados durante la compilación: 10/KLOC.
- Densidad de defectos encontrados durante las pruebas unitarias: 5/KLOC.
- Densidad de defectos encontrados después de la prueba unitaria: 0.

Objetivos del rol.

Líder del equipo: El líder del equipo es el responsable de que el equipo funcione adecuadamente. Los objetivos específicos del líder del equipo son:

- Construir y mantener un equipo eficaz.
- Motivar a todos los miembros del equipo a trabajar tenazmente en el proyecto.
- Resolver la mayoría de los problemas del equipo que se susciten.
- Tener a todos los miembros del equipo y al instructor informados sobre los avances.
- Actuar efectivamente como moderador en las reuniones.

Administrador de desarrollo: El administrador de desarrollo está principalmente involucrado con la producción y funcionalidad de un producto de alta calidad de desarrollo. Los objetivos principales del administrador de desarrollo son las siguientes:

- Generar un producto de alta calidad.
- Identificar plenamente las destrezas y habilidades de cada miembro del equipo.

Administrador de planeación: Los objetivos principales del administrador de planeación son guiar al equipo en la producción de un plan detallado y guiar el progreso del producto de acuerdo al plan. Los objetivos específicos del administrador de planeación son:

- Producir un plan completo, preciso y detallado para cada miembro del equipo y para todo el equipo.
- Proporcionar los reportes del estado semanal del equipo.

Administrador de calidad y proceso: El objetivo principal del administrador de calidad y proceso es asegurar que el equipo esté siguiendo TSPi. Los objetivos específicos del administrador de calidad y proceso son:

- Asegurar que todos los miembros del equipo reporten y usen apropiadamente los datos del proceso de TSPi.
- Asegurar que el equipo siga fielmente TSPi y producir un producto de calidad.
- Asegurar que todas las metas del equipo sean bien realizadas y reportadas.

- Asegurar que todas las metas del equipo sean reportadas con exactitud y que los reportes sean colocados en la carpeta del proyecto.

Administrador de configuración: El principal objetivo del administrador de configuración es garantizar que el proyecto está propiamente soportado y controlado. Los objetivos específicos del administrador de configuración son:

- Asegurar que el equipo tiene disponibles las herramientas y métodos para que apoyen su trabajo.
- No realizar cambios sin autorización en la línea base.
- Todos los riegos y asuntos del equipo son registrados en el sistema de seguimiento.
- Asegurar que el equipo conoce los objetivos de rehúso para el ciclo de desarrollo.

Responsabilidad	LE	AD	AP	ACP	AC
Construye y mantiene un equipo efectivo.	X				
Resuelve asuntos entre los integrantes del equipo	X				
Realiza seguimiento y reporte del progreso del equipo	X				
Dirige cada reunión	X				
Se reúne con el instructor	X				
Realiza el mantenimiento a la carpeta del proyecto	X				
Ayuda a asignar las tareas en el equipo	X				
Dirige todo el trabajo de desarrollo.		X			
Dirige la planeación del equipo y el seguimiento del progreso.			X		
Dirige el seguimiento y planeación de calidad				X	
Proporciona apoyo al proceso del equipo.				X	
Dirige cada inspección.				X	
Mantiene los estándares y glosarios.				X	
Realiza el reporte de cada reunión.				X	
Alerta al equipo en los problemas de calidad.				X	
Obtiene las herramientas y apoyo necesario.					X
Maneja la administración de configuración.					X
Dirige la mesa de control de configuración.					X
Dirige el rehúso del equipo.					X
Realiza el seguimiento de asuntos y riesgos.					X
Mantiene el glosario del sistema					X
Desarrolla el producto.	X	X	X	X	X
Realiza planes personales	X	X	X	X	X
Realiza seguimiento en el trabajo personal	X	X	X	X	X
Genera productos de calidad	X	X	X	X	X
Sigue prácticas profesionales disciplinadas.	X	X	X	X	X

Tabla 3: Roles y sus responsabilidades [2]

Donde:

- Líder del equipo (LE).
- Administrador de Desarrollo (AD).
- Administrados de Planeación (AP).
- Administrador de Calidad y Proceso (ACP).
- Administrador de Configuración (AC).

Objetivos del Producto.

En esta parte se describe la construcción del producto. Si existe algún tipo de antecedente, se explican las características en las que se encuentra y dónde va a iniciar el producto.

Estándar para entrega de documentación.

El estándar de documentación permite al equipo mantener una misma línea en las características de la información, de manera tal que no existan copias de una misma versión ni confusiones entre los documentos en cada fase del proyecto.

El estándar de documentación contiene lo siguiente:

- Uso de procesador de textos y sus versiones.
- Tipo de letra y tamaño, ya sea tanto para títulos principales como para subtítulos.
- Tabuladores, margen, tamaño de pie de página.
- Tipo de alineación.
- Color de tinta y tamaño de la hoja para la impresión.
- Espacio de interlineado y tipo de viñeta.
- Formato de fecha.
- Características del contenido en la cabecera de cada documento.
- Identificación de cada documento en el pie del documento.

Carpeta del Proyecto.

TSPi propone a los equipos mantener una carpeta para el proyecto. El objetivo es crear un estándar para salvaguardar todos los puntos importantes de información del proyecto. El líder del equipo es responsable para establecer y mantener la libreta del proyecto, y todos los miembros del equipo son responsables para proveer los materiales necesarios para la carpeta. A continuación se describen las diferentes secciones que posee la misma:

- Resumen.
- Reportes de los Ciclos y del Proyecto.
- Tareas Planificadas y Tareas Actuales.
- Documentos de Procesos.
- Datos de los Componentes.
- Datos y Planes de Pruebas.
- Reportes de Inspecciones.
- Documentos de Trabajo.

Reuniones del equipo y la primera reunión.

Después de que el equipo se ha establecido y los roles han sido asignados, es tiempo de hacer la primera reunión. Estas reuniones son hechas para dar al equipo la oportunidad de discutir acuerdos y las metas en el ciclo. Esta primera reunión también ayuda a entender el propósito de las reuniones semanales.

El equipo debe establecer un tiempo estándar, un lugar para las reuniones y reunir al equipo entero cada semana. Otra función de las reuniones semanales es analizar los datos del equipo de la semana previa. Revisando los datos individuales, los miembros del equipo pueden ver en que parte se encuentran trabajando de acuerdo al plan, dónde necesitan balancear y ajustar sus planes personales.

Los datos necesarios para efectuar estas reuniones se almacenan en los formularios **SEMANA** (Ver Anexo 4), **TAREAS** (Ver Anexo 5) y **HORARIO** (Ver anexo 6) que describe TSP.

2.2.2 Fase de Estrategia.

El diseño conceptual.

Es recomendable hacer este diseño, ya que éste constituye un plan para desarrollar el producto posteriormente. Aquí se selecciona la mejor alternativa para construir el producto incluyendo todas las funciones que se quieren desarrollar en los tres ciclos del proyecto y cuáles desarrollar en cada uno de ellos. Además, se deben tener definidos los componentes a utilizar para crear el producto final o por lo menos saber lo que hace cada uno. Es recomendable tener solamente el concepto del diseño y una estructura general del producto y obviar cualquier diseño real mientras se están analizando e inspeccionando los requerimientos, ya que los ingenieros están tan enfrascados en hacer un buen plan, que emplean tiempo de más en esta tarea sin necesidad.

Una vez creado el diseño conceptual se deben estimar el tamaño y el tiempo requerido para realizar el producto utilizando el método matemático de estimación PROBE, el cual se incluye en una funcionalidad de la herramienta Process Team Dashboard.
Para este fin se propone el uso del formulario **ESTRAT** (Ver Anexo 7).

Administración de Riesgos.

Un riesgo se detecta como un suceso con probabilidades de ocurrir y que afecta negativamente el proyecto. Para evitarlos o controlarlos, es conveniente identificarlos desde el principio y determinar las precauciones adecuadas para ello, el equipo deberá evaluar los riesgos que sean más significantes del proyecto, tomando una decisión cual de éstos son los más probables. De esta manera se clasifican

tomando la escala de B (Baja), M (Mediana), A (Alta), dependiendo el impacto sobre el proyecto y clasificarlos. Se propone el uso del formulario **ITL** (Ver Anexo 8)

Una Estrategia de Reuso.

Desarrollar una gran cantidad de código genera riesgos ya que las funciones podrían requerir sustancialmente más código. Uno de los caminos a reducir los riesgos es adoptar una estrategia de reuso. Se puede desarrollar código que se sabe que podría utilizarse más adelante. Esta estrategia puede reducir el esfuerzo, antes de desarrollar el producto se puede especificar y diseñar las partes que se pueden reusar, crear una biblioteca de reuso e incorporarla cuando el producto se esté desarrollando. En TSPi el administrador de configuración es el encargado de proporcionar las partes de reciclaje que se van a guardar en la biblioteca.

Para esta fase se propone además el formulario **LOGT** (Ver Anexo 9), que utiliza PSP, para registrar el tiempo que se empleó en las estimaciones del tamaño y del tiempo de desarrollo del producto respectivamente y el tiempo empleado en la administración de riesgos.

Producir un plan de Configuración.

Cuando el equipo no usa un plan adecuado de configuración, no conocen qué módulos han sido aprobados, aumentados, o probados. Se construyen productos con errores o no se conocen las pruebas que han sido corridas o cuáles defectos han sido inyectados y por lo tanto esto es tiempo perdido, por lo que se necesitará rehacer el trabajo que en realidad ya se había completado. Este tipo de problemas son realmente inquietantes porque cuando pasa, usualmente se está a punto de terminar el modulo o el ciclo, y se tiene muy poco tiempo para recobrarlo.

El proceso de administrar la configuración tiene los siguientes pasos [2]

- Copiar la versión de cada elemento del producto.
- Registrar todos los cambios hechos por cada línea base.
- Registrar quién hace los cambios.
- Registrar cuando se hicieron los cambios.
- Registrar qué cambios fueron hechos.
- Registrar porqué se hicieron los cambios.

Por lo tanto, los miembros del equipo buscan la versión final de cualquier elemento del programa. También se conocerán los cambios hechos, cuándo y por qué. El administrador de configuración produce el plan de configuración y lo revisa con el resto del equipo. Para este plan de configuración se recomienda utilizar el formulario **CCR** (Ver Anexo 10)

2.2.3 Fase de Planeación.

El objetivo es realizar el plan de desarrollo del proyecto y el plan de calidad para este fin se proponen los formularios **SUMQ** (Ver Anexo 11) y **SUMP** (Ver anexo 12). Aplicar el criterio de estándares de calidad para generar un producto de alta calidad.

Plan Balanceado.

Con este plan se persigue que ningún integrante del equipo trabaje más que otro y que ninguno tenga que esperar por otro para realizar su trabajo. Cuando un plan no está balanceado, el cumplimiento de las tareas o compromisos de algún ingeniero puede demorar al equipo completo dependiendo de la cantidad de trabajo que tenga el mismo, por lo tanto retrasaría el trabajo del resto y esto ocasionaría una pérdida de tiempo.

Un plan está balanceado cuando todos los ingenieros completan sus tareas planeadas en el tiempo acordado. Ayuda a dirigir y administrar el trabajo, además se analizan los requerimientos, se revisan e inspeccionan los productos, y se documenta la información. También se puede desarrollar un plan de prueba, ejecutar las pruebas y reportar cada trabajo.

Plan de Inspecciones.

Para llevar a cabo un desarrollo con calidad se debe hacer una planeación que permita obtener tiempo suficiente para revisiones e inspecciones. Se recomienda utilizar las siguientes líneas guías:

- Revisiones e inspecciones en requerimientos a lo más 2.0 páginas por hora cuyo texto tendrá interlineado sencillo.
- Revisiones e inspecciones en diseño de alto nivel a lo más 5 páginas por hora de diseño.
- Revisiones e inspecciones en diseño detallado a lo más 100 líneas por hora en pseudocódigo.
- Revisiones e inspecciones de código fuente a lo más 200 LOC/hora.

Al momento de hacer revisiones no se asegura que se encontrarán defectos, por eso es necesario utilizar métodos de revisión efectivos. Para revisiones e inspecciones de código se recomienda utilizar una lista de verificación personal y actualizarla frecuentemente a partir de los defectos de compilación y pruebas encontrados en los programas que se han revisado e inspeccionado. Para realizar

revisiones de diseño efectivas es necesario que el diseño esté bien documentado y que se analice la lógica del programa. En este caso se propone el Formulario **INS** (Ver Anexo 13) que recoge los datos referentes a las inspecciones realizadas.

El formulario **LOGT** (Ver Anexo 9) que propone PSP en la primera y la segunda etapa, es el que se debe llenar en esta fase ya que aquí se debe registrar el tiempo que se tomó en la planificación contando el tiempo de inspecciones y de corrección de defectos.

2.2.4 Fase de Requerimientos.

El objetivo es describir el proceso de requerimientos en TSPi. Explicar qué son los requerimientos, porqué son necesarios y discutir las características más importantes.

En esta fase el equipo genera la Especificación de Requerimientos de Software (ERS). En tal documento se hace una descripción clara de lo que será el producto; deberá incluir el criterio preciso para evaluarlo cuando esté terminado y asegurar que las funcionalidades sean las correctas. También proporciona retroalimentación al cliente acerca de lo que se pretende construir.

Se debe saber exactamente lo que el producto debe hacer antes de construirlo. Durante el desarrollo de los requerimientos se revisan las necesidades del cliente y se formulan preguntas acerca de las funciones del producto, si existieran necesidades que no son claras. A través de este proceso se logra un común acuerdo acerca de lo que se va construir; por ésta razón es que cada integrante debe participar en la definición de los requerimientos y se define un proceso bien estructurado para construirlos. Los requerimientos se escriben en lenguaje coloquial y se revisan con los usuarios para verificar si realmente es lo que desean. El administrador de desarrollo dirige al equipo para formar el formato ERS e identificar todas las tareas que se deben realizar. Después de dividirlas en secciones el líder del equipo asigna a los integrantes y a establecer acuerdos en fechas de términos de cada una.

Extracción de los requerimientos.

Para obtener los requerimientos se interroga a los clientes, usuarios y otros involucrados, para determinar lo que se necesita realmente.

Los pasos importantes para la obtención de los requerimientos son los siguientes:

- Evaluar la factibilidad del sistema.
- Conocer y comprender los asuntos organizacionales.
- Identificar a todos los involucrados en el proyecto.
- Registrar las fuentes de los requerimientos.
- Definir el ambiente operativo del sistema.

- Evaluar asuntos del negocio.
- Definir las restricciones del dominio.
- Registrar la razón de los requerimientos.
- Realizar un primer prototipo de los requerimientos.
- Definir los casos de uso.
- Definir los procesos operacionales.

Especificación de los requerimientos (ERS).

En este documento el equipo describe las funciones que se planean desarrollar, cómo se pretende generar el producto, puesto que es una responsabilidad de la fase del diseño.

En TSPi el equipo se concentrará en los requerimientos operativos, mencionando a continuación los principales:

- Requerimientos funcionales: entradas, salidas, procesos y casos de uso.
- Requerimientos de interfaz externos: usuario, hardware, software y comunicaciones.
- Restricciones del diseño: formatos de archivo, lenguajes, estándares, compatibilidad.
- Atributos: disponibilidad, seguridad, mantenimiento, conversión.
- Otros requerimientos: Base de datos, instalación.

1 Tabla de contenido.
2 Introducción.
2.1 Propósito del documento Especificación de Requerimientos de Software.
2.2 Definición del problema.
2.3 Información del equipo.
3 Requerimientos funcionales.
3.1 Definición de los requerimientos funcionales del sistema
3.2 Requerimientos del ciclo 1.
3.3 Requerimientos del ciclo 2.
3.4 Estructura top-down (estructura de lo general a lo particular)
4 Definición de reglas utilizadas en los requerimientos.
5 Requerimientos de interfaz externa.
5.1 Interfaz del usuario.
5.2 Formatos en pantalla.
6 Restricciones en diseño/implementación.
6.1 Acuerdos en estándares.
6.2 Restricciones en desarrollo.
7 Requerimientos especiales del sistema
7.1 Documentación.
7.2 Compatibilidad.
8 Referencias y Fuentes de información.

Tabla 4: Puntos clave del ERS [2]

Para asegurar que el documento ERS este completo se debe, enumerar los párrafos y secciones para identificar cada requerimiento.

En la documentación del ERS se realiza una definición breve y clara de lo que se pretende construir. Para los casos de uso, se listan los pasos en un guión simple, se usan tablas y listas con viñetas, tratar de evitar párrafos con demasiado texto. Una forma de asegurar el acuerdo entre el equipo es hacer el caso de uso de cada función, los cuales pueden servir como escenarios de pruebas del sistema para verificar las funciones del producto. Cada tarea terminada se debe entregar al administrador de desarrollo quien las debe unir en un borrador de ERS y dar copias a los demás ingenieros para realizar la revisión.

Plan de Pruebas del Sistema.

A partir de la fase de requerimientos se empiezan a planear las pruebas del sistema, para asegurar que contenga los acuerdos pactados entre los usuarios y el equipo de trabajo. Probar bajo condiciones de error considerando la usabilidad y los resultados de recuperación. Todo esto se debe mencionar en el plan y explicar cuáles se evaluarán, cuáles no y las razones para eso.

Inspección de los requerimientos y del Plan de Pruebas del Sistema.

Se debe realizar una inspección en equipo del borrador del ERS y del Plan de Pruebas del Sistema. Se recomienda tener tiempo suficiente para tal inspección (una hora por página con espaciado sencillo). El objetivo de la inspección es poder encontrar problemas e inconsistencias antes de iniciar la labor de diseño. Todos los datos obtenidos en las inspecciones se deben guardar en el formulario **INS** (Ver Anexo 13).

El formulario **LOGT** (Ver Anexo 9) se propone en esta etapa para registrar el tiempo que tomó especificar los requerimientos, elaborar el plan de pruebas de sistema, el tiempo que tomó realizar las inspecciones a cada uno de estos documentos y el tiempo de corrección de los mismos. También se propone el formulario **LOGD** (Ver Anexo 14) para registrar los defectos encontrados durante las inspecciones.

Actualizaciones de requerimientos.

Después de que aplicó la inspección en los requerimientos y se obtuvo la aprobación, se deben actualizar los documentos ERS y Plan de Pruebas del Sistema de acuerdo con los problemas encontrados en la inspección. Las secciones corregidas se proporcionarán al administrador de desarrollo, quien las reúne para obtener el documento final del ERS y distribuir las copias al resto del equipo.

Revisión del ERS por el usuario.

Después de producir, inspeccionar, y corregir el ERS y el plan de pruebas del sistema, los usuarios finales deben leer el ERS y estar de acuerdo en lo que se describe en ellos. Esto es una base para generar un producto de calidad y para prevenir los cambios después del ERS aprobado.

Línea base de los requerimientos.

El administrador de configuración documenta la línea base el documento ERS, el equipo puede solicitar cambios solamente utilizando el procedimiento de control de cambios mediante el formato de solicitud de cambios **CCR** (Ver Anexo 10).

2.2.5 Fase de Diseño.

El objetivo es proporcionar un diseño completo y de alta calidad que se utilice como base para la fase de implementación.

Diseño es el principio creativo mediante el cual se decide cómo construir un producto, se debe contener una especificación completa y precisa de la construcción del producto. En un diseño completo se definen las partes principales de un producto, se describe cómo esas partes interactúan y se especifica cómo unirlas para producir el producto final. Cuando el diseño de alto nivel es completo y preciso, los ingenieros rápidamente pueden producir los diseños detallados de los componentes; para eso, necesitan conocer las especificaciones funcionales completas de cada componente, sus interfaces y comportamientos de estado.

Cuando se diseña un producto en equipo, las principales preguntas conciernen cómo producir el diseño y el orden en el cual se van a diseñar las diferentes partes de los productos. Un problema común en el diseño de sistemas grandes de software es la necesidad de definir la estructura completa del sistema antes de especificar cualquier otra cosa. Mientras los diseñadores del sistema producen las especificaciones de los componentes externos, se pueden identificar las alternativas posibles para diseñar.

Estándares de diseño.

Existen varios estándares del diseño, de los cuales se consideraron los más importantes:

Convenciones de nombrado:

Es el primer estándar en el cual se especifica la estructura del nombramiento. El administrador de configuración debe establecer el glosario del sistema. Se deben definir los nombres de los tipos jerárquicos de programas (tales como sistema, producto, componente, módulo u objeto), las convenciones usadas en nombres de programas, archivos, variables y parámetros; y los procedimientos para establecer, controlar y cambiar nombres.

Formato de interfaz:

Aquí se definen los formatos y el contenido de las interfaces de los componentes. Esto consiste en indicar parámetros para variables, código de error u otras condiciones.

Mensajes del sistema y de error:

Es conveniente establecer los procedimientos para mensajes de error y del sistema. Un sistema útil debe tener consistencia y los mensajes deben ser comprendidos fácilmente.

Estándares de representación:

El estándar de representación del diseño define el producto que se generará a partir del diseño. Es importante definir y utilizar estos estándares porque una representación del diseño imprecisa o ambigua puede producir problemas graves en la implementación y en las pruebas. Por eso es necesario documentar cada diseño, usar especificación operacional, funcional, de estados, lógica y algunos otros tipos de representaciones, esto permite ahorrar tiempo y producir un diseño revisado, el cual pueda ser verificado antes de la implementación.

Estándares de documentación de reuso:

Los estándares de documentación marcan la diferencia entre las partes de rehúso y las que no lo son. Un buen rehúso permite que los ingenieros puedan utilizar un programa sin tener que analizar el código fuente. Por eso las partes de rehúso deben contener una especificación completa del comportamiento externo, también es buena idea añadir una sección de comentarios al principio del programa fuente en cada parte.

La estructura completa del producto:

Para definir la estructura completa del producto es necesario agrupar las funciones por componente con las referencias del párrafo ERS para cada función en el formulario **ESTRAT** (Ver Anexo 7). Posteriormente se anotan cuáles son las funciones que deberán ser desarrolladas en cada ciclo de desarrollo y se listan las líneas de código estimadas para cada una.

La especificación de diseño (EDS).

Mientras se produce el documento EDS, se definen las interfaces externas y las especificaciones funcionales de cada componente, se genera un conjunto completo de escenarios que cubren las funciones externas de cada uno. El paso final en el diseño de alto nivel es producir los diversos documentos de diseño especificando la lógica y la estructura del programa principal del sistema para asegurar que las funciones de los componentes y las interfaces están completamente especificadas. Al producir las EDS, cada integrante debe revisar su trabajo y arreglar los problemas, después de eso se debe entregar al administrador de desarrollo para que lo incorpore en el borrador completo del EDS y distribuya copias al equipo.

Calidad en la parte de reuso.

La alta calidad es una parte importante en la estrategia de rehusó. Para obtener una calidad de alto nivel se utiliza un proceso definido completamente y efectuar revisiones personales e inspecciones entre colegas sobre el diseño y el código.

Deberá considerar las siguientes preguntas al decidir reutilizar un programa:

- ¿El programa tiene funciones correctas?
- ¿La interfaz del programa es portable en la nueva aplicación?
- ¿Todos los materiales necesarios están disponibles? Por ejemplo: código fuente, casos de pruebas, datos de pruebas e instrucciones de la aplicación.
- ¿El programa es portable en cuanto a estándares? como el nivel del lenguaje; estándares de codificación; estándares en nombramiento y en archivos; y estándares en mensajes y ayuda.
- ¿Se puede demostrar que el programa es de alta calidad?

Si existe alguna respuesta negativa es recomendable dedicarle tiempo considerable al componente para rehúso.

Plan de Pruebas de integración (PPI).

Es importante generar éste documento en las especificaciones de diseño. Con las pruebas de integración se van a revisar y verificar todas las interfaces entre los componentes del sistema, para ello también se debe indicar la manera para hacer las pruebas. Para asegurar que a todas las interfaces se les han aplicado las pruebas es conveniente inspeccionar el Plan de pruebas de integración al inspeccionar la EDS.

Inspección del diseño.

El administrador de calidad y proceso dirige al equipo en la inspección del borrador de EDS y del PPI. El equipo completo debería participar en la inspección para encontrar el número máximo de defectos y asegurar que todos entienden el diseño.

Se deben utilizar el formulario **INS** (Ver Anexo 13) para guardar los datos pertinentes a las inspecciones; el formulario **LOGD** (Ver Anexo 14), que propone PSP, para registrar la información de los defectos encontrados y el formulario **LOGT** (Ver Anexo 9) para registrar el tiempo que tomó producir las EDS, realizar el plan de pruebas de integración y sus respectivas inspecciones.

Actualización del diseño.

Después de la inspección del diseño, corregir los errores identificados en la EDS y en el PPI, si fuese necesario haga que uno o dos de los ingenieros revisen las correcciones, después de eso proporcione la EDS y las secciones del plan de pruebas al administrador de desarrollo, quien produce y distribuye los documentos terminados.

Criterio de éxito.

Los productos del diseño son los siguientes:

- La Especificación de Diseño de Software terminada, inspeccionada y corregida, incluyendo todos los materiales de diseño necesarios.
- El Plan de Pruebas de Integración terminado, inspeccionado y corregido.
- La forma de registro de Defectos REGD.
- Los datos de los ingenieros en tiempo y defectos registrados en las formas debidas.
- Copia de todos los materiales de diseño en la carpeta del proyecto.

2.2.6 Fase de Implementación.

El objetivo es describir el proceso de implementación a través del criterio de diseño, estándares de implementación, estrategias de implementación, revisiones e inspecciones

Estándares de implementación.

Una vez que el equipo está de acuerdo con respecto de los estándares que se necesitarán, el equipo puede discutir sobre el contenido de las especificaciones estándares que usarán. Los estándares de implementación agregan elementos a los estándares definidos durante la fase de diseño y los extienden.

A continuación se tratan los siguientes temas relacionados con los estándares:

- Nombre, interfaces y mensajes estándares.
- Estándares de codificación.

Nombre, interfaz y mensajes estándares.

Los estándares de nombres, de interfaz y de mensajes desarrollados durante la fase de diseño se revisan para asegurar que son apropiados y están siendo utilizados.

La lista de rutinas de rehuso para verificar que estén terminadas y que todos los integrantes del equipo las estén utilizando.

El glosario de nombres para asegurarse de que cada módulo esté usando el mismo nombre para el mismo producto y que todos los nuevos nombres se van agregando al glosario.

Los nombres de los componentes, los nombres de las variables compartidas, de los parámetros y de los archivos se revisan para verificar su consistencia.

Las interfaces y los mensajes estándares para asegurarse de que hayan sido definidos, registrados en el glosario de nombres y que son conocidos y utilizados por los integrantes del equipo.

Estándares de Codificación.

Un estándar común de codificación asegura la consistencia en el código del equipo; esto facilita la realización de inspecciones, permite que se hagan de una manera más rápida y que sean efectivas. Un estándar de codificación bien estructurado define una sección para comentarios, lo cual permite mayor flexibilidad al momento de realizar incrementos en ciclos subsecuentes.

Diseño detallado y revisiones en el diseño.

Se debe producir un diseño detallado para cada módulo que planean implementar y hacer una revisión personal de ese diseño, prestando atención a los lazos y a toda la lógica compleja.

Desarrollo de Pruebas.

Después de que se arreglaron los errores que se encontraron en la revisión, se desarrolla el código para las pruebas unitarias. Generalmente, durante el desarrollo de las pruebas se encuentran más errores del diseño que en la inspección de diseño o en las pruebas unitarias, por eso es conveniente aplicar las pruebas antes de la inspección en el diseño detallado.

Inspección en el diseño detallado.

Después de la aplicación de pruebas, el administrador de calidad y proceso dirige la inspección del diseño detallado. Para este fin se proponen los formularios **INS** (Ver Anexo 12), para llenar los datos de la inspección, el **LOGD** (Ver Anexo 13), para registrar los principales defectos encontrados y el **LOGT** (Ver Anexo 9) para determinar el tiempo que tomó la inspección.

Revisiones de codificación y código.

Después de la inspección de diseño, se realiza la revisión de la codificación. También se revisa la consistencia en nombres, inicializaciones, puntuaciones, llamadas de secuencias y de inclusión, todos los errores que se encuentren en el código deben ser anotados en el formulario **LOGD** (Ver Anexo 13). Se debe llevar el control del tiempo utilizado en la búsqueda de defectos en el formulario **LOGT** (Ver Anexo 9). Se pide a otro ingeniero que revise el código antes de que se compile.

Inspección en Código.

Después de la compilación se compara el tiempo dedicado al diseño y su revisión, al código y su revisión con los tiempos del plan de calidad del equipo cuyos datos se encuentran en el formulario **SUMQ** (Ver Anexo 11). Se aconseja utilizar la siguiente tabla de Criterio Estándar de Calidad:

- El tiempo dedicado al diseño debería ser mayor que el tiempo de la codificación.
- El tiempo dedicado a la revisión del diseño debería ser mayor que el 50% del tiempo del diseño.
- El tiempo dedicado a las revisiones de código debería ser mayor que el 50% del tiempo de codificación y mayor que el 75%.
- La tasa de revisión fue menor de 200 LOC por hora.

Se deben registrar los datos obtenidos durante estas inspecciones en el formulario **INS** (Ver Anexo 12) y del tiempo que tomó dicha inspección la forma **LOGT** (Ver Anexo 9)

Después de la compilación se revisa la calidad de los componentes para determinar si se incluyen en el sistema, si se someten a más revisiones o si se reestructura. Posteriormente se actualiza el formulario **SUMQ** (Ver Anexo 11) para el sistema y sus componentes.

2.2.7 Fase de Pruebas

El objetivo es integrar las pruebas y su documentación. Indicar los objetivos, las estrategias y planeación de las pruebas.

Estrategia de Pruebas de TSPi.

La estrategia de pruebas tiene como objetivo verificar que los productos a los que se les aplican las pruebas son de alta calidad. Las actividades principales son las siguientes:

- Utilizar las partes desarrolladas a las que se les ha aplicado pruebas unitarias para construir el sistema.
- Utilizar las pruebas de integración para verificar que el sistema está debidamente construido, que todas las partes se encuentran desarrolladas y que funcionan en conjunto.
- Validar con las pruebas del sistema que el producto realiza los requerimientos del sistema.

De manera simultánea se puede:

- Identificar módulos o componentes de baja calidad y entregarlos al administrador de calidad y proceso para su evaluación y depuración.
- Identificar componentes de baja calidad que estén causando problemas, aún después de que se ha depurado; regresarlos al administrador de calidad y proceso para que trabaje de nuevo con ellos o los reemplace.

Se construyen los planes de las pruebas de integración y del sistema. Para la construcción se reúnen las diferentes partes del sistema dentro de un sistema completo de manera que se encuentre listo para la fase de pruebas. A esto se le llama la construcción del sistema. Las pruebas de integración aseguran que todas las partes han sido incluidas en el sistema y que sus interfaces son compatibles. Finalmente las pruebas del sistema validan las funciones y el desempeño del sistema de acuerdo a los requerimientos.

Estrategia de pruebas del sistema.

Consiste en evaluar cada una de las funciones bajo condiciones de estrés; evaluar la usabilidad y el desempeño. Una de las estrategias consiste en juntar todas las áreas funcionales tratando de cubrir todos los aspectos antes de pasar a la siguiente. La estrategia evita que las pruebas se dupliquen, pero no trabaja con el comportamiento del sistema.

Otra estrategia consiste en aplicar pruebas a las funciones de nivel más bajo en condiciones normales, anormales y de estrés. Después se deberá continuar al nivel superior y probar con funciones agregadas para verificar que funciona y se aplican las pruebas bajo condiciones normales y de estrés. Se deberán aplicar las mismas pruebas a los niveles superiores hasta cumplir con el sistema completo. La estrategia es útil para los sistemas de baja calidad porque existen funciones que no funcionan desde el inicio. La desventaja es que se requiere bastante tiempo para probar todas las combinaciones de funciones en un sistema grande.

La siguiente estrategia es el camino inverso. Se inicia con las funciones de nivel superior y posteriormente se pasa a niveles inferiores, aplicando pruebas bajo condiciones normales y de estrés. El sistema se prueba de acuerdo a la funcionalidad, casos de uso y guiones de pruebas.

Esta estrategia consiste en reunir todas las partes al mismo tiempo. Primero se tienen dos partes funcionando y después se agrega la tercera y se vuelven a poner a funcionar. Entonces se pueden poner la cuarta y quinta parte y así sucesivamente. La desventaja que tiene esta estrategia es que requiere mayor trabajo en la fase de pruebas.

Planeación de las pruebas.

Se realiza en diferentes momentos de acuerdo al proceso de TSPi. El plan de pruebas describe que se planea aplicar, el orden de las mismas y los materiales necesarios. Se necesita el plan de pruebas para construir las pruebas de integración, y las actividades de prueba del sistema. Aunque el construir y el planear las pruebas puede ser simple, es importante planear las pruebas antes de hacerlas.

Con el plan completo se debe mostrar la manera en la que cada requerimiento será probado y cómo cada escenario o guión cubre las áreas de los requerimientos, también se debe saber qué áreas serán probadas y cuáles tendrán un amplio cubrimiento. Además se deberá nombrar a cada prueba por anticipado, estimar los defectos encontrados en cada fase, el tiempo total en la detección de defectos y el tiempo de pruebas totales. Después se deberán estimar los materiales requeridos.

Al final de la planeación de las pruebas se deberá tener:

- Una lista de todos los pasos que se van a desarrollar en las pruebas.
- Los materiales requeridos, las características para cada una y su tiempo de desarrollo.
- Los posibles resultados que generarán.
- Estimación del tiempo de ejecución libre de defectos, de los defectos que serán encontrados y del tiempo total para cada prueba.
- Estimación del trabajo requerido.
- Los objetivos de cada prueba.
- El responsable de desarrollar cada material y la fecha final de terminación.

Cuando estas pruebas se ejecutan, se deberá tener lo siguiente en el registro de pruebas:

- El nombre de quien registra la prueba.
- Nombre o número de las pruebas.
- El producto y configuración que se está probando.
- Los resultados de las pruebas.
- Herramientas o facilidades utilizadas.
- Si se requirió la intervención de algún operador y cuánto tiempo.
- Se deberá anexar el formato de registro de pruebas en la carpeta.

Para estas funciones es imprescindible proponer la utilización de los formularios **LOGPRUEBA** (Ver Anexo 15), **LOGT** (Ver Anexo 9), los cuales se deben actualizar. En el primero se guardarán los datos referentes a las pruebas y en el segundo el tiempo que tomó cada prueba.

Módulo de defectos.

Cuando las pruebas no están bien pensadas, se pueden generar muchos errores después de las pruebas. En otras palabras, la mayoría de los errores que se encuentren en las pruebas muchos de ellos no se podrán corregir. Utilice los datos de prueba para evaluar el riesgo que el sistema tendría en una o más partes defectuosas. Ordene los datos de defectos por módulo para encontrar cuales tuvieron la mayoría de los defectos encontrados en cada prueba. Generalmente, aquellos módulos con la mayoría de los defectos mantendrán los mismos después de las pruebas y de manera general no serán encontrados. Se aconseja que los módulos con esas características se suspendan las pruebas temporalmente; se realice nuevamente la inspección y corrijan tales componentes defectuosos antes de continuar con las pruebas.

Para esta función es necesario proponer el formulario **LOGD** (Ver Anexo 13), en el se registrarán los defectos encontrados en las pruebas.

Seguimiento en los datos de defecto.

Para el seguimiento y análisis de los defectos por módulo, se necesitan los datos de cada defecto para cada prueba. Es importante que en las reuniones semanales se revisen los defectos encontrados en la construcción, integración o pruebas del sistema con el equipo completo. Posteriormente el administrador de calidad y proceso deberá dirigir una revisión de todas las pruebas de construcción e integración, y de los defectos de las pruebas del sistema con el equipo completo. También alguien deberá buscar y arreglar los defectos no encontrados. Se deberá actualizar el proceso de acuerdo a los cambios identificados.

Documentación.

En la fase de pruebas de TSPi, algunos miembros del equipo hacen un bosquejo y las revisiones de la documentación del usuario mientras que el resto del equipo realiza las pruebas. Durante el primer desarrollo del ciclo es buena idea asignar más ingenieros a las pruebas y después en los ciclos subsecuentes incrementar el número de ingenieros asignados a la documentación.

La documentación es parte esencial de cada producto de software. El mejor momento para documentar una función es después de que se ha diseñado. Por otro lado si se aplaza el trabajo de documentación, este podría tomar mucho más tiempo. TSPi incluye el trabajo de generación de documentación en la fase de pruebas.

Diseño de la documentación.

Después de que se construyó el producto, es natural querer describir lo que se construyó. Escribir manuales de usuario útiles es un reto para los ingenieros de software. Una guía completa para la determinación de la calidad de un manual es la tabla de contenidos. Si el manual está organizado con respecto al diseño del producto, esto debe de seguir las necesidades del lector y no de la estructura del producto. En general la primera sección explica al usuario cómo empezar. Después se deberá explicar al usuario qué puede hacer con el producto. A continuación se presenta una guía útil:

- Utilizar un glosario para definir los términos que no estén en el diccionario estándar.
- Incluir una sección con mensajes de error y de recuperación de errores.
- Tener un índice de términos importantes.
- Proporcionar una tabla de contenido detallado.

Línea de documentación.

El primer paso en la documentación es producir un bosquejo detallado. Empezar con un nivel alto y después diseñarlo. Antes de empezar a escribir el texto, verificar el trazo para asegurar que se cubren todos los objetivos en la construcción de los escenarios de usos. La única excepción son aquellos escenarios que serán cambiados en subsecuentes ciclos en el desarrollo. Si se describen en ese momento probablemente sería pérdida de tiempo.

Después de que se termina la línea de documentación se revisa con los ingenieros que están creando el plan de pruebas para asegurar que se entiendan las mismas funciones en la misma dirección y que nadie ha omitido algo importante. La documentación y el plan de pruebas dan una perspectiva diferente para el uso. De hecho el equipo encuentra más defectos en la documentación y en la planeación de las pruebas que durante las pruebas.

Estilo de escritura.

En general hay que escribir frases cortas, utilizar palabras simples y aplicar viñetas cuando sea necesario. Cuando se organizan las palabras en forma de lista, es más fácil para el usuario entender el trabajo y ver cómo se hace éste.

Revisión de la documentación.

Se aconseja que uno o más colegas revisen la documentación generada. Si el producto es para clientes o usuarios, aplique pruebas de usuario para asegurarse de que la escritura clara, precisa, completa y entendible. A continuación se presentan los siguientes elementos para las revisiones:

- Organización de los documentos. ¿El documento está organizado de acuerdo a lo que el usuario hará o de acuerdo a sus necesidades y no a la estructura del contenido del producto?

- Terminología del documento. ¿El documento presume conocimiento que el usuario probablemente no tenga? Debe definir cualquier palabra en el diccionario.
- Contenido del documento. ¿El documento cubre el contenido del producto?
- Precisión. ¿El desarrollo de los métodos y procedimientos es de acuerdo a su descripción?
- Legibilidad. ¿Es fácil de leer el documento? Lea el documento verificando que es fácil decir lo que está escrito.
- Entendimiento. ¿Las personas entienden lo que está escrito? Generalmente, esta pregunta es la más difícil de responder. Las mejores pruebas consisten en solicitar a alguien que no ha utilizado el sistema que lo haga. Obsérvese sus reacciones, registre los problemas y modifique el documento de acuerdo a los problemas identificados.

2.2.8 Fase de Postmortem.

En el Postmortem se revisa el trabajo del equipo para asegurar que se han completado todas las tareas necesitadas y todos los datos requeridos y registrados. También se revisa lo hecho en el ciclo, qué fue lo correcto y ver cómo hacer el trabajo mejor para el siguiente ciclo. El postmortem nos da un camino ordenado para identificar las áreas que necesitan ser mejoradas y crear los cambios necesarios.

En TSPi el ciclo concluye en la fase de postmortem, por lo cual nos da un camino estructurado para aprender y mejorar. En el postmortem se examina qué se hizo comparado con lo que se planeó hacer. Se busca oportunidades para mejorar y decidir si cambiar las prácticas para el siguiente ciclo.

En la fase de postmortem, se verán los cambios para el otro ciclo. En el primer ciclo TSPi da la línea base; se evalúan los productos producidos, el esfuerzo realizado, y los pasos del proceso que se deben seguir. Se determina qué tan adecuado fue el plan con el proceso seguido. Se identifican los problemas, determinan sus causas, se trazan medidas de prevención. La fase de postmortem verifica oportunidades de mejoras y específicamente se decide dónde y cómo incorporar esos cambios en el personal y en el proceso del equipo.

Propósito de mejora en el proceso.

La clave en las mejoras exitosas son los cambios pequeños. Cada mejora pequeña ayudará un poco y combinando esos pequeños cambios se verá un cambio significativo. El problema con los cambios pequeños es que son fáciles de olvidar por lo que se propone el formulario **PIP** (Ver Anexo 16) para registrar cada cambio que se efectúe. Estas ideas pueden incluir en las mejoras prácticas personales, mejoras de las herramientas y cambio en los procesos.

Revisión de calidad.

Como parte de la revisión del proceso se compara el desempeño personal y del equipo de acuerdo al plan de calidad. Se inicia con un análisis sobre los datos referentes a los errores y se evalúa uno o más errores en las pruebas del sistema, se deberá evaluar la parte del proceso que tuvo deficiencias para realizar mejoras en el futuro. Guíese mediante las siguientes preguntas:

- ¿Cómo comparó el desempeño actual con el planeado?
- ¿Cómo se puede aprender a partir de ésta experiencia?
- ¿Se debería aplicar un criterio personal o por equipo en ciclos posteriores?
- ¿Dónde se pueden aplicar mejoras y cuáles son las razones de ello?

Para postmortem posteriores evalúe las mejoras realizadas y juzgue la efectividad de las mismas. Guíese mediante las siguientes preguntas:

- ¿Tuvo deficiencias el equipo y dónde no?
- ¿Qué metas u objetivos son factibles de aplicarse en ciclos o proyectos posteriores?
- ¿Dónde se modificarían los procesos utilizados?

Evaluaciones de roles.

El líder del equipo examina cada rol, en estas evaluaciones se fijan los objetivos. Se inicia con el líder del equipo y después se revisan los otros roles, para hacer esto se consideran las siguientes preguntas para cada rol:

- ¿Dónde surgieron los problemas?
- ¿Dónde se pueden realizar mejoras?
- ¿Qué metas mejoraron y cuales de ellas podrían utilizarse para el siguiente ciclo o proyecto?

Preparar el reporte del ciclo.

Se reporta lo que se produjo, el proceso utilizado y los roles desempeñados; lo que si se pudo aplicar y lo que no fue conveniente, para ello se pueden hacer sugerencias que pueden aplicarse en el siguiente ciclo. Se deberá describir el desempeño que cada integrante tuvo respecto al rol que desempeñó, de acuerdo a los indicativos por TSPi así como el rol correspondiente al área de desarrollo. Si es posible,

se justifican las conclusiones con datos actuales del equipo y se comparará el desempeño en ciclos anteriores. Tal reporte debe ser breve y objetivo.

El reporte del ciclo deberá tener el siguiente contenido:

- Índice.
- Resumen.

Reporte de rol, en cuanto a:

- Liderazgo.
- Desarrollo
- Planeación.
- Proceso.
- Calidad.
- Apoyo o soporte.
- Reporte de los ingenieros.

Reporte de roles.

El líder del equipo será el responsable de generar el índice y el resumen, en ese último se describirán fortalezas y debilidades de cada área brevemente. El en reporte de roles se sugiere añadir dos secciones, una en la que se describirá la evaluación sobre el rol desempeñado individualmente y la evaluación sobre el desempeño del equipo de acuerdo a la perspectiva de TSPi sobre cada rol. El objetivo de este reporte consiste en proporcionar una guía para desempeñar de mejor manera cada rol tomando en cuenta las recomendaciones escritas.

El líder del equipo deberá analizar el desempeño del equipo desde la perspectiva del liderazgo, tomando en cuenta asuntos de motivación y acuerdos establecidos. También deberá tomar en cuenta las reuniones del equipo y mejorar esa responsabilidad en el futuro.

El administrador de desarrollo deberá comparar el contenido del producto con los requerimientos y evaluar la efectividad de la estrategia de desarrollo; para ello deberá analizar si la estrategia generó los resultados esperados y si deberá identificar que cambios podrían aplicarse. Respecto al diseño y a la implementación se deberá analizar el rehuso aplicado, el desempeño, la compatibilidad, instalación, mantenimiento, seguridad, etcétera.

El administrador de planeación compara el desempeño del equipo real con lo que se planeó, tomando en cuenta las horas semanales y los valores ganados, también analizar el desempeño del equipo en ciclos anteriores en otros proyectos.

El administrador de calidad y proceso compara los datos actuales con los objetivos de calidad establecidos para describir el desempeño del equipo. Muestra las tendencias de calidad de los ciclos de desarrollo terminados hasta la fecha actual. En la sección de proceso deberá evaluar la disciplina del mismo, tomando en cuenta hasta qué grado los integrantes siguieron el proceso, también deberá revisar el seguimiento en los estándares establecidos y en las revisiones aplicadas.

El administrador de configuración describe las facilidades de soporte, problemas enfrentados y mejoras en cualquier área. Hace comentarios sobre el procedimiento de control de cambios y la administración de configuración, tomando en cuenta aspectos que intervinieron en los cambios, los impactos de los mismos, hace recomendaciones para un mejor manejo de ciclos o proyectos futuros.

Reporte de ingenieros.

Cada ingeniero debe describir el desempeño personal de acuerdo a lo planeado y la calidad del trabajo que realizó. Es importante indicar cómo mejorar el trabajo para la siguiente vez. Cada ingeniero deberá entregar al líder del equipo sus comentarios. El administrador de calidad y proceso revisa el reporte completo y sugiere las correcciones necesarias. Cuando el reporte final del ciclo esté correcto deberá entregarse al instructor y a los integrantes del equipo.

Evaluaciones de los roles.

Consiste en llenar el formato de Evaluación del Equipo por Colegas. En tal formato se evalúa el desempeño del equipo y de cada rol. En la primera sección se evalúa el trabajo requerido y la dificultad de los roles en cada una de las actividades respectivas. Se recomienda utilizar el formulario **PEER** (Ver Anexo 17).

Además de estos artefactos que se generan en la fase de Postmortem, se debe obtener el resumen del plan de proyecto, es decir, los datos de la forma **SUMP** (Ver Anexo 12) que deben estar actualizados.

2.3 Herramienta Propuesta:

Se aconseja hacer uso de la herramienta Dashboard, esta tiene el objetivo de captar el estado real del trabajo actual durante el desarrollo del software. En un espacio muy pequeño el Dashboard le da acceso a un cronómetro para medir las actividades, un formulario de entrada de defectos para capturar información sobre los defectos, guiones y formularios a seguir para procesos definidos.

Un checkmark que permite una rápida y fácil manera de especificar que la fase de desarrollo ha sido completa, menús que permiten la navegación a través de la jerarquía de trabajo del proyecto, y un menú de configuración que permite acceder a otras opciones de la herramienta. Esta herramienta permite el acceso al método PROBE. Además las métricas pueden ser medidas rápidas, fácil y acertadamente.

Esta herramienta es muy útil para llenar los formularios propuestos, estos siempre están esperando por los datos que se le van a introducir y el Dashboard los registra sin la necesidad de utilizar botones de aceptar o enviar.

2.4 Conclusiones Parciales.

En el desarrollo de este capítulo ha quedado expuesta la estrategia a aplicar en el Polo PetroSoft, la cual fue el resultado de la investigación desarrollada para la elaboración de este trabajo de tesis.

Como parte de esta investigación en la encuesta realizada a líderes del Polo se obtuvo como resultado un pobre uso del Proceso de Software en Equipo, lo que constituyó la base para el planteamiento de la estrategia que fue planteada como solución de este trabajo de diploma en dicho Polo con el objetivo de lograr un proceso de desarrollo de software con mayor calidad.

Capítulo 3

Validación de la Estrategia

3.1 Introducción.

En el presente capítulo se evalúa la estrategia propuesta en la investigación por un personal con alta capacitación y conocimiento sobre el tema. Con este fin se utiliza el Método de multicriterio de expertos. Es un método que está basado en la experiencia y conocimiento de personas considerados especialistas en la materia a tratar.

3.2 Evaluación por el Método de Multicriterio de Expertos.

Se le denomina experto, tanto al individuo en sí como a un grupo de personas u organizaciones capaces de ofrecer valoraciones conclusivas de un problema en cuestión y hacer recomendaciones respecto a sus momentos fundamentales con un máximo de competencia.

El método de evaluación de expertos se emplea para comprobar la calidad y efectividad de los resultados de las investigaciones, tanto en su concepción teórica como en su aplicación en la práctica social, es decir, el impacto que se espera obtener con la aplicación de los resultados teóricos de la investigación en la práctica.

Generalmente la evaluación multicriterio de expertos se encuentra asociada a los conceptos:

- Decisión: Elección de una de las alternativas posibles para solucionar un problema.
- Alternativas: Cada una de las soluciones posibles a un problema, dotadas de ventajas e inconvenientes diferentes.
- Criterios: Distintos aspectos de la realidad que inciden de alguna manera en las ventajas o inconvenientes de las alternativas disponibles como soluciones al problema.

3.3 Selección de los expertos.

Se efectúa la selección de 5 expertos. Los mismos deben contar con una reconocida experiencia y prestigio profesional avalados por su alta calificación científico-técnica, conocimiento profundo de la Calidad de Software y Gestión de proyectos, haber obtenido resultados satisfactorios en la producción o la investigación. Se necesita que los expertos cuenten con una serie de características, como son:

- Competencia.
- Creatividad.
- Disposición a participar en la encuesta.
- Conformidad.
- Capacidad de análisis.
- Espíritu colectivista y autocrítico.
- Efectividad de su actividad profesional.

El grupo de expertos seleccionados estará integrado por:

Identificador	Expertos
E1	Ing. Mabel Guerra Saumell
E2	Ing. Gretchen Guillermo Hernández
E3	Ing. Armando Ortíz Cabrera
E4	Ing. Alfonso Chaveco Laurencio
E5	Ing. Ramses Ibarrola Suárez

Tabla 5: Expertos e Identificadores del Proceso de Evaluación de Expertos.

3.4 Búsqueda de alternativas.

Es de notar que en este método se parte, precisamente, de buscar las diferentes alternativas que se pretenden valorar. Luego de haberlas determinado, se procede con el proceso de consenso y selección de la mejor alternativa. Una vez determinadas las posibles acciones se indaga entre los expertos si consideran que esas son las que conformarán la lista final, o en caso contrario, si existen desacuerdos al respecto.

Las alternativas que se proponen para la valorar las propuestas están basadas en cuatro criterios fundamentales:

- Criterio del Método Científico: referente al nivel de calidad de la fundamentación teórica
- Criterios de Implantación: encierra los aspectos relacionados con usabilidad del modelo.
- Criterios de Generalización: recoge las acotaciones concernientes al nivel de comprensión de la investigación y la propuesta.
- Criterios de Impacto: posibilidades de trascendencia de la propuesta.

Estos criterios se especifican en las siguientes alternativas:

Indicadores	Alternativas
A1	Nivel de Calidad de la Investigación.
A2	Aportes Científicos Novedosos.
A3	Novedad Científica de la Investigación.
A4	Necesidad de Uso de la estrategia.
A5	Satisfacción de las necesidades de la producción.
A6	Nivel de comprensión.
A7	Facilidades de uso.
A8	Nivel de adaptación a diferentes entornos de producción de SW.
A9	Contribución al proceso de desarrollo de SW.
A10	Posibilidades de aplicación.
A11	Posibilidades de automatización.

Tabla 6: Alternativas e Indicadores del Proceso de Evaluación de Expertos.

3.5 Ponderación de las Alternativas.

Los expertos asignan ponderación o peso a cada una de las alternativas, con el objetivo de ordenarlas según su influencia en le aspecto tratado, o según su importancia. En este caso se considerará que el peso estará entre 1 y 5, y significa la opinión que brinda un experto sobre la alternativa a tratar.

5- Excelente.

4- Es muy buena, sin embargo se pudo haber perfeccionado.

3- Es Buena.

2- Es insuficiente.

1-Se recomienda que no sea considerada una alternativa a valorar, pues no se cumple en la investigación con los objetivos de la alternativa.

Posteriormente se intentará conocer en qué alternativas los expertos concuerdan, para ello se calcula de la siguiente manera el:

Coeficiente de concordancia.

$C_c = (1 - V_n / V_t) * 100$

Donde:

C_c: coeficiente de concordancia

V_n: cantidad de expertos en contra del criterio predominante

V_t: cantidad total de expertos

$60 \leq C_c \geq 85$ de acuerdo al grado de precisión.

$R_j = \Sigma C_{ij}$

C_{ij}: opinión del experto i para la acción j.

De esta manera, se obtuvo la valoración de los expertos en cada una de las alternativas a valorar. Se muestran los resultados arribados en la siguiente tabla:

Alternativas	Expertos						
	E1	E2	E3	E4	E5	Cc	Rj
A1	5	5	4	4	4	60	22
A2	4	4	3	3	3	60	17
A3	5	5	3	3	3	60	19
A4	5	5	5	5	4	80	24
A5	5	5	4	5	4	60	23
A6	5	3	4	4	4	60	20
A7	4	5	5	5	3	60	22
A8	5	5	3	5	4	60	22
A9	5	5	3	5	4	60	22
A10	5	5	3	5	3	60	21
A11	5	2	4	4	4	60	19

Tabla 7: Asignación de Pesos a las Alternativas.

3.6 Análisis de los resultados de la evaluación del Modelo.

Según la encuesta (Ver Anexo 18) realizada a los expertos se logró un elevado nivel de concordancia entre ellos al valorar las alternativas propuestas. Este resultado se puede confirmar con los valores del coeficiente de concordancia obtenidos en la Tabla 7, donde todos los valores cumplen con el grado de precisión establecido ($60 \leq C_c \geq 85$), coincidiendo en que:

La investigación cuenta con excelentes niveles de calidad, y de aportes novedosos. La propuesta brinda una alta contribución al proceso de desarrollo de software, y en la gestión de proyectos. Es adaptable a los diferentes entornos de producción de software, satisfaciendo así las necesidades de producción.

La propuesta tiene buen nivel de comprensión y facilidad de uso. Brinda la posibilidad de ser aplicada en los proyectos de desarrollo de software de la UCI.

Para valorar la calidad de la investigación se contó con las opiniones de 5 expertos en el tema (Ver Anexo 19).

3.7 Conclusiones parciales

En el capítulo que acaba de ser presentado se ha establecido la validación de la estrategia a través del método de multicriterio de expertos, para el mismo se contó con la valoración de 5 expertos en el tema los cuales fueron escogidos por sus calificaciones científico técnicas y los resultados alcanzados en su labor profesional. La valoración fue muy buena, se obtuvieron opiniones favorables de la estrategia presentada en el capítulo 2.

Conclusiones

Con este trabajo se logró proponer una estrategia del Proceso de software en Equipo para mejorar la calidad de los productos de software en los proyectos del Polo PetroSoft.

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

- No existen los conocimientos necesarios del Proceso de Software en Equipo en el Polo PetroSoft, no utilizan este procedimiento para expandir este tema tan importante que es la calidad de software.
- La reducida experiencia del personal hace que los productos no se desarrollen de la forma debida, lo que trae consigo que la calidad de los productos no sea la esperada, retrasos en los tiempos planificados, aumento de los costos y la completa insatisfacción del cliente para el cual se trabaja.
- No utilizan al máximo los recursos disponibles para la realización de los productos.
- No realizan la estimación de los riesgos, ni planifican el trabajo como es debido.
- Se espera que con la estrategia propuesta se resuelvan todos los problemas existentes en dicho Polo.

Recomendaciones

Establecer la estrategia de Proceso de Software en Equipo en los proyectos del Polo productivo PetroSoft y utilizarla como referencia para otros proyectos de desarrollo de software independientemente de su naturaleza.

- Todos los integrantes del Polo PetroSoft deben llevar a cabo una capacitación más profunda sobre todo lo relacionado con Proceso de Software Personal ya que si no tienen una previa capacitación en este proceso no van a entender lo relacionado al Proceso de Software en Equipo.
- Realizar un estudio más profundo del proceso de Software en Equipo con el fin de que posean una cultura general sobre el tema y se tome conciencia de la importancia de este proceso para poder llevar a cabo un producto con la calidad requerida.
- Analizar y aplicar consecuentemente las fases planteadas en proyectos de dicho Polo.
- Realizar un seguimiento del trabajo de estos proyectos bajo este nuevo enfoque y velar por los resultados de su aplicación.
- De resultar exitosa la estrategia en estos proyectos, proponer su extensión a todos los proyectos productivos que conforman el Polo.
- Estudiar la posible aplicación de esta guía en otros Polos y proyectos de la universidad.

Referencias Bibliográficas

1. Palacio, Juan. Gestión y procesos en empresas de software. Noviembre de 2005. Disponible en: [http://www.navegapolis.net/content/view/156/59/]
2. Sánchez, M.A.M., Metodologías de Desarrollo de Software. 7 de junio del 2007. Disponible en: [http://www.informatizate.net/articulos/metodologias_de_desarrollo_de_software_07062004.html]
3. Bayona, S., Calvo Manzano, J., Cuevas, G., San Feliu, T., Team Software Process (TSP): Mejoras en la estimación, calidad y productividad de los equipos en la gestión del software. 1 de enero 2007. VOL. 4. Disponible en: [http://www.aemes.org/rpm/descargar.php?volumen=4&numero=1&articulo=2.]
4. Vereau, E., Mejores prácticas para desarrolladores freelance. Introducción al PSP. Junio de 2006. Disponible en: [http://www.vereau.org/wp-images/images/files/pres-psp-chiclayo.pdf]
5. Casañola, Y. T., Evaluación teórica de la adopción del enfoque de Factorías de Software. 2007. Universidad de las Ciencias Informáticas. Disponible en: [http://www.intempres.pco.cu/Intempres2006/Intempres2006/Evaluacion%20de%20trabajos/Yaim%ED%20Trujillo%20Casa%F1ola%20P.pdf]
6. Esperanca Amengual, A. M., Como puede utilizarse TSP para mejorar un proceso según la norma ISO/IEC 15504. Barcelona 3 de Octubre de 2006. Disponible en: [http://webdiis.unizar.es/~franjr/slides/ADIS06.pps]
7. Cerrillo, D., Estimación del Software. Mayo de 1999. Universidad de Castilla-La Mancha. Disponible en: [http://alarcos.inf-cr.uclm.es/doc/pgsi/doc/esp/T9899_DCerrillo.pdf]

8. SEI, Capability Maturity Model Integration (CMMISM), 2000, disponible en: [http://www.sei.cmu.edu/cmm/cmms/cmms.integration.html]

9. Estrada, M.L.A.F., Medir el proceso de control de configuración, ¿Una utopía para la Industria Nacional de Software? Instituto Superior Politécnico "José Antonia Echevarría" Ciudad Habana. 2001. Disponible en:
[www.inf.udec.cl/revista/ediciones/edicion9/febles.pdf]

10. Fernández, G., Estándar de Codificación. Abril 2005. Disponible en:
[http://www.elguille.info/colabora/NET2005/giovannyfernandez_EstandarCodificacionNET.htm]

11. Luna, D. A. A., "Personal Process Software and Team Software Process". Instituto Tecnológico y de estudios superiores de Monterrey. Noviembre 2006. Disponible en:
[http://homepages.mty.itesm.mx/al1106336/PSP_TSP.doc]

12. McHale, J., A PSP/TSP Approach to CMMI Transition. Retrieved November 11, 2006. Disponible en:
[http://www.dtic.mil/ndia/2001cmmi/mchale.pdf]

13. Upchurch, R (2001). Personal Software Process. Retrieved November 11, 2006. Disponible en:
[http://www2.umassd.edu/SWPI/PersonalSoftwareProcess/PSP.html]

14. Monasor, M. J., Calidad de Proceso. Una orientación hacia el desarrollo distribuido del software. Universidad de Castilla-La Mancha. Enero 2008. Disponible en:
[http://alarcos.inf-cr.uclm.es/doc/cmsi/trabajos/Miguel%20Jimenez%20-%20Calidad%20de%20Proceso%20-%20Doc.pdf]

Bibliografía

1. Ramiro, R.A., Tarea 2 Ingeniería de Software. 2007. Disponible en:[http://rlydj.blogspot.com/2007/08/tarea-2.html]
2. Humphrey, W.S., Introduction to the Team Software Process. 1999.
3. Humphrey, W.S., Introduccion al Proceso de Software Personal. 1989.
4. Fugetta. Software Process. In International Conference on Software Engineering. 2000.
5. Blaya, I., Gestión por procesos. 2006.
6. Pressman, R.S., Ingeniería del Software. Un enfoque práctico. Quinta edición ed. 2002.
7. Lídice D Alón, H.D.P., Propuesta de Mecanismos de Gestión de Calidad Interna para el proyecto Registro y Notarías. 2007, Universidad de las Ciencias Informáticas.
8. Palacio, J., Gestión y procesos en empresas de software 2005. Disponible en:

   [http://www.navegapolis.net]
9. Scalone, F., estudio comparativo de los modelos y estándares de calidad del software. 2006, Universidad Tecnológica Nacional Facultad Regional: Buenos Aires.
10. Ruvalcaba, M. (2005) Procesos de Software. Disponible en: [http://www.sg.com.mx/content/view/4/11/]
11. Humphrey, W., S., Introduction to the Team Software Process. TSPi.
12. Cuevas, A.G., Gestión del Proceso de software. 2003.
13. Humphrey, W.S., Addison, W., Introduction to the Team Software Process. 2005.
14. Humphrey, W.S., Addison, W., Coaching Development Teams. 2006.
15. Jacobson I., G. B., J Rumbaugh. Addison W, the Unified Software Development Process.1999.
16. Piattini, García, "Calidad en el desarrollo y mantenimiento del software", RA-MA Editorial, Madrid, 2003.
17. Sommerville, L., Ingeniería del Software. Séptima Edición ed. Madrid 2005.

18. Yordanis P, Gómez, Jacqueline, G, C., Principios estratégicos para la guía del proceso de desarrollo de software educativo en la Universidad de las Ciencias Informáticas. Julio 2007. Ciudad Habana.

19. Fernández, J. A. V., TSP/PSP Dirigiendo la Mejora de Procesos desde las Trincheras del Desarrollo del Software. Centro de excelencia para la industria del Software. Tecnológico de Monterrey.

20. Amengual, E., El trabajo en Equipo. Noviembre 2007.

Anexos

Anexo 1

Entrevista al líder del proyecto SAGUIR.

1. ¿Utilizan el Proceso de Software en Equipo en el proyecto?
2. ¿Estrategia de trabajo que utilizan?
3. ¿Como es el desarrollo del proyecto, Rápido_____, Lento_____ o de acuerdo a lo planificado_____?
4. ¿Realizan una adecuada planificación del trabajo?
5. ¿Cómo se desarrolla la Gestión de Riesgos?
6. ¿Qué actividades realizan para motivar al personal involucrado en el proyecto?

Anexo 2

Entrevista a personal involucrado en el proyecto SAGUIR

1. ¿Los miembros del proyecto reciben cursos de superación?
2. ¿El líder del proyecto cuenta con los conocimientos básicos del proceso de desarrollo de software?
3. ¿Se lleva un control detallado del trabajo que se esta realizando?
4. ¿Los productos son entregados a tiempo?
5. ¿Los productos salen con la calidad esperada por el cliente?

Anexo 3: Formulario INFO (Hoja de información del estudiante)
(Tomado de: Humphrey, W.S., Introduction to the Team Software Process. 1999.)

Nombre: _____ Instructor: _____
Fecha: _____ Notas anteriores: _____
Mejor: _____ Fecha de graduación esperada: _____

Breve descripción de su experiencia relevante y sus intereses:

_____
_____
_____

Breve descripción de su trabajo en otros equipos de proyectos:

_____
_____
_____

Breve descripción de cualquier cargo de dirección o liderazgo que haya tenido (en el trabajo o en cualquier organización)

_____
_____
_____

Si la tiene, establezca sus preferencias de equipos:

_____
_____

Liste su plan de clases y otros tiempos planificados para actividades tales como deportes, trabajo, recreación, etc.							
Tiempo	Lunes	Martes	Miércoles	Jueves	Viernes	Sábado	Domingo
8:00-9:00							
9:15-10:15							
10:30-11:30							
11:45-12:45							
13:00-14:00							
14:15-15:15							
15:30-16:30							
16:45-17:45							

Diga el rango de preferencia 1(menos) 5(más) para servir en los roles del equipo					
Líder del Equipo	1	2	3	4	5
Director de Desarrollo	1	2	3	4	5
Director de Planificación	1	2	3	4	5
Director de Calidad/Proceso	1	2	3	4	5
Director de Soporte	1	2	3	4	5

Instrucciones para llenar el Formulario INFO

Objetivo	Usar este formulario para describir tus intereses y experiencias
General	• Completar este formulario y entregar al instructor durante el primer período de laboratorio del curso de TSP_i • El instructor lo usará para formar los equipos y asignar los roles • La información de planificación es necesaria tanto para formar los equipos como para conocer cuando reunirse en la semana • Para preguntas acerca de los roles ver Anexo E y en la parte III • Usar páginas adicionales si fuera necesario
Encabezamiento	Entrar • Tu nombre, el del instructor y la fecha • El número de créditos que esperas del curso • Tu mejor campo de estudio • Tu fecha de graduación esperada
Experiencia relevante e intereses	• Listar cualquier experiencia e intereses que consideres ayude al instructor a formar el equipo y asignar los roles • Ejemplo: experiencia con PSP, diseño y desarrollo de bases de datos, etc.
Otros proyectos en equipo	• Listar cualquier experiencia en equipos que consideres ayude al instructor a formar el equipo y asignar los roles • Ejemplo: tipos de proyectos, roles ejecutados, las herramientas y los métodos usados
Liderazgo o dirección	• Listar cualquier experiencia de liderazgo o dirección que consideres ayude al instructor a formar el equipo y asignar los roles • Ejemplos: director de negocio de un club, trabajar como asistente de enseñanza, etc.
Preferencias de equipos	• Si deseas trabajar con un grupo particular, menciónelos • De lo contrario, no necesitas decir nada
Plan	• Listar los horarios que tienes comprometidos para clases y otras actividades • Si los horarios no coinciden con el dado establezca el período aproximado y inclúyalo antes de los horarios precisos
Preferencias de roles	• Establezca las preferencias de roles de 1(menos deseados) a 5(más deseados) • Note que puede listar varios roles como 1 o como 5, si se siente que estos son igualmente deseados o indeseados

Anexo 4: Formulario Semana (Reporte de estado semanal)
(Tomado de: Humphrey, W.S., Introduction to the Team Software Process. 1999.)

Nombre **Equipo** **Instructor**
Fecha **Ciclo No.** **Semana No.**

Datos semanales	Planificado	Real
Horas proyecto para esta semana.		
Horas proyecto en este ciclo hasta la fecha.		
Valor devengado para esta semana.		
Valor devengado en este ciclo hasta la fecha.		
Total de horas para las tareas terminadas en esta fase hasta la fecha.		

Datos semanales de los miembros del equipo.	Horas Planificadas	Horas Reales	Valor Planificado	Valor Obtenido
Líder del equipo.				
Director de desarrollo.				
Director de planificación.				
Director del proceso/ Calidad.				
Director de soporte.				
Totales				

Tareas a desarrollar terminadas.	Horas Planificadas	Horas Reales	Valor Obtenido	Semana Planificada
Totales				

Seguimiento de los issue y riesgos.	
Nombre de issue y riesgos.	Estado

Otros elementos significativos.

Instrucciones para llenar el Formulario Semanal (reporte de estado semanal)

Propósito	• Use este formulario para preparar reportes de estado semanal.
General	• Cada miembro del equipo completará este formulario cada semana mostrando el trabajo terminado la última semana y la planificación para la siguiente semana. • Cada semana, el director de planificación prepara una copia del formulario semanal con un resumen del estado del equipo y los cumplimientos de la semana. • Adjuntar hojas adicionales si es necesario.
Encabezamiento	• Entre su nombre y el del instructor. • Entre el nombre del equipo, No. del ciclo, fecha y numero de la semana.
Datos semanales	• Entre el total de horas realmente utilizada en el proyecto esta semana y las horas planificadas para la semana. • También entre las horas reales y planificadas acumuladas durante este ciclo de desarrollo. • Entre el valor planificado y el valor real obtenido para la semana. • Entre el valor planificado y el valor obtenido acumulados para el ciclo de desarrollo hasta la fecha. • Entre el total de horas planificadas y reales para las tareas terminadas en este ciclo de desarrollo hasta la fecha.
Datos semanales de los miembros del equipo	Para cada reporte de los miembros del equipo: • Entre el tiempo total real y planificado para cada ingeniero. • Entre el valor planificado y obtenido del ingeniero para la semana. • Entre el total de horas planificadas y real trabajadas del ingeniero.
Datos semanales del equipo	Para el reporte consolidado del equipo: • Entre el tiempo total real y planificado para el equipo. • Entre el valor planificado y obtenido para el equipo en la semana. • Entre el total de horas planificadas y real trabajadas por el equipo.
Tareas a desarrollar terminadas	Para las tareas terminadas esta semana: • Entre el nombre de cada tarea. • Entre el tiempo total real y planificado para esta tarea. • Entre el número de la semana en que fue planificada. • Entre el valor obtenido para la tarea.
Seguimiento del riesgo e issue	• Para los riesgos e issue rastreados, resuma el estado y cualquier cambio importante esta semana.
Otros elementos significativos	• Liste cualquier logro o evento significativo ocurrido durante la semana. • En el caso de los roles podría ser estándares de codificación completados, cambios en los procedimientos de control aprobados. • En caso de los desarrolladores diseño, codificación, inspección o prueba de varios elementos del producto.

Anexo 5: Formulario TAREA (Plantilla para la planificación de tareas)
(Tomado de: Humphrey, W.S., Introduction to the Team Software Process. 1999.)

Nombre
Equipo
Fecha
Instructor

Datos ejemplo del equipo B

Parte/Nivel Cambiar Contador/Sistema
Ciclo 1

Tarea				Horas del plan							Plan tamaño/valor					Real		
Fase	Parte	Nombre de tarea	# de ingenieros	Líder del Equipo	Director de Desarrollo	Director de Planificación	Director de	Director de Soporte	Horas totales del equipo	Horas acumuladas	Tamaño de las Unidades	Tamaño	#. Semana	Valor planificado	PV Acumulado	Horas	Horas acumuladas	# Semana
St	S	Inicio y estrategia	5	2.9	5	1.3	1.3	1.3	11.8	11.8	Páginas	2	1	3.4	3.4	12.3	12.3	1
Mg	S	Manejo y Misc.	5	3.0	3.0	3.0	3.0	3.0	15.0	26.8			1	4.3	7.7	10.0	22.3	1
Pl	S	Tareas del plan y horario	5	4.5	4.5	9.0	4.5	4.5	27.0	53.8			2	7.7	15.4	13.3	35.6	2
Pl	S	Generar planes Q&T	5	3.5	3.5	3.5	5.0	3.5	19.0	72.8			2	5.4	20.8	9.9	5.5	2
Mg	S	Manejo y Misc.	5	3.0	3.0	3.0	3.0	3.0	15.0	87.8			3	4.3	25.1	3.4	48.9	2
Rq	S	Revisar estado necesitado	5	4.0	6.5	2.0	2.0	2.0	16.5	104.3	Páginas	3	3	4.7	29.9	17	65.9	2
Rq	S	Producir SRS	5	4.0	5.5	2.0	2.0	2.0	15.5	119.8	Páginas	54	3	4.4	34.3	15.6	81.4	3
Rq	S	Plan cheq. Sistema	0						0.0	119.8			3	0.0	34.3	0.0	81.4	3
Rq	S	Inspección de req.	4	2.0		2.0	2.0	2.0	8.0	127.8	Páginas	5	3	2.3	36.6	2.8	84.2	3
Mg	S	Manejo y Misc.	5	2.0	2.0	2.0	2.0	2.0	10.0	137.8			4	2.9	39.5	5.0	90.1	3
Hd	S	Diseño a alto nivel	5	2.0	3.5	1.0	2.0	1.0	9.5	147.3	Páginas	10	4	2.7	42.2	16.5	106.6	4
Hd	S	SDS	5	4.0	2.0	4.0	4.0	4.0	18.0	165.3	Páginas	20	4	5.2	47.3	9.5	116.1	4
Hd	S	Diseño a alto nivel	5	2.0	3.5	1.0	2.0	1.0	9.5	147.3	Páginas	10	4	2.7	42.2	16.5	106.6	4
Hd	S	Plan de integración	5				2.0	2.0	2.0	169.3	Páginas	10	4	2.7	42.2	16.5	106.6	4

Hd	S	Inspección HLD	5	2.5	0.3	2.5	2.5	2.5	10.3	179.5	Páginas	10	5	2.9	51.4	0.0	120.1	4
Mg	S	Manejo y Misc.	5	2.0	2.0	2.0	2.0	2.0	10.0	189.5			5	2.9	54.3	21.5	141.6	5
Dd	P	Diseño detallado	4	3.0	3.0	3.0		3.0	12.0	201.5	Líneas	137	5	3.4	57.7	40.2	181.3	5
Dr	P	Revisión DLD	4	2.0	2.0	2.0		2.0	8.0	209.5	Líneas	137	5	2.3	60.0	10.5	192.4	5
Td	5	Desarrollo de prueba	1					1.5	1.5	211.0			5	0.4	60.5	0.0	192.4	5
Di	P	Revisión DLD	5	3.0	3.0	3.0	3.0	3.0	15.0	226.0	Líneas	137	5	4.3	64.8	8.9	201.3	5
Cd	P	Código	5	3.6	3.6	3.6	3.6	3.6	18.0	244.0	Líneas	410	5	5.2	69.9	8.1	269.4	5
Cr	P	Revisión de código	5	3.0	3.0	3.0	3.0	3.0	15.0	259.0	LOG	410	5	4.3	74.2	7.3	216.7	5
C	P	Compilación	5	0.4	0.4	0.4	0.4	0.4	2.0	261.0	LOG	410	5	0.6	74.8	1.0	217.7	
Ci	P	Inspección de código	5	1.0	1.0	1.0	1.0	1.0	5.0	266.0	LOG	410	5	1.4	76.2	2.7	_220.4	5
UT	P	Chequeo de la unit	5	1.5	1.5	1.5	1.5	1.5	7.5	273.5	LOC	410	5	2.1	78.4	18.7	2759.1	
Mg	S	Manejo and 6c.	5	4.0	4.0	4.0	4.0	4.0	20.0	293.5			6	5.7	84.1	11.2	250.3	6
5t	S	Construcción e integración	3		2.0		2.0	2.0	6.0	299.5	LOG	410	6	1.7	85.8	1.6	251.8	6
St	S	Chequeo del sistema	3		4.0		4.0	4.0	12.0	311.5	LOC	410	6	3.4	89.3	15.0	266.9	6
Dc	S	Documentación	2	5.0		5.0			10.0	321.5	Página	15	6	2.9	92.1	1.6	268.4	6
I'M	S	Luego de terminado	5	2.0	2.0	2.0	2.0	2.0	10.0	331.5			6	2.9	95.0	2.0	270.4	6
Mg	S	Manejo y misc.	5	5.5	3.0	3.0	3.0	3.0	17.5	349.0			6	5.0	100	9.5	280.0	6

Totales			73.4	73.3	68.6	64.8	68.8	349					100		280		

Instrucciones para llenar la plantilla de planificación TAREA

Propósito	• Estimar el tiempo de desarrollo de cada tarea del proyecto. • Calcular el valor planificado por cada tarea del proyecto. • Estimar las fechas de terminación planificadas por cada tarea. • Proporcionar una base para dar seguimiento al progreso de la planificación incluso cuando las tareas no se hayan planificado en el orden planeado.
General	• Expanda este formulario o use varias páginas cuando lo requiera. • Incluya todas las tareas significativas. • Utilice nombres y números para las tareas que soporten la actividad y sean consistente con la estructura del producto y el proyecto. • Donde sea posible utilice la herramienta de TSP para planificar.

Anexo 6: Formulario Horario (Plantilla para planificar el horario)
(Tomado de: Humphrey, W.S., Introduction to the Team Software Process. 1999.)

Nombre — Fecha
Equipo Ejemplo de los datos del B — Instructor
Parte/Nivel Cambiar — Ciclo 1

		Plan			Actual			
No. de semana	Fecha	Horas directas	Horas acumuladas	Valores planeados acumulados	Horas del equipo	Horas acumuladas	Valor ganado en la semana	Valor ganado acumulativo
1	9/14	26.8	26.8	7.7	27.0	27.0	7.7	7.7
2	9/21	56.5	83.3	20.9	44.4	71.4	22.2	29.9
3	9128	47.0	130.3	36.6	35.8	107.2	9.6	39.5
4	10/5	47.8	- 178.1	48.5	64.0	171.2	9.0	48.5
5	10/12	107.0	285.1	78.4	75.4	246.6	29.9	78.4
6	10/19	63.9	349.0	100.0	33.4	280.0	21.6	100.0

Instrucciones para llenar el formulario de HORARIO

Propósito	• Registrar las horas estimadas y actuales gastadas por semanas. • Mostrar el valor planeado acumulativo por semana. • Rastrear valor devengado contra valor planeado a medida que se completan las tareas.
General	• Expanda esta plantilla o use páginas múltiples como necesite. • Complete en conjunto con el formulario TAREA. • Donde sea posible, use la herramienta de soporte TSPi para la planificación. • Si usa la herramienta de soporte TSPi, completará todos los cálculos para los formularios TAREA y HORARIO. • Si no, deberá hacer los cálculos usted mismo.
Encabezamiento	• Introducir nombre, fecha, equipo, y nombre del instructor. • Nombre de la parte o ensamble y su nivel. • Introducir número del ciclo.
Número de semana	Desde el comienzo del ciclo, entre el número de semana, comenzando con 1.
Fecha	• Entrar la fecha para cada semana • Escoger un día estándar en la semana por ejemplo lunes.
Plan: horas directas	• Introducir el número de horas planificadas para trabajar cada semana • El tiempo que no se trabaje considérelo como vacaciones, etc. • Considere otras actividades cometidas como clase ,reunión y otros planes
Plan:horas acumuladas	Introducir las horas planeadas planificadas a través de cada semana
Plan: valor planeado acumulado	Para cada semana tomar el plan de horas acumuladas del formulario HORARIO y : • En el formulario TAREA, encuentre la tarea de menor o igual plan de horas acumuladas y anote el PV(valor planeado) acumulado • Introducir el PV en el formulario HORARIO para esa semana • Si el valor acumulado para la primera semana aún continua éntrelo nuevamente
Real	Durante el desarrollo, introducir las horas reales , las horas acumuladas, y valor devengado acumulado para cada semana • para determinar el estado contra el plan, compare el valor planeado acumulado con el valor devengado acumulado real • Compare horas planeadas acumuladas con horas reales acumuladas • Si está atrasado en el horario y las horas reales son menores que las del plan, el tiempo gastado no es suficiente. • Si está atrasado en el horario y las horas reales son mayores o iguales que las del plan, el problema es de pobre planificación.

Anexo 7: Formulario ESTRAT (Plantilla para planificar la estrategia)
(Tomado de: Humphrey, W.S., Introduction to the Team Software Process. 1999.)

Nombre _____ Fecha _____
Equipo _____ Instructor _____
Parte / Nivel _____ Ciclo _____

Referencias	Funciones	LOC del ciclo			Horas de ciclo		
		1	2	3	1	2	3
Total							

Instrucciones para llenar el formulario ESTRAT

Propósito	• Este formulario es usado para registrar decisiones estratégicas • Es usado durante el desarrollo de la estrategia para asignar funciones del producto a ciclos • Es usado también en el diseño de alto nivel para asignar funciones de SRS a componentes
General	• Este formulario sugiere la forma de registrar decisiones estratégicas • Use este u otro formato que contenga los mismos datos
Encabezamiento	• Introducir nombre, fecha, equipo, y nombre del instructor • Nombre de la parte o ensamble y su nivel • Introducir número del ciclo
Referencias	Usar esta columna para listar la instrucciones necesarias, o los párrafos de SRS o el número de sentencia para cada función
Funciones	En esta columna, listar todas las funciones a ser incluidas en el producto en todos, los ciclos
LOC del ciclo	• Use esta columna para los LOC estimados para cada función • Introducir los LOC estimados para cada función bajo el número del ciclo que incluirá esa función • Si usted planea implementar la función en dos o incluso tres partes del ciclo, introducir lo nuevo estimado y las LOC cambiadas para cada ciclo • Si una de las funciones está incluida en las LOC de otra función, márquelo con una X
Horas del ciclo	• Use esta columna para el tiempo estimado que se requiere para desarrollar cada función • Introducir el tiempo estimado para cada función bajo el número del ciclo en el cual usted planea incluir la misma. • Si usted planea implementar la función en dos o incluso tres partes del ciclo, introducir el tiempo de desarrollo estimado para cada ciclo. • Si una de las funciones está incluida en las horas de otra función márquelo con una X

Anexo 8: Formulario ITL (Riesgos)
(Tomado de: Humphrey, W.S., Introduction to the Team Software Process. 1999.)

Nombre: _____ Fecha: _____
Equipo: _____ Instructor: _____
Parte/Nivel: _____ Ciclo: _____

Fecha	Riesgo/Problema	Número	Prioridad	Propietario	Fecha FU	Resuelto

Descripción: _____

_____

Instrucciones para llenar el Formulario ITL

Propósito	• Use este formulario para registrar y rastrear los problemas y riesgos del proyecto.
Responsabilidades	• El director de soporte mantiene el registro de rastreo de riesgo. • En cada reunión del equipo, el director de soporte proporciona el estado de los datos sobre los riesgos y problemas. • Ejemplos son el número de problemas o riesgos con las fechas, el número con fechas actuales, etc. • El equipo decide quién se encargará de cada riesgo o problema y cuándo.
General	• Los problemas son ciertos; sin acción, ellos probablemente causarán problemas. • Los riesgos son cosas que pueden o no ocurrir. • Registre un riesgo o problema en cada segmento del formulario. • Use tantas copias adicionales del formulario como necesite. • Si el número de riesgos o problemas se vuelve demasiado grande, use un sistema bases de datos para rastrearlos.
Encabezamiento	• Entre su nombre, fecha, nombre del equipo y nombre del instructor. • Nombre el ensamblaje y su nivel. • Entre el número del ciclo.
Fecha	• Entre la fecha en que el riesgo o problema se introdujo en el sistema ITL.
Riesgo/Problema	• Entre una R para un riesgo y una P para un problema.
Número	• Asigne un número de control para cada riesgo y problema.
Prioridad	• Para los riesgos, entre la evaluación del riesgo. • Use A, M o B para las evaluaciones altas, medias o bajas. • Evalúa el riesgo en términos de probabilidad e impacto del horario. • Una prioridad de AM puede ser un riesgo con una probabilidad alta de ocurrir y un impacto medio del horario sobre el proyecto.

Propietario	• Nombre del ingeniero que rastrea el riesgo o problema.
Fecha de seguimiento	• Anote la fecha para la que el riesgo o problema debe resolverse.
Fecha de resolución	• Registre la fecha en que el riesgo o problema fue resuelto.
Descripción	Describa el riesgo o problema tan claro como sea posible. Por ejemplo: • Un cambio a ser hecho o una interface a ser verificada. • Una pregunta de los requerimientos a responder. • Una posible demora en la obtención de una herramienta de soporte. Describa el riesgo o problema completamente para que más tarde te permita a ti o a alguien tomar una acción necesaria.

Anexo 9: Formulario LOGT (Registro de Tiempo)
(Tomado de: Humphrey, W.S., Introduction to the Team Software Process. 1999.)

Nombre: _____ Fecha: _____
Equipo: _____ Instructor: _____
Ensamblaje: _____ Ciclo: _____

Fecha	Inicio	Parada	Tiempo de interrupción	Delta tiempo	Fase/ Tarea	Ensamblaje	Comentarios

Instrucciones para llenar el Formulario LOGT

Propósito	• Use este formulario para registrar el tiempo gastado en cada fase del proyecto.
General	• Mantener un registro y anotar la tarea y elemento del producto para cada entrada o mantener registros separados para cada tarea mayor. • Registrar todo el tiempo gastado en el proyecto. • Registrar el tiempo en minutos. • Ser tan exacto como sea posible. • Si se necesita espacio adicional, usar otra copia del formulario. • Si se olvida el tiempo de inicio, parada o tiempo de interrupción para una tarea, entrar la mejor estimación.
Encabezamiento	• Entre su nombre, fecha, nombre del equipo y nombre del instructor. • Nombre el ensamblaje. • Entre el número del ciclo.
Fecha	• Entre la fecha cuando hizo la entrada. • Por ejemplo, 10/18/99.
Inicio	• Entre el tiempo cuando se comenzó a trabajar en una tarea. • Por ejemplo: 8:20
Parada	• Entre el tiempo cuando paró de trabajar en esa tarea. • Por ejemplo: 10:56
Tiempo de interrupción	• Registrar cualquier tiempo de interrupción que no fue gastado en la tarea y la razón por la interrupción. • Si tiene varias interrupciones, entre su tiempo total. • Por ejemplo, 37 – tomando un descanso.
Delta tiempo	• Entre el tiempo que realmente gastó trabajando en la tarea, menos el tiempo de interrupción. • Por ejemplo, desde las 8:20 hasta las 10:56 menos 37 minutos es 119 minutos.
Fase/Tarea	• Entre el nombre u otra designación de la fase o tarea en que trabajó. • Por ejemplo, planificación, programación, prueba, etc.
Ensamblaje	• Si la tarea fue para un único producto, entre el nombre del producto.
Comentarios	• Entre otros comentarios pertinentes que puedan recordarle más tarde cualquier circunstancia no usual con respecto a esta actividad. • Por ejemplo, tenía una pregunta de los requerimientos y necesitaba ayuda.

Anexo 10: Solicitud de configuración de cambio (CCR)
(Tomado de: Humphrey, W.S., Introduction to the Team Software Process. 1999.)

Formulario CCR

Nombre: _____ Fecha: _____
Equipo: _____ Instructor: _____
Parte/Nivel: _____ Ciclo: _____

Información del producto
Nombre del producto _____ Propietario del producto _____
Producto/Tamaño de cambio _____ Tamaño medido _____
Inspección reciente _____ Moderador _____

Dirección auxiliar

_____

Cambio en la información
Motivo del cambio:

_____
_____
_____
_____
_____

Beneficios del cambio:

_____
_____
_____
_____
_____

Impacto del cambio:

_____
_____
_____
_____
_____

Descripción del cambio: (Para el código fuente, unir el listado; para cambios en el código, incluir el número de defectos (si hay alguno) y el listado de los segmentos del programa cambiados y no cambiados)

_____
_____
_____
_____

Estado
Aceptad☐ Información Adicior☐: Re☐hazado:
Información necesaria:

_____
_____
_____
_____

Aprobación
Propietario del producto _____ Fecha:
_____
Calidad/Director del Proceso _____ Fecha:
_____
CCB _____ Fecha:
_____

Instrucciones para llenar el formulario CCR

Propósito	Este formulario es usado cuando someten un elemento a la pizarra de configuración de control por inclusión en el producto de la línea base.
General	• El elemento puede ser un producto para estar en la línea base o un cambio a un producto de la línea base. • Todos los productos nuevos son revisados y aprobados por el CCB antes de incluirlo como producto de la línea base. • Todos los cambios a los productos de la línea base deben someterse de antemano a la aprobación del CCB. • Una copia del producto o cambio debe proporcionarse con el CCR
Encabezamiento	• Entrar nombre, fecha, nombre del equipo, y nombre del instructor. • Nombre de la parte o ensamblaje y su nivel. • Entrar número del ciclo.
Información del producto	Entrar la siguiente información en el producto o cambio. • Nombre del producto o número de identificación • Dueño del producto , usualmente fabricador • Tamaño estimado o real del producto o cambio • Medida de tamaño usada ; páginas o LOC nuevas y cambiadas Si el producto ha sido inspeccionado, proveer: • Tipo de inspección y cuando fue hecha • Nombre del moderador de la inspección • Una copia del formulario INS. Describa la ubicación de la copia de seguridad del producto o donde se retienen los cambios. También de la dirección del archivo.
Cambio de información	Dar la razón por la que se hacen los cambios. Ejemplo de razones: • Para arreglar el defecto *#xyz* • Para actualizar el diseño para un cambio de código actualizado • Para cambiar el SRS en respuesta a una solicitud del cliente aprobada. Estime los beneficios de cambio. Ejemplo de beneficios: • Habilite función *abc* para trabajar correctamente. • Mejore la ejecución por *qr%* Estime el impacto de cambio • Las horas de ingeniería que se requirieron para hacer el cambio. • Calendario de tiempo estimado para hacer el cambio • Otros impactos tales como otros cambios afectados Cambie la descripción. Aquí, se da una descripción del cambio. • 13 LOC cambiadas • 3 líneas cambiadas en el SRS y 11 líneas cambiadas en el SDS
Aprobaciones	Estos espacios son completados por las partes indicadas cuando ellos aprueban la submisión.

Anexo 11: Formulario SUMQ (Plan de Calidad).
(Tomado de: Humphrey, W.S., *Introduction to the Team Software Process*. 1999.)

Nombre: Fecha:
Equipo: Ejemplo de Datos del Equipo B Instructor:
Parte/Nivel: Cambio Contador/Sistema Ciclo:

Resumen de tarifas	Plan	Real
LOC/hora	1.17	3.43
% Reusabilidad (% de LOC total)	0	0
Nuevo Reusado (% de N&C LOC)	0	0
Por ciento libre de defectos (PDF)		
En compilación	80	20
En chequeo de unidad	90	20
En construcción e integración	*95*	60
En chequeo del sistema	99	40
Defecto/página		
Inspección de requerimientos	1.2	0
Inspección de HLD	0.7	0
Defectos/KLOC		
Revisión DLD	20.2	55.2
Inspección DLD	6.1	15.6
Revisión de código	65.0	27.1
Compilación	15.1	28.1
Inspección de código	10.6	0
Chequeo de unidad	4.1	16.6
Construcción e integración	0.4	6.2
Chequeo del sistema	0.1	6.2
Desarrollo total	151.2	170.7
Proporción de Defectos		
Revisión/compilación de código	4.3	0.96
revisión/chequeo de unidad DLD	4.9	3.31
Proporciones de Tiempo de Desarrollo (%)		
Requerimientos Inspección/Requerimientos	25	9
HLD inspección/HLD	37	0
DLD/código	67	497
DLD revisión/DLD	67	26
Código revisión/código	83	91
A/FR	2_.23	0.89
Tarifas de Revisión		
DLD líneas/hora	17.1	30.0
Código LOC/hora	27.3	131.1
Tarifas de Inspección		
Requerimientos páginas/hora	0.63	3.93
HLD páginas/hora	0.98	Inf.
DLD líneas/hora	9.13	35.71

Código LOC/hora	82.00	362.64
Frecuencia de inyección de defectos (Defectos/HLD)	Plan	Real
Requerimientos	0.25	0.00
HLD	0.25	0.00
DLD	0.75	2.11
Código	2.06	5.69
Compilación	0.50	9.80
Chequeo de la unidad	0.00	0.32
Construcción e integración	0.00	0.00
Chequeo de unidad	0.00	0.07
Frecuencia de eliminación de defectos (Defectos/HLD)		
Inspección de requerimientos	0.70	0.00
Inspección HLD	0.64	Inf.
Revisión DLD	1.03	5.03
Inspección DLD	0.17	1.69
Revisión de código	1.78	3.55
Compilación	3.10	26.47
Inspección de código	0.87	0.00
Chequeo de la unidad	0.22	0.85
Construcción e integración	0.02	3.87
Chequeo del sistema	0.00	0.40
Rendimientos de la fase		
Inspección de requerimientos	70	Inf.
Inspección HLD	70	Inf.
Revisión DLD	70	58.9
Desarrollo de la prueba		
Inspección DLD	70	40.5
Revisión de código	70	40.6
Compilación	50	56.3
Inspección de código	70	0.0
Chequeo de la unidad	90	59.3
Construcción e integración	80	54.5
Chequeo del sistema	80	100.0
Rendimientos del proceso		
% antes de compilar	81.3	74.1
% antes del chequeo de la unidad	97.0	86.6
% antes de la construcción e integración	99.7	93.3
% antes del chequeo al sistema	99.9	96.9
% antes de la entrega del sistema		

Instrucciones para llenar formulario SUMQ

Propósito	Este formulario retiene los datos de calidad planeados y reales para partes o ensambles
General	• Donde sea posible establecer objetivos basados en los datos históricos. • Donde los datos no estén disponibles usar la QUAL estándar para guiarse. • Antes de hacer el plan de calidad, se debe tener completo el formulario SUMP con los datos de tamaño y tiempo de desarrollo de la fase en proceso.
Fabricación del plan de calidad	Para hacer el plan de calidad , se debe hacer lo siguiente: • Estimar los defectos incluidos en cada fase (use los datos planeados y el QUAL estándar para los defectos incluidos por hora y para las horas gastadas por fase). • Estimar el rendimiento de cada fase de defectos eliminados (QUAL estándar). • Los defectos eliminados en cada fase son estimados: el número de defectos de la fase de entrada, tiempo de rendimiento estimado para la fase dividido por 100. • Examine los valores de defectos / KLOC por la racionalidad. • Si los valores de defectos / KLOC no son razonables, ajustar los tiempos de la fase, las proporciones de defectos incluidos, o los rendimientos (use QUAL estándar como guía). • Cuando los números sean razonables, el plan de calidad estará completo.
Registro de datos de calidad reales	Para completar el plan de calidad con los valores reales, entrar los siguientes datos: • Registra el tiempo de desarrollo en el registro de tiempo y resumir en el SUMP. • Registrar los defectos encontrados en el registro de defectos y resumir en el SUMP. • Entrar el tamaño de cada producto producido y resumir en el SUMP Con los datos de SUMP completos, complete el formulario SUMQ con la herramienta TSP.
Herramienta TSPi	• Si usa la herramienta, completará todos los cálculos de SUMQ. • Sin la herramienta tendría que hacer los cálculos SUMQ a medida que complete los pasos anteriores. • Loa cálculos de calidad se hacen siguiendo las

	instrucciones de abajo :
Encabezamiento	• Introducir nombre, fecha, equipo, y nombre del instructor. • Nombre de la parte o ensamble y su nivel. • Introducir número del ciclo.
Índice de resumen	• LOC/ horas: LOC nuevo y cambiado dividido por las horas de desarrollo totales. • % de reuso: Porcentaje de LOC totales que fueron reusados. • % de nuevo reuso: Porcentaje de LOC nueva y cambiada que fueron insertadas en la librería de reuso.
Por ciento libre de defectos (PDF)	• PDF se refiere al porcentaje de componentes del programa que no tienen en la fase de desarrollo y en la fase de prueba. • Así, si 3 de los 10 componentes del programa no tienen defectos en la compilación, el programa podría tener un 30% de PDF. • Fundamentar el plan de valores PDF en el QUAL estándar.
Defectos /páginas y Defectos /KLOC	• Fije los valores del plan de defectos/ páginas y de defectos / KLOC durante la planificación. • Los defectos/ páginas son calculados como (No de defectos/ No de páginas). • Los defectos / KLOC son calculados como (1000*Nn de defectos/ N&C loc).
Índice de defectos	• Estos son los índices del número de defectos encontrados en varias fases. • Sin embargo el índice de (revisión del código) compilación es el índice de defectos encontrados en la revisión del código. • Estos índices pueden también ser calculados a partir de los valores de defectos/ KLOC. • Cuando el denominador del valor de la fase es 0, entre inf.

Índice del tiempo de desarrollo (%)	• Estos son los índices de tiempo gastado en cada fase de desarrollo. • Sin embargo el índice de DLD/ compilación es el índice del tiempo gastado en el diseño del detalle por el tiempo gastado en la codificación del programa. • Calcular los índices reales y planeados desde los datos de SUMP • Cuando el denominador del valor de la fase es 0, entre inf.
A/ FR	• A/ FR es calculado como el índice de apreciación del tiempo de fallo. • Tiempo de apreciación es el tiempo gastado en la revisión e inspección de programas. • Tiempo de fallo es el tiempo gastado en la compilación y prueba de programas. • Para calcular A/FR, dividir la revisión del diseño del detalle total, revisión del código, y tiempo de inspección por los tiempo de compilación y prueba de unidad total.

Anexo 12: Formulario SUMP (Resumen del Plan).
(Tomado de: Humphrey, W.S., *Introduction to the Team Software Process*. 1999.)

Nombre		Fecha	
Equipo	Datos ejemplo del equipo B	Instructor	
Parte/Nivel	Cambiar Contador/Sistema	Ciclo	

Tamaño del producto		Plan	Real
Páginas requeridas (SRS)		5	11
Otras páginas de texto			
Páginas de alto nivel de diseño (SDS)		10	18
Líneas de diseño detalladas		137	316
LOC base (B) (medido)		- --	--
LOC borrado (D)			
		(Estimado)	(Contado)
LOC modificado (M)			
		(Estimado)	(Contado)
LOC Añadido (A)		410	961
		(N-M)	(T-B+D-R)
LOC Reusado (R)			
		(Estimado)	(Contado)
LOC Total Nuevo y Modificado (N)		410	961
		(Estimado)	(A+M)
LOC Total (T)		410	961
		(N+B-M-D+R)	(Medido)

LOC Total Nuevo Reusado			
LOC Objeto Estimado (E)			
Intervalo de mayor Predicción (70 %)			
Intervalo de menor Predicción (70 %)			
Tiempo en fase (horas)	Plan	Real	Real %
Manejo y misceláneas	87.5	61.5	21.95
Iniciación	11.8	12.3	4.39
Planeamiento y estrategias	46.0	23.2	8.29
Requerimientos	32.0	32.6	11.64
Plan de prueba del sistema			
Inspección de Requerimientos	8.0	2.8	1.00
Diseño a alto nivel	27.5	26.0	9.29
Plan de prueba de integración	4.0	4.0	1.43
Inspección de diseño a alto nivel	10.2		
Implementación de la planificación			
Diseño detallado	12.0	40.3	14.39
Revisión del diseño detallado	8.0	10.6	3.79
Desarrollo de la prueba	1.5		
Inspección del diseño detallado	15.0	8.9	3.18
Código	18.0	8.0	2.86
Revisión de código	15.0	7.3	2.61
Compilación	2.0	1.0	0.36
Tiempo en fase (horas)	Plan	Real	Real
Inspección del código	5.0	2.7	0.96
Chequeo de la unidad	7.5	18.7	6.68
Construcción e integración	6.0	1.5	0.54
Chequeo del sistema	12.0	15.0	5.36
Documentación	10.0	1.6	0.57
Después de terminado	10.0	2.0	0.71
Total	349.0	280.0	
Tiempo Total UPI (70 %)			
Tiempo Total LPI (70 %)			
Defectos inyectados	Plan	Actual	Actual %
Planeamiento y estrategia			
Requerimientos	8		
Plan de chequeo del sistema			
Inspección de requerimientos			
Diseño de alto nivel	7		
Plan de chequeo de integración			
Inspección del diseño a alto nivel			
Diseño detallado	9	85	51.81
Revisión del diseño detallado		9	5.49
Desarrollo de la prueba			
Inspección del diseño detallado			
Código	37	46	28.04
Revisión de código		7	4.29

Compilación	1	10	6.10
Inspección de código			
Chequeo de la unit		6	3.66
Construcción e integración			
Chequeo del sistema		1	0.61
Desarrollo total	62	164	
Defectos Eliminados	Plan	Actual	Actual %
Planeamiento y estrategia			
Requerimientos			
Plan de chequeo del sistema			
Inspección de requerimientos	5.60		
Diseño a alto nivel			
Plan de chequeo de integración			
Inspección de diseño a alto nivel	6.58		
Diseño detallado		4	2.44
Revisión de diseño detallado	8.27	53	32.31
Desarrollo de la prueba			
Inspección del diseño detallado	2.48	15	9.15
Código		11	6.71
Revisión de código	26.65	26	15.85
Compilación	6.21	27	16.46
Inspección de código	4.35		
Chequeo de unidad	1.68	16	9.76
Construcción e integración	0.15	6	3.66
Chequeo de sistema	0.03	6	3.66
Desarrollo total	62.00	164	

Instrucciones para llenar el Formulario SUMP

Propósito	Este formulario sostiene el plan y los datos actuales para los ensamblajes del programa.
General	• Un ensamblaje podría ser un sistema con productos múltiples, un producto con componentes múltiples, o un componente con módulos múltiples. • Una parte podría ser un objeto, módulo, componente, o producto. • Nota: los ensamblajes a bajo-nivel o módulos típicamente no tienen ningún dato del sistema-nivelado, así como los requerimientos, el diseño o prueba del sistema.
Usando la herramienta TSPi	Cuando está usando el TSPi, los valores del plan se generan automáticamente. • El tiempo y el tamaño de los datos son procesados desde los formularios TASK y SUMS. • Los valores de defectos se generan automáticamente durante el proceso de planeo de la calidad (SUMQ). Los valores reales también se generan automáticamente por la herramienta TSPi.

	• Los valores de tamaño y tiempo vienen de los formularios LOGT, TASK y SUMS. • Los datos defectuosos vienen de los formularios LOGD. Cuando no está usando el TSPi, siga las instrucciones debajo.
Encabezamiento	• Entre su nombre, fecha, nombre del equipo, y nombre del instructor. • Nombre el ensamblaje y su nivel. • Entre el número de ciclo.
Columnas	• Plan: Esta columna sostiene la parte o datos del plan de ensamblaje. • Actual: Para los ensamblajes, esta columna sostiene la suma de los datos reales para las partes del ensamblaje (al nivel más bajo, los módulos).
Tamaño del producto	• Para el texto y diseños, entre sólo el nuevo y modificado tamaño de los datos. • Para partes del programa o ensamblajes, entre todos los datos de LOC indicados. • Obtenga los datos del formulario SUMS.
Tiempo en fase	• Entre el tiempo real y estimado por fase. • Al nivel más bajo, obtenga estos datos del formulario TASK. • Para los ensamblajes de alto nivel, obtenga los datos de tiempo de cada parte de nivel de los totales en el formulario SUMT y los datos del nivel de ensamblaje del nivel de ensamblaje del formulario TASK. • Por ejemplo, el tiempo HLD vendría del formulario TASK y el módulo total del tiempo de chequeo de la unidad vendría del formulario SUMT. • Actual %: Entre el por ciento del tiempo de desarrollo real por fase.
Defectos inyectados	• Entre los defectos inyectados estimados y reales por fase. • Entre el defecto estimado mientras se esté produciendo el plan de calidad. • Para los módulos, obtenga los datos reales del LOGD para esos módulos. • Para los ensamblajes, reciba los datos de defectos módulo-nivel de los totales del formulario SUMDI y los datos del nivel de ensamblaje del formulario de ensamblaje LOGD. • Por ejemplo, los defectos de HLD vendrían del formulario de ensamblaje LODG y los defectos de código del módulo del total vendría del formulario SUMDI. • Actual %: Entre el por ciento de los defectos reales inyectado por la fase.
Defectos Eliminados	• Entre defectos estimados y reales eliminados por la fase. • Entre el defecto estimado mientras produce el plan de calidad. • Para los módulos, obtenga los datos reales del LOGD para esos módulos. • Para los ensamblajes, obtenga los datos de defectos módulo-nivel de los totales del formulario SUMDR y datos del nivel-ensamblaje del formulario LOGD. • Por ejemplo, la revisión de defectos HLD vendría del ensamblaje del formulario LOGD y el módulo total de la revisión de defectos de código vendría del formulario SUMDR. • Actual %: Entre el por ciento de los defectos reales quitado por la fase.

Anexo 13: Formulario INS (Reporte de Inspecciones)
(Tomado de: Humphrey, W.S., *Introduction to the Team Software Process*. 1999.)

Nombre: _____ Fecha: _____
Equipo: _____ Instructor: _____
Parte/Nivel: _____ Ciclo: _____
Moderador: _____ Propietario: _____

Datos del ingeniero

Nombre	Defectos		Datos de preparación			Campo Est.
	Mayor	Menor				
Totales:						

Datos de defecto

No.	Descripción del defecto	Defectos		Ingenieros(encontrando los mayores defectos)					
		Mayor	Menor					A	B
Totales:									
Defectos únicos									

Resumen de la Inspección
Total del defecto por A: _____
Total de defectos (AB/C) _____
Tiempo de reunión: _____

Tamaño del Producto:
Total de defectos por B: _____
Números encontrados (A+B-C):_____
Horas de inspección total: _____

Métrica de tamaño
C (# común): _____
Número pendiente: _____
Índice global: _____

Instrucciones para llenar el formulario INS

Propósito	• Recolectar y analizar los datos de inspección
General	• Estos datos necesitan recogerse durante la inspección porque generalmente no pueden obtenerse después • Registrar los datos de preparación al inicio de la reunión de la inspección • Complete el formulario al final de la reunión de inspección • Es útil para tener los números de páginas y de líneas en el texto impreso del producto
Encabezamiento	• Entre su nombre, nombre del equipo, nombre del instructor y la fecha
Datos del ingeniero	• Para cada inspector entre el nombre y el tiempo de preparación y además: • El número de defectos mayores y menores que el inspector encontró • Las COC, líneas y páginas que el inspector inspeccionó (los inspectores pueden concentrarse en secciones del programa) • El índice de preparación en LOC, líneas de seudocódigo o páginas por hora • Entre el tiempo total de preparación, los defectos totales mayores y menores y el índice global • El moderador calcula los rendimientos totales y de los ingenieros al final de la reunión
Datos del defecto: No.	• Entre un número por cada defecto encontrado en la inspección • Es generalmente más conveniente usar la línea del documento y el número de página
Datos del defecto: Descripción del defecto	• Describe cada defecto y chequea si este es menor o mayor
Datos del defecto: Ingenieros	• En la primera fila debajo del encabezamiento Ingenieros entre las iniciales de cada ingeniero que está participando de la inspección • Para los defectos mayores regístrelos en la columna de cada ingeniero que encontró ese defecto durante la inspección
Resumen	Al final de la inspección complete los datos de resumen • Para las inspecciones a los requerimientos o al diseño de alto nivel, entre las páginas de texto • Para las inspecciones a los diseños detallados entre las LOC o líneas de seudocódigo • Para el código fuente, entre LOC del código fuente Entre el tiempo de reunión, las horas totales de la inspección y el índice global de inspección
Resumen: Estimados Defectos persistentes	• Después que todos los defectos fueron entrados, cuente los defectos mayores que cada ingeniero encontró que otro ingeniero no encontró (defectos únicos) • Identifique el ingeniero que encontró más defectos únicos • Chequee cada defecto que cada ingeniero encontró en la columna A • En la columna B, registre todos los defectos encontrados por los otros ingenieros • Cuente los defectos comunes (C para los comunes) entre A y B

	• Los defectos totales estimados para el producto son AB/C • Redondeo los resultados fraccionarios al entero más cercano • El número encontrado en la inspección es A+B-C • El número pendiente es el total menos los encontrados (AB/C)-(A+B-C) • Esta estimación de defecto es confiable solo cuando ambos números A y B son mayores que 4 y A-C y B-C son mayores que 1 • Incluso con este criterio, el error del defecto total puede ser del 10% o más • Cuando el rendimiento de uno o más ingenieros es del 70% o mayor, los rendimientos son generalmente bastante confiables • Si A=B=C ha encontrado todos los defectos • Si varios ingenieros encuentran un mismo gran número de defectos únicos, repita estos cálculos, usando cada uno de estos ingenieros como A y use los números más largos resultantes como el total de defectos estimados

Anexo 14: Formulario LOGD (Registro de defectos).

(Tomado de: Humphrey, W.S., Introduction to the Team Software Process. 1999.)

Tipos de Defectos

10 Documentación
20 Sintaxis
30 Construcción, Empaquetamiento
40 Asignación
50 Interface
60 Comprobación
70 Datos
80 Función
90 Sistema
100 Ambiente

Nombre

Nombre: _____ Fecha: _____
Equipo: _____ Instructor: _____
Ensamblaje: _____ Ciclo: _____

Fecha	Número	Tipo	Inyectado	Eliminado	Tiempo de arreglo	Defecto arreglado

Descripción: _____

_____

Instrucciones para llenar el Formulario LOGD

Propósito	• Use este formulario para retener los datos sobre los defectos encontrados y rectificados.
General	• Mantener un registro separado para cada componente del programa. • Registrar cada defecto por separado y completamente. • Si necesita espacio adicional, use otra copia del formulario.
Encabezamiento	• Entre su nombre, fecha, nombre del equipo y nombre del instructor. • Nombre el ensamblaje. • Entre el número del ciclo.
Fecha	• Entre la fecha cuando encontró el defecto.
Número	• Entre el número del defecto. • Para cada componente del programa (o módulo), use un número secuencial comenzando con 1 (o 001, etc.)
Tipo	• Entre el tipo del defecto de la lista de los tipos de defectos resumida en la esquina superior derecha del formulario. • Ver el Apéndice G para la especificación del tipo de defecto. • Use su mejor criterio en la selección de cuál tipo de defecto aplicar.
Inyectado	• Entre la fase en la que fue inyectado este defecto. • Use su mejor criterio.
Eliminado	• Entre la fase durante la cual arregló el defecto. • Esta es por lo general la fase en la que encontró el defecto.
Tiempo de Arreglo	• Entre el tiempo que le tomó arreglar el defecto. • Este tiempo puede ser determinado por un cronómetro o a criterio.
Defecto Arreglado	• Si usted o alguien inyectó este defecto mientras estaba arreglando otro, registre el número del defecto arreglado inadecuadamente. • Si no puede identificar el número del defecto, entre una X en el cuadro de Defecto Arreglado.
Descripción	• Escriba una breve descripción que sea bastante clara para recordarte el error y por qué lo cometiste. • Si el defecto fue inyectado en un ciclo anterior, da el número del ciclo y coloca un * por la fase en la que fue inyectado (como programación* o diseño*)

Anexo 15: Formulario de LOGPrueba (Registro de prueba)
(Tomado de: Humphrey, W.S., Introduction to the Team Software Process. 1999.)

Nombre _____ Fecha _____
Equipo _____ Instructor _____
Parte/Nivel _____ Ciclo _____

Fecha	Prueba / Fase	Producto	Comienzo	Parada	Tiempo de interrupción	Tiempo transcurrido	Problemas	Comentarios

Instrucciones para llenar el Formulario LOGPRUEBA

Propósito	• El registro de prueba contiene un resumen de las pruebas y los resultados obtenidos. • Con los datos del registro de prueba, se puede determinar rápidamente cuáles pruebas se ejecutaron y cuáles encontraron la mayoría de los defectos. • También se puede determinar el tiempo de la prueba y los defectos encontrados por hora de prueba. • Los datos de este registro pueden ayudar a seleccionar la estrategia más eficiente para programas modificados con la prueba de la regresión.
General	• Usar este registro para rastrear la unidad, integración y prueba del sistema. • Una prueba (test run) es una ejecución ininterrumpida de la prueba. • Cuando una prueba se detiene, para arreglar el programa que está siendo probado o para cambiar los materiales de prueba, esa prueba se concluyó. • Cuando la prueba continua después del cambio, esta se registra como una nueva prueba.
Encabezamiento	• Entre su nombre, fecha, nombre del equipo y nombre del instructor. • Nombre el ensamblaje y su nivel. • Entre el número de ciclo.
Fecha	• Entre la fecha en que se ejecutó cada prueba.
Prueba/Fase	• Identifique cada prueba y fase. • Use el mismo nombre o número como que en la prueba del plan.
Producto	• Nombre el producto que se está probando.
Inicio	• Entre el tiempo cuando se comenzó la prueba.
Parada	• Entre el tiempo cuando se detuvo la prueba, porque finalizó con éxito o porque se suspendió por alguna razón relacionada con la prueba o el programa.
Tiempo de interrupción	• Si la prueba se suspendió temporalmente por una razón no relacionada con el programa, esto es un tiempo de interrupción. • Por ejemplo, el probador tomó un descanso.
Delta tiempo	• El tiempo entre el que comenzó y se detuvo la prueba, menos el tiempo de interrupción.
Problemas	• El número de informes de problemas de prueba. • Si se decide que uno o más problemas son defectos, anotarlos debajo de los comentarios.
Comentarios	Anotar brevemente los resultados de la prueba. Por ejemplo: • La prueba finalizó con éxito. • La prueba se completó, pero el resultado fue incorrecto. • La prueba falló con 3 problemas, 1 defecto. • Donde sea posible dar información adicional sobre los resultados de la prueba, configuración de la prueba o algo más que permita reproducir los resultados de la prueba.

Anexo 16: Formulario PIP (Mejoras del proceso).
(Tomado de: Humphrey, W.S., Introduction to the Team Software Process. 1999.)

Nombre _____ Fecha _____
Equipo _____ Instructor _____
Parte/ nivel _____ Ciclo _____
Proceso _____ Fase _____

Número PIP _____ Prioridad _____

Descripción del problema (Breve descripción del problema encontrado y su impacto)

_____
_____

Descripción de la propuesta (Describe los cambios sugeridos lo más completo posible, incluyendo formularios afectados, escrituras, etc.)

_____
_____

Someta el formulario PIP completo a la calidad / Administre el proceso y mantenga una copia.

_____
No escribir debajo de esta línea

de control de PIP _____ Organización _____
Recibido _____ Confirmación de recepción _____
Actualizado _____ Cerrado _____
Cambios _____
_____

Instrucciones para llenar el formulario PIP (propuesta mejorada del proceso TSP)

Propósito	• Registrar problemas del proceso e ideas mejoradas. • Proporcionar un registro ordenado de las ideas mejoradas del proceso para un posterior mejoramiento.
General	Usar el formulario de PIP para: • Registrar ideas mejoradas del proceso cuando ocurran. • Establecer prioridades para sus planes mejorados • Describir las lecciones aprendidas y las condiciones inusuales. Tener a mano los formularios PIP mientras se esté usando TSP. • Registrar problemas del proceso incluso sin soluciones propuestas. • Someter los PIPs para el uso en mejoras de procesos.
Encabezamiento	• Entrar nombre, fecha , equipo y nombre del instructor • Nombre de la parte o ensamblaje y su nivel. • Entrar número del ciclo. • Entrar proceso y fases del proceso en el lugar apropiado.
Número PIP	• Úselo para identificación.

Prioridad	• Indicar si la prioridad de PIP es urgente, normal o de rutina. • De la razón de prioridad describiendo el problema.
Descripción del problema	Describe el problema lo más claro posible • Dificultad encontrada • Impacto en el producto, el proceso, y usted. Incluya los problemas relacionados si son relevantes.
Descripción de la propuesta	• Describa su propuesta de proceso mejorada lo más explícito posible • Donde sea posible, haga referencia de elementos afectados en el proceso específico y las palabras o entradas a ser cambiadas • Donde se listan varios problemas, indique en el problema la propuesta con la que se relaciona.
Someter PIP completo	Después de completar el PIP • Mantenga una copia • Someta una copia de calidad / administre proceso
Calidad / Administración de proceso	Use el espacio a fondo del formulario PIP para rastrear el estado de PIP Revise todos los PIPs y: • Confirmación de recibo • Grupos duplicados y relacionados con PIPs • Identificar PIPs de alta prioridad. Someta copias de PIPs al instructor.

Anexo 17: Formulario PEER (Evaluación en pares)
(Tomado de: Humphrey, W.S., *Introduction to the Team Software Process*. 1999.)

Nombre_____ Equipo_____
Instructor_____
Fecha_____ Ciclo No._____ Semana No._____

Para cada rol, evalúe el trabajo requerido y la dificultad relativa en % durante este ciclo:

Rol	Trabajo requerido	Dificultad del rol
Líder del equipo		
Director de desarrollo		
Director de planificación		
Director de proceso/calidad		
Director de soporte		
Contribución total (100%)		

Valoración del equipo contra cada criterio. De 1 (inadecuado) a 5 (superior).

Espíritu del equipo	1	2	3	4	5
Efectividad total	1	2	3	4	5
Experiencia provechosa	1	2	3	4	5
Productividad del equipo	1	2	3	4	5
Calidad del proceso	1	2	3	4	5
Calidad del producto	1	2	3	4	5

Valoración del rol para una completa contribución. De 1 (inadecuado) a 5 (superior).

Espíritu del equipo	1	2	3	4	5
Efectividad total	1	2	3	4	5
Experiencia provechosa	1	2	3	4	5
Productividad del equipo	1	2	3	4	5
Calidad del proceso	1	2	3	4	5
Calidad del producto	1	2	3	4	5

Valoración de cada rol para la asistencia y apoyo. De 1 (inadecuado) a 5 (superior).

Espíritu del equipo	1	2	3	4	5
Efectividad total	1	2	3	4	5
Experiencia provechosa	1	2	3	4	5
Productividad del equipo	1	2	3	4	5
Calidad del proceso	1	2	3	4	5
Calidad del producto	1	2	3	4	5

Valoración de cada rol según su buen funcionamiento. De 1 (inadecuado) a 5 (superior).

Espíritu del equipo	1	2	3	4	5
Efectividad total	1	2	3	4	5
Experiencia provechosa	1	2	3	4	5
Productividad del equipo	1	2	3	4	5
Calidad del proceso	1	2	3	4	5
Calidad del producto	1	2	3	4	5

Instrucciones para llenar el Formulario PEER

Propósito	• Este formulario mantiene el equipo y sus evaluaciones en pares.
General	• Los equipos llenan el formulario PEER en el postmortem de cada ciclo • Completar cada entrada • Llenar tu propia evaluación tan bien como la del resto • Al reverso del formulario o en hojas adjuntas adiciones comentarios o sugerencias que desee.
Encabezamiento	• Entre su nombre y el del instructor. • Entre el nombre del equipo, No. del ciclo, fecha y # de la semana.
Trabajo requerido y dificultad del rol	Para cada rol: • Estimar la cantidad relativa de trabajo que sientes que fue requerido por cada rol durante el ciclo de desarrollo • También valorar el rol en términos de dificultad total Para realizar esta evaluación • Proporcione % estimados respecto al total (100%) • Ejemplo: si siente que todos los roles fueron iguales en la misma medida, escriba: 20, 20, 20, 20 y 20. • Si alguno roles implican más trabajo y dificultad, podría escribir: 20, 30, 15, 20 y 15
Evaluación del equipo como un todo	Para una escala de 1 (inadecuado) a 5 (superior) evalúe al equipo en: • Espíritu del equipo • Efectividad total • Si este proyecto para usted representó una experiencia a recordar. • Productividad • Calidad del proceso y el producto
Contribución total del rol	• Evalúe cada rol por la contribución que siente que hizo el ingeniero en ese rol al proyecto durante el ciclo de desarrollo • Circule el # aplicable (1 para la menor contribución y 5 para la mayor) • La evaluación requiere no ser ralativa. Esto es, que todos podrían ser 5, todos 1 o cualquier combinación
Utilidad y soporte total del rol	• Evalúe cada rol por lo útil que siente que fue el ingeniero en ese rol al proyecto durante el ciclo de desarrollo •
Ejecución total del rol	• Evalúe cada rol según cuán bien usted sienta que fue ejecutado • Esta evaluación se basa en cuán bien usted sienta que las tareas y responsabilidades del rol fueron manejadas • Circule el # aplicable (1 para la menor contribución y 5 para la mayor) • La evaluación requiere no ser ralativa. Esto es, que todos podrían ser 5, todos 1 o cualquier combinación.

Anexo 18

Encuesta realizada a expertos en el tema.

Usted ha sido seleccionado, por su calificación científico_técnica y los resultados alcanzados en su labor profesional, como experto para evaluar los resultados de esta investigación. En este cuestionario debe precisar el peso (en un rango 1-5) que le concede a cada criterio de evaluación de acuerdo a su opinión. Para un mejor entendimiento de la propuesta, se han dividido las interrogantes en cuatro criterios generalizadores (Criterios del método científico, Criterios de Implantación, Criterios de generalización, Criterios de impacto).

Criterios a Evaluar	Peso
1. Criterios del método científico	
1.1 Nivel de Calidad de la Investigación	
1.2 Aportes Científicos Novedosos	
1.3 Novedad Científica de la Investigación	
2. Criterios de Implantación	
2.1 Necesidad de Uso de la estrategia.	
2.2 Satisfacción de las necesidades de la producción	
3. Criterios de generalización	
3.1 Nivel de comprensión	
3.2 Facilidades de uso	
3.3 Nivel de adaptación a diferentes entornos de producción de SW.	
4. Criterios de impacto	
4.1 Contribución al proceso de desarrollo de SW.	
4.3 Posibilidades de aplicación.	
4.4 Posibilidades de automatización.	

Anexo 19

Opiniones de los expertos.

Mabel Gerra Samuell

Considero como muy bueno el trabajo, debido a que cumple en primer lugar con los objetivos que se plateó el diplomante en la investigación, y que da solución a un problema que existe no solo en los proyectos productivos de la facultad sino en todos los de la Universidad.

La estrategia propuesta esta claramente explicada, lo cual le da facilidad de uso y aplicación por los equipos de desarrollo, además esta organizada por fases y se identifica en cada una quienes son los responsables de generar cada uno de los formularios y plantillas necesarios, algo que da organización a la hora del trabajo.

Opino que esta estrategia esta muy bien elaborada y que está basada en una investigación válida y profunda acerca del tema en cuestión, y que su aplicación mejorará de manera importante el desarrollo de software en los proyectos de la facultad. Propongo además que su uso una vez probado en nuestros proyectos, sea extendido a otros proyectos de la universidad.

Gretchen Guillermo Hernández

Considero que la tesis está muy buena y que abarca uno de los problemas principales del proceso de desarrollo de software, se debe aplicar en el Polo lo más pronto posible para poder lograr una mejor organización y un mejor trabajo en el desarrollo de los proyectos productivos del Polo y de la Facultad de forma general.

Armando Ortiz Cabrera

La estrategia que se presenta en este trabajo sobre la aplicación de técnicas de trabajo en equipo (TSP). Presenta un horizonte my bueno y aplicable a los proyecto de área de desarrollo. Presenta varios puntos claves o críticos que están contemplados a lo largo del ciclo de desarrollo de software que si se violan o no se toman con seriedad pueden provocar que el proyecto fracase o no tenga la calidad esperada para el cliente.

Mi criterio es que se tome en cuenta y se pongan en aplicación esta estrategia que a un menor o largo plazo los proyectos de la facultad 9 podrán ser cada vez mejores y así poder fidelizar al cliente y tener soluciones más genéricas que puedan ser aplicables a distintas entidades o empresas. Desde el punto de vista teórico está muy bien fundamentado y avalado por expertos en desarrollo de software y trabajo en equipo.

Alfonso Chaveco Laurencio

La estrategia de TSP que se propone para el Polo de Petrosoft está muy bien pensada, de esta forma se logra que todos los equipos y los integrantes de cada proyecto hablen el mismo idioma, que exista un mayor entendimiento del trabajo en el equipo. Se logra un producto con mayor calidad, una planificación del proyecto más exacta y con mejor calidad. Permite un mayor aprovechamiento del tiempo de trabajo y de los recursos que se disponen, permite hacer una distribución del trabajo por roles lo cual conlleva a un mejor producto y a una entrega en tiempo, se eliminan las entregas tardías de los proyectos.

Ramses Ibarrola Suárez

Considero que la investigación tiene un alto nivel científico, la estrategia que se presenta en este trabajo sobre la aplicación de técnicas de trabajo en equipo (TSP) resulta de gran importancia aplicada en proyectos productivos. Este trabajo puede ayudar en gran medida a mejorar la calidad de los productos, siempre y cuando el equipo de trabajo cumpla con la estrategia planteada en el mismo.

Mi criterio sobre la investigación es que está apta para comenzar a ser aplicada en los proyectos de la UCI luego que haya sido verificada por los expertos correspondientes, ya que su autora ha mostrado en ella un alto grado de confiabilidad y calidad.

Glosario de Términos

Software: Conjunto de programas elaborados por el hombre, que controlan la actuación del computador, haciendo que éste siga en sus acciones una serie de esquemas lógicos predeterminados.

Producto: Conjunto de atribuciones tangibles e intangibles que incluye el empaque, color, precio, prestigio del fabricante, prestigio del detallista y servicios que prestan este y el fabricante.

Calidad*:* Un proceso de mejoramiento continuo, en donde todas las áreas de la empresa participan activamente en el desarrollo de productos y servicios, que satisfagan las necesidades del cliente, logrando con ello mayor productividad.

Procesos: Conjunto de actividades o acciones interrelacionadas entre sí que, a partir de una o varias entradas de información, materiales o de salidas de otros procesos, dan lugar a una o varias salidas también de materiales (productos) o información con un valor añadido.

Estándares: Normas de desempeño definidas para una actividad, un proceso, un producto o un servicio.

Equipo: Un equipo consiste en al menos dos personas que trabajan juntos hacia una meta, objetivo o misión común, donde cada persona tiene asignados roles o funciones específicas y además completar la misión requiere alguna forma de dependencia entre los miembros del grupo.

ERS (Especificación de Requerimientos de Software): Es un documento donde el equipo describe las funciones que se planean desarrollar y cómo se pretende generar el producto.

EDS (Especificación de Diseño de Software): Es un documento donde se describe la lógica y la estructura del programa principal del sistema para asegurar que las funciones de los componentes y las interfaces estén completamente identificadas.

PPI (Plan de Pruebas de integración): Es un documento que va a contener todas las interfaces a las que se le han aplicado las pruebas de integración.

Nivel: Período de tiempo entre dos hitos principales de un proceso de desarrollo.

Proyecto: Conjunto de actividades interrelacionadas, con un inicio y una finalización definida, que utiliza recursos limitados para lograr un objetivo deseado.

PROBE (PROxy Based Estimating): Estimación basada en la evaluación. Es un método creado con el fin de realizar estimaciones tanto del tamaño del programa como de los recursos del mismo.