Universidad de las Ciencias Informáticas
Facultad 4

Título: **Propuesta de medición a los proyectos de la Facultad 4 basados en PSP y TSP.**

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Autores: Dunier Isanto Fernández Garcés
Orliandi López Labañino

Tutora: Ing. Dayami Rodríguez Brito

Ciudad de La Habana, Junio del 2008

"Año 50 de la Revolución"

DECLARACIÓN DE AUTORÍA

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

Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

_____
Dunier Isanto Fernández Garcés
Autor

_____
Orliandi López Labañino
Autor

_____
Ing. Dayami Rodríguez Brito
Tutora

DATOS DE CONTACTO

Ing. Dayami Rodríguez Brito.
Profesora Instructora recién graduada en adiestramiento.

Graduado en el 2007 de Ingeniera en Ciencias Informáticas en la Universidad de las Ciencias Informáticas (UCI).

Ha impartido las asignaturas de Ingeniería de Software I y II, Gestión de Software, Teleinformática I y II, Investigación de Operaciones y Probabilidades y Estadísticas. Algunas de estas asignaturas como alumna ayudante y las dos últimas como profesora en la facultad 4.

Ha trabajado en proyectos nacionales e internacionales en el perfil de calidad.

Especialista del grupo de Métricas de la Dirección de Calidad de Software de la Infraestructura Productiva de la UCI.

Tiene varias publicaciones relacionadas con el tema de las métricas de software y ha participado en varios eventos como: UCIENCIAS, FORDES.

Correo electrónico: drodriguezb@uci.cu

AGRADECIMIENTOS

Quisiera agradecer a mis padres por ser los mejores del mundo, por su preocupación por mí en cada momento, por su sacrificio, su apoyo y comprensión durante este tiempo.

A mis hermanos que marcaron el camino a seguir, por su confianza y consejos, gracias…

A mis abuelitos que tanto quiero.

A mis cuñadas por sus pícaras conversaciones y divertidos momentos.

A toda mi familia en general, en especial a mi tía Antonia: ahora si soy un gigante.

A mi gran amigo Asdrúbal Antonio (el negro) por estos cinco años de amistad, por haber sido mi mayor crítico, por su ayuda incondicional.

A Indalesio (Indale), por sus consejos oportunos, por ser más que un amigo para mi y mis hermanos.

A mi compañero de tesis Dunier Isanto (Titi), porque sin su esfuerzo y determinación me hubiese resultado excesivamente imposible desarrollar este trabajo.

A nuestra tutora Dayami Rodríguez Brito, que hizo su mayor esfuerzo por ayudarnos.

A mis amistades por su dedicación, cariño y por su enorme apoyo emocional. A Maylin, Meylin, Betty, Brocard (el primo), Yeniset, Ania, Emilio, Amiris, Kenia, Yaidelin.

…y a todos los que no puse, pero que de una forma u otra han contribuido a mi formación profesional y personal, que no se me olvidan, a todos, muchas gracias.

Orliandi

A la Revolución por darme la oportunidad de estudiar en una universidad como esta.

A mis padres porque sin ellos hoy no estuviera aquí, por guiarme por el buen camino, por creer en mi.

A mis hermanos por su apoyo y su confianza en mí.

A mis amigos por ser durante estos cinco años mi familia dentro de la UCI especialmente a Liliam, Leysa, Liuba, Yilian, Rigo, Marta y Dalia.

A mi compañero de tesis Orliandi por su gran dedicación y las largas horas de trabajo en la realización de este trabajo

A nuestra tutora Dayami por su gran ayudad en la realización de este trabajo.

A los que de una forma u otra han colaborado en la realización de este trabajo

Dunier

DEDICATORIA

A mis padres por su sacrificio y dedicación

Dunier

A mis padres y hermanos…

Orliandi

RESUMEN

La planificación, en el mundo de la Industria del Software, se ha convertido en uno de los principales retos para la gestión de proyectos y una actividad fundamental para desarrollar software de alto rendimiento. Por su importancia, se requiere que el proceso de desarrollo tenga la calidad suficiente, como para garantizar la calidad del producto que se obtenga.

Se considera imprescindible para planificar la aplicación de buenos métodos de estimación, por la creciente influencia que ejercen en el control preciso, predecible y repetido sobre los procesos de producción y los productos de software.

En la Universidad de las Ciencias Informáticas (UCI), las mediciones de software no se están llevando con el rigor y compromiso adecuado, entre varios motivos porque aún no se ha creado una cultura de medición desde un nivel personal y como consecuencia hasta una escala mayor; que sería la institución, lo que conlleva a planificaciones irreales, poca información histórica para realizar estimaciones que permitan hacer planificaciones reales, para que no surjan atrasos en la entrega del producto y este que de con la calidad adecuada.

El objetivo principal del Proceso Personal de Software (PSP) y el Proceso de Software en Equipo (TSP), es crear las disciplinas en el proceso de desarrollo de software. El presente trabajo se centra en la realización de una propuesta de aplicación de dichos modelos, con el objetivo de realizar mediciones para obtener datos reales que permitan hacer mejores estimaciones, que admitan la toma de decisiones necesarias para obtener un producto con mayor calidad.

Se realiza además un estudio de los niveles de PSP y TSP y las métricas que se utilizan en el proceso de desarrollo de software para PSP y TSP, así como un análisis de la aplicación de estos niveles en los proyectos productivos de la facultad 4. Y como resultado se proponen una serie de tareas específicas para llevar a cabo, con algunas métricas asociadas, que contribuyen a elevar la calidad de software en la universidad.

PALABRAS CLAVE

Planificación, estimación, mediciones, método de estimación, PSP, TSP.

ÍNDICE

DECLARACIÓN DE AUTORÍA I

DATOS DE CONTACTO II

AGRADECIMIENTOS III

DEDICATORIA V

RESUMEN VI

INTRODUCCIÓN 1

CAPÍTULO 1: MARCO TEÓRICO 4

1.1 Introducción. 4

1.2 Calidad del Software. 5

1.2.1 Control de la Calidad. 6
1.2.2 La Garantía de la Calidad. 6
1.2.4 ¿Cómo obtener un software con Calidad? 6
1.2.5 ¿Cómo controlar la Calidad del Software? 7

1.3 Planificación del proyecto. 8

1.4 Estimación del proyecto de software. 8

1.4.2 Estimación basada en el proceso. 9

1.5 Métodos de estimación. 9

1.5.1 Empíricos 9
1.5.2 Heurísticos 9
1.5.3 Juicio del Experto: 10
1.5.4 Puntos de Casos de Uso: 10
1.5.5 COCOMO 10
1.5.6 Puntos de Función 11

1.6 Métricas del software. 11

1.7 Estándares de calidad. 13

1.8 PSP. 14

1.8.1 Principios de PSP. 16
1.8.2 Niveles del PSP. 16

1.9 TSP. 18

1.10 TSPi. 19

Principios básicos del TSPi. 19
Diseño del TSPi. 20

1.11 Conclusiones. 21

CAPÍTULO 2: PROPUESTA DE APLICACIÓN 23

2.1 Introducción. 23

2.2 Proyectos de software desarrollados en la facultad 4. 23
2.2.1 Banco 23
2.2.2 Aduana: Gestión Integral de Recursos Humanos. 23
2.2.3 Aduana: Información Adelantada 24
2.2.4 Aduana: Gestión de los Procesos Aduanales 24
2.2.5 ResiSoft 24
2.2.6 SIGEP: Automatización de las Instalaciones Penitenciarias. 25
2.2.7 MINFAR 25
2.3 Análisis de los resultados de la encuesta. 25
2.4 Propuesta de Aplicación de TSP y PSP. 32
2.4.1 Registro de Tiempo. 33
2.4.2 Registro de Defectos. 34
2.4.3 Resumen del Plan de Proyecto. 34
2.4.4 Medición del Tamaño. 35
2.4.5 Estimación del Tamaño. 35
2.4.6 Lanzamiento del Proyecto. 36
2.4.7 Estrategia de Desarrollo. 36
2.4.8 Plan de Proyecto. 36
2.4.9 Posmorten. 37
2.4.10 Líder de Equipo. 37
2.4.11 Jefe de Desarrollo. 38
2.4.11 Jefe de Planificación. 38
2.4.13 Jefe de Proceso y la Calidad. 38
2.4.14 Jefe de Soporte. 38
2.4.15 Métricas. 39
2.5 Conclusiones. 43
CAPÍTULO 3: EVALUACIÓN TÉCNICA 45
3.1 Introducción. 45
3.1 Proceso de selección de expertos. 45
3.1.1 Selección de expertos que conforman el panel. 45
3.1.2 Confirmación de la participación de los expertos. 45
3.2 Elaboración del cuestionario. 46
3.3 Resultados de la evaluación. 46
3.3.1 Resultados obtenidos 48
3.4 Conclusiones. 50
CONCLUSIONES GENERALES 51
RECOMENDACIONES 52
REFERENCIAS BIBLIOGRÁFICAS 53
BIBLIOGRAFÍA 55
GLOSARIO DE TÉRMINOS 57

ANEXOS 59

Anexo #1: Estándar de Tipos de Defecto. 59

Anexo #2: Cuaderno Registro Defectos. 59

Anexo #3: Resumen del Plan de Proyecto. 62

Anexo #4: Cuaderno Registro Tiempos. 64

Anexo #5: Encuesta Realizada a los proyectos. 66

Anexo #6: Datos de los miembros del Panel de Expertos. 72

Anexo #7: Encuesta realizada al Panel de Expertos. 72

ÍNDICE DE TABLAS Y FIGURAS

Tabla 1: Roles de TSP por proyectos 31
Tabla 2: Línea Base de PSP0 y PSP0.1 32
Tabla 3: Métricas Bases. 39
Tabla 3: Métricas Derivadas 41
Tabla 4 Valores emitidos por los expertos. 47
Tabla 5 Valores de suma de rangos 48
Figura 1: Relación entre CMMI, TSP y PSP 15
Figura 2: La evolución de PSP. 17
Figura 3: Estructura del TSP 18
Figura 4: Estructura y flujo de TSPi. 21
Figura 5: Mediciones en los proyectos 26
Figura 6: Planificación de las Actividades 27
Figura 7: Gestión de la Panificación del Tiempo 28
Figura 8: Prueba y Reportes de Pruebas 28
Figura 9: Estimación y Medición del Tamaño 29
Figura 10: Plan de proyecto 30
Figura 11: Procesos de TST presentes en los proyectos 31

INTRODUCCIÓN

Hoy día la informática, como nueva alternativa en un mundo cada vez más desarrollado tecnológicamente, se ramifica en todos los sectores de la sociedad. Son muchas las instituciones que han apostado a la informática para minimizar los costos en sus procesos, brindar un mejor servicio o mejorar el desempeño.

Cuba, no está ajena en sus ansias de desarrollo por una economía fuerte, emprendedora y sostenida. Surge así la primera universidad al calor de la Batalla de Ideas, la Universidad de Ciencias Informáticas (UCI); con el objetivo de formar profesionales únicos es su tipo, vinculando siempre la docencia, producción e investigación.

La producción de Software constituye una gran fuente de ingreso para la universidad y para el país, y la satisfacción del cliente con un producto eficiente y la calidad requerida, representa uno de los principales objetivos de la institución.

Las organizaciones que emplean a los ingenieros preparados en el Proceso Personal de Software (PSP) y el Proceso de Software en Equipo (TSP), logran una reducción significativa en los errores que presenta el software, una reducción en el tiempo que toma desarrollar el software. Es más fácil y rápida la estimación del tiempo que se necesita para terminar el producto y en caso que sea necesario realizar algún cambio es fácil hacerlo, los costos de desarrollo se reducen en gran proporción, se tiene control total sobre el proceso de desarrollo y la productividad aumenta tanto personal como colaborativamente; por lo que la calidad del producto que se tiene al final es la más óptima.[1]

El PSP, muestra cómo los ingenieros:

- Gestionar la calidad de los proyectos.
- Mejorar la estimación y la planificación.
- Reducir los defectos de los productos.

El TSP, junto con el PSP, ayuda a la ingeniería de alto rendimiento a:

- Garantizar la calidad de los productos de software.
- Crear productos de software de seguridad.
- Mejorar la gestión de los procesos en una organización.

La gestión de un proyecto software comienza con un conjunto de actividades que globalmente se denomina planificación del proyecto. Antes de que el proyecto comience el gestor y el equipo de software deben realizar una estimación del trabajo a realizar y el tiempo que transcurrirá desde el comienzo hasta el final de su realización.[2]

La planificación de proyectos en una organización ayuda a mejorar la gestión del desarrollo de software, y esto traerá como consecuencia una disminución de los problemas y errores, favoreciendo las relaciones y comunicación entre las personas y grupos de organización, y de éstos con los clientes, lo que traerá consigo el aseguramiento de la calidad del software a lo largo del proceso de desarrollo.[3]

Aunque la estimación es más un arte que una ciencia, resulta una actividad importante que no debe llevarse a cabo de forma descuidada. Existen técnicas útiles para la estimación del tiempo y el esfuerzo. Las mediciones del proyecto y del proceso proporcionan una perspectiva histórica y potente introducción para generar estimaciones cuantitativas. La experiencia anterior puede ayudar en gran medida al desarrollo y revisión de las estimaciones, dado que la estimación es la base de las demás actividades de planificación del proyecto, y que sirve como guía para una buena ingeniería de software, no es en absoluto aconsejable embarcarse sin ellas.

Para garantizar la calidad es necesario la utilización de métricas adecuadas que permitan medir la calidad del proyecto (en realidad, se compara los parámetros de calidad de éste con estimaciones realizadas mediante el uso de estándares o datos que aporta la experiencia en otros proyectos).

La métrica; es el término que describe muchos y muy variados casos de medición. Siendo una métrica una medida estadística que se aplica a todos los aspectos de calidad de software, los cuales deben ser medidos desde diferentes puntos de vista como el análisis, construcción, funcional, documentación, métodos, proceso, usuario, entre otros.[4]

En la UCI, no se tiene definido cómo realizar las mediciones en los proyectos, de forma tal que estas medias puedan almacenarse en un repositorio de información central, con vistas a analizar la información y así estimar, planificar, tomar decisiones, darle seguimiento y control a los proyectos productivos en la universidad, ocurriendo similarmente la misma problemática a nivel de Facultad. Una buena práctica para realizar estas mediciones sería la aplicación de PSP y TSP.

Atendiendo a la **situación problémica** expuesta anteriormente, como **problema científico** se plantea la siguiente interrogante: ¿Cómo realizar una propuesta de medición basada en PSP y TSP a los proyectos de la facultad 4?

A partir del problema planteado el **objeto de estudio** de esta investigación está enfocado sobre el proceso de medición en los proyectos productivos de la facultad 4.

El **campo de acción** lo constituye la medición en los proyectos productivos de la facultad 4 utilizando PSP y TSP.

Como **objetivo general** se propone elaborar una propuesta de aplicación de PSP y TSP para realizar mediciones a los proyectos de la facultad 4 con vistas a resolver el problema planteado.

Se definen los siguientes **objetivos específicos:**

- Realizar un estudio de las tendencias mundiales sobre la utilización de las medidas que brindan el PSP y TSP.
- Caracterizar la aplicación de PSP y TSP en los proyectos de la facultad 4.
- Realizar una propuesta de aplicación de PSP y TSP para medir en los proyectos de la facultad 4.
- Realizar la evaluación técnica de la propuesta.

Para lograr los objetivos trazados, se llevan a cabo las siguientes **tareas de la investigación:**

- Realizar búsquedas bibliográficas sobre conceptos de métodos de estimación, planificación, métricas, mediciones e indicadores, PSP, TSP, repositorio de información.
- Comparar las definiciones más representativas.
- Realizar encuestas y entrevistas a proyectos en la facultad 4 para verificar si se mide y cómo se mide, en caso de que se realicen mediciones.
- Investigar si en la facultad 4 se aplica el PSP y TSP en los proyectos productivos y los beneficios que trae consigo las medidas resultantes para los proyectos y la universidad.
- Realizar una propuesta de medición utilizando PSP y TSP en los proyectos productivos de la facultad 4.
- Realizar la evaluación técnica de la propuesta.

CAPÍTULO 1: MARCO TEÓRICO

1.1 Introducción.

En la actualidad la producción de software se ha convertido en una de las principales e importantes actividades evidenciada por la alta dependencia de la sociedad, donde cada vez la exigencia crece por la necesidad de crear más y mejores productos software, lo más rápido y barato posible. Muchos proyectos de desarrollo son actualmente tan grandes y complejos que un pequeño grupo de especialistas no pueden manejarlos solos. Desafortunadamente no existe una señal de una nueva tecnología mágica para resolver esos problemas. Se debe mejorar la calidad y predictibilidad del trabajo o la sociedad tendrá que abandonar esos sistemas de software más sofisticados y sufrir los daños causados por sistemas inseguros e inestables.

Los métodos intuitivos de desarrollo de software usados hoy en día son aceptables solo porque no existen alternativas. La mayoría de los profesionales del software son notablemente creativos, pero la mayoría hacen realmente un trabajo pobre. Y no es sorpresa que las prácticas pobres producen productos pobres. La mayoría de los productos de software pueden ser creados para ser explotados, pero solo después de hacerle pruebas extensivas y reparaciones.

A media que los productos crecen y se hacen más complejos; los mismos son usados para aplicaciones críticas y el potencial de errores se incrementa. La industria del software ha respondido a esta amenaza con incrementadas y rigurosas pruebas consumidoras de tiempo.

El software actualmente controla la mayoría de los negocios, gobiernos y sistemas militares.

- Las fábricas son controladas por software.
- Los productos más avanzados son controlados por software.
- Las finanzas, la administración y las operaciones de negocios se ejecutan por software.

Los proyectos de software, grandes y pequeños, fallan por 4 razones:

1) Los acuerdos del proyecto son irreales.

- Mientras mayor sea el proyecto, menor la influencia que se tiene en el mismo.

2) Los grandes proyectos son difíciles de controlar.

- Hoy día, pocos desarrolladores tienen planes personales.

- Sin un plan, no se puede conocer el estado del trabajo.
- Si no se conoce dónde está, la administración no puede comprender el estado del trabajo.
- Si la administración no comprende el estado del trabajo, no pueden administrar los proyectos.

3) Los problemas de calidad se vuelven peores con el tamaño del proyecto.

- En los sistemas de software, si alguna parte tiene problemas de calidad, el sistema tendrá problemas de calidad.
- Si los desarrolladores no gestionan la calidad, sus equipos no pueden administrar la calidad.

4) Para ser efectivos, los equipos necesitan liderazgo y asesoramiento.

- Los líderes deben crear motivación en los equipo.
- El adiestramiento desarrolla cohesión de equipo.
- Equipos cohesionados, motivados y comprometidos, hacen el trabajo mejor.

Desde la década del 70, la calidad de software ha sido motivo de preocupación para especialistas, ingenieros, investigadores y comercializadores de software, los cuales han realizado gran cantidad de investigaciones al respecto con dos objetivos fundamentales:

1. ¿Cómo obtener un software con calidad?
2. ¿Cómo evaluar la calidad del software?

Ambas interrogantes conllevan amplias respuestas, pero están estrechamente ligadas con el concepto de la calidad del software, que es el resultado de la primera y la fuente de la segunda.

1.2 Calidad del Software.

La calidad del software es el grado con el que un sistema, componente o proceso cumple los requerimientos especificados y las necesidades o expectativas del cliente o usuario.[5].

La Calidad de software no es más que la concordancia del software producido con los requerimientos explícitamente establecidos, con los estándares de desarrollo prefijados y con los requerimientos implícitos no establecidos formalmente, que desea el usuario.[6]

La calidad del software es medible y varía de un sistema a otro o de un programa a otro. Por ejemplo, un software elaborado para el control de naves espaciales debe ser confiable al nivel de "cero fallas"; un software hecho para ejecutarse una sola vez no requiere el mismo nivel de calidad; mientras que un producto de software para ser explotado durante un largo período (10 años o más), necesita

ser confiable, mantenible y flexible para disminuir los costos de mantenimiento y perfeccionamiento durante el tiempo de explotación.

La calidad del software puede medirse después de elaborado el producto. Pero esto puede resultar muy costoso si se detectan problemas derivados de imperfecciones en el diseño, por lo que es imprescindible tener en cuenta tanto la obtención de la calidad, como su control durante todas las etapas del ciclo de vida del software.[7]

1.2.1 Control de la Calidad.

El control de la calidad es una serie de inspecciones, revisiones y pruebas utilizadas a lo largo del proceso del software, para asegurar que cada producto cumple con los requisitos que le han sido asignados. El control de la calidad incluye un bucle de realimentación del proceso que creó el producto. La combinación de medición y realimentación permite afinar el proceso cuando los productos de trabajo creados fallan al cumplir sus especificaciones. Este enfoque ve el control de la calidad como parte del proceso de fabricación.

Las actividades de control de la calidad pueden ser manuales, completamente automáticas o una combinación de herramientas automáticas e interacción humana. Un concepto clave del control de la calidad es que se hayan definido todos los productos y las especificaciones mensurables en las que se puedan comparar los resultados en cada proceso. El bucle de realimentación es esencial para reducir los defectos producidos.

1.2.2 La Garantía de la Calidad.

La garantía de la calidad consiste en la auditoría y las funciones de información de la gestión. El objetivo de la garantía de la calidad es proporcionar la gestión para informar los datos necesarios sobre la calidad del producto, por lo que se va adquiriendo una visión más profunda y segura de que la calidad del producto está cumpliendo sus objetivos. Por supuesto, si los datos proporcionados mediante la garantía de calidad identifican problemas, es responsabilidad de la gestión afrontar los problemas y aplicar los recursos necesarios para resolver aspectos de calidad.

1.2.4 ¿Cómo obtener un software con Calidad?

La obtención de un software con calidad implica la utilización de metodologías o procedimientos estándares para el análisis, diseño, programación y prueba del software que permitan uniformar la filosofía de trabajo, en aras de lograr una mayor confiabilidad, mantenibilidad y facilidad de prueba, a la

vez que eleven la productividad, tanto para la labor de desarrollo, como para el control de la calidad del software.

La política establecida debe estar sustentada sobre tres principios básicos: tecnológico, administrativo y ergonómico.

El principio tecnológico define las técnicas a utilizar en el proceso de desarrollo del software.

El principio administrativo contempla las funciones de planificación y control del desarrollo del software, así como la organización del ambiente o centro de ingeniería de software.

El principio ergonómico define la interfaz entre el usuario y el ambiente automatizado.

La adopción de una buena política contribuye en gran medida a lograr la calidad del software, pero no la asegura. Para el aseguramiento de la calidad es necesario su control y evaluación.

1.2.5 ¿Cómo controlar la Calidad del Software?

Para controlar la calidad del software es necesario, ante todo, definir los parámetros, indicadores o criterios de medición, ya que, como bien plantea Tom De Marco: "usted no puede controlar lo que no se puede medir".

Las cualidades para medir la calidad del software son definidas por innumerables autores, los cuales las denominan y agrupan de formas diferentes. Por ejemplo, John Wiley define métricas de calidad y criterios, donde cada métrica se obtiene a partir de combinaciones de los diferentes criterios. La Metodología para la evaluación de la calidad de los medios de programas de la CIC (Ciberhábitat, Ciudad de la Informática), de Rusia, define indicadores de calidad estructurados en cuatro niveles jerárquicos: factor, criterio, métrica, elemento de evaluación, donde cada nivel inferior contiene los indicadores que conforman el nivel precedente. Otros autores identifican la calidad con el nivel de complejidad del software y definen dos categorías de métricas: de complejidad de programa o código, y de complejidad de sistema o estructura.

Todos los autores coinciden en que el software posee determinados índices medibles que son las bases para la calidad, el control y el perfeccionamiento de la productividad.

Una vez seleccionados los índices de calidad, se debe establecer el proceso de control, que requiere los siguientes pasos:

- Definir el software que va a ser controlado: clasificación por tipo, esfera de aplicación, complejidad, de acuerdo con los estándares establecidos para el desarrollo del software.

- Seleccionar una medida que pueda ser aplicada al objeto de control. Para cada clase de software es necesario definir los indicadores y sus magnitudes.
- Crear o determinar los métodos de valoración de los indicadores: métodos manuales como cuestionarios o encuestas estándares para la medición de criterios experimentales y herramientas automatizadas para medir los criterios de cálculo.
- Definir las regulaciones organizativas para realizar el control: quiénes participan en el control de la calidad, cuándo se realiza, qué documentos deben ser revisados y elaborados, etc.

1.3 Planificación del proyecto.

El objetivo de la Planificación del proyecto de Software es proporcionar un marco de trabajo que permita al gestor hacer estimaciones razonables de recursos, costos y planificación temporal. Estas estimaciones se hacen dentro de un marco de tiempo limitado al comienzo de un proyecto de software, y deberían actualizarse regularmente a medida que progresa el proyecto. Además las estimaciones deberían definir los escenarios del mejor caso, y peor caso, de modo que los resultados del proyecto pueden limitarse.

El objetivo de la planificación se logra mediante un proceso de descubrimiento de la información que lleve a estimaciones razonables, pero si no se cuenta con una buena base histórica de información casi se podría afirmar que las estimaciones en el futuro convertida en planificación podrían crear conflictos en los indicadores del proyecto, afectando los compromisos de tiempo, costo y recursos pactados desde un inicio. [3]

1.4 Estimación del proyecto de software.

Estimación: predicción que tiene la misma probabilidad de estar por encima o por debajo del valor actual.[8]

En efecto, Ana María Moreno Sánchez Capuchino dice: “La primera tarea en la gestión de proyectos es la estimación.” [9]

La estimación, como actividad antecesora y constante de lo que luego será la planificación, proporciona valores aproximados de costos, tiempos y esfuerzos que se necesitarán para el desarrollo del producto a construir. Esa aproximación, requiere experiencia, acceder a una buena información histórica y el coraje de confiar en predicciones (medidas) cuantitativas cuando todo lo que existe son

datos cualitativos[10]; sin embargo, contar con dicha información en etapas tempranas de desarrollo, permite tomar decisiones importantes antes de llevarlo a cabo, es decir que de cierta forma se está prediciendo el futuro y así lo expresa también Ana Sánchez Capuchino definiendo la estimación como: "el proceso que proporciona un valor, a un conjunto de variables para la realización de un trabajo, dentro de un "rango aceptable de tolerancia".[9]

1.4.2 Estimación basada en el proceso.

Es la técnica más común para estimar un proyecto, es basar la estimación en el proceso que se va a utilizar, es decir, el proceso se descompone en un conjunto relativamente pequeño de actividades o tareas, y en el esfuerzo requerido para llevar a cabo la estimación de cada tarea.

Al igual que las técnicas basadas en problemas, la estimación basada en el proceso comienza en una delineación de las funciones del software obtenidas a partir del ámbito del proyecto. Se mezclan las funciones del problema y las actividades del proceso. Como último paso se calculan los costos y el esfuerzo de cada función y la actividad del proceso de software.

1.5 Métodos de estimación.

Existen diferentes métodos de estimación de los cuales va a depender una buena gestión de los proyectos como son:

1.5.1 Empíricos:

Donde los datos que soportan la mayoría de los modelos de estimación se obtienen de una muestra limitada de proyectos. Por esta razón el modelo de estimación no es adecuado para todas las clases de software y en todos los entornos de desarrollo. Por lo tanto los resultados obtenidos de dichos modelos se deben utilizar con prudencia.

1.5.2 Heurísticos:

Los métodos heurísticos suelen usarse como extensiones de otros métodos. Las heurísticas son reglas empíricas, desarrolladas mediante experiencia, que obtienen conocimiento acerca de relaciones entre atributos del modelo empírico. Las heurísticas se pueden utilizar para ajustar estimaciones realizadas con otros métodos.

1.5.3 Juicio del Experto:

Las opiniones de los expertos se basa principalmente en juicios emitidos por uno o varios expertos avalados por su experiencia en entornos similares y apoyados, en algunos casos, en datos objetivos obtenidos de proyectos anteriores y almacenados[11], aunque no se incluyen dentro del marco de trabajo para seleccionar métodos de estimación, ya que estos métodos no se pueden caracterizar fácilmente.

1.5.4 Puntos de Casos de Uso:

Este método estima el esfuerzo de desarrollo de un producto de software a partir de los Casos de Uso y algunos factores de complejidad técnica y ambiente que influyen en el desarrollo. Fue propuesto originalmente por Gustav Karner y posteriormente refinado por muchos otros autores.

Este método exige la existencia de un modelo de casos de uso, por lo que se deberá comenzar a aplicar, una vez que se tenga algún entendimiento del dominio del problema o cuando se estén realizando las labores de arquitectura y dimensionamiento del tamaño del sistema[12].

El método utiliza los actores y casos de uso identificados para calcular el esfuerzo que costará desarrollarlos. A los casos de uso se les asigna una complejidad basada en transacciones, que son pares de pasos acción-usuario->respuesta-sistema de los escenarios de los casos de uso. A los actores se les asigna una complejidad basada en el tipo de actor, es decir, si son interfaces con usuarios o si son interfaces con otros sistemas (API o Protocolo). También se utilizan factores de entorno y de complejidad técnica para afinar el resultado[13] Una vez asignada la complejidad, se calculan los puntos de caso de uso no ajustados, el TCF (factor de complejidad técnica) y el EF (factor del entorno). Con ellos, se calculan los puntos de caso de uso o UCP, que finalmente se traducen a esfuerzo en horas-hombre con un sencillo cálculo [13].

Una de las principales desventajas de este método es que no existe una teoría de cómo escribir o estructurar correctamente los casos de uso, por lo que todas las medidas de tamaño y estimación serán afectadas por la rigurosidad de los analistas[12].

1.5.5 COCOMO:

Barry Boehm, en su libro clásico sobre economía de la Ingeniería del Software, introduce una jerarquía de modelos de estimación de Software con el nombre de COCOMO, por su nombre en Ingles

(Constructive, Cost, Model) modelo constructivo de costos. La jerarquía de modelos de Boehm esta constituida por los siguientes:

- **Modelo I.** El Modelo COCOMO básico calcula el esfuerzo y el costo del desarrollo de Software en función del tamaño del programa, expresado en las líneas estimadas.
- **Modelo II.** El Modelo COCOMO intermedio calcula el esfuerzo del desarrollo de software en función del tamaño del programa y de un conjunto de conductores de costos que incluyen la evaluación subjetiva del producto, del hardware, del personal y de los atributos del proyecto.
- **Modelo III.** El modelo COCOMO avanzado incorpora todas las características de la versión intermedia y lleva a cabo una evaluación del impacto de los conductores de costos en cada caso (análisis, diseño, etc.) del proceso de ingeniería de Software.

1.5.6 Puntos de Función:

El método de puntos de función fue creado por Allan Albretch y se basa principalmente en la identificación de los componentes del sistema informático en términos de transacciones y grupos de datos lógicos que son relevantes para el usuario en su negocio. A cada uno de estos componentes les asigna un número de puntos por función basándose en el tipo de componente y su complejidad; y la sumatoria de esto, da los puntos de función sin ajustar. El ajuste es un paso final basándose en las características generales de todo el sistema informático que se está contando. [14]

Los objetivos para calcular Puntos de Función son:

- Medir lo que el usuario pide y lo que el usuario recibe.
- Medir atributos independientemente de la tecnología utilizada en la implantación del sistema.
- Proporcionar una métrica de tamaño que de soporte al análisis de la calidad y la productividad.
- Proporcionar un medio para la estimación del software.
- Proporcionar un factor de normalización para la comparación de distintos software.

1.6 Métricas del software.

A la hora de planificar un proyecto, es esencial hacer las estimaciones del esfuerzo humano, a través de las mediciones de software, obteniendo registros históricos de datos. Se mide el software con el objetivo de indicar la calidad del producto y evaluar la productividad de las personas

involucradas durante el desarrollo del software, para evaluar los beneficios derivados del uso de nuevas técnicas y herramientas; para establecer una línea base para la estimación y para justificar el uso de nuevas herramientas y la necesidad de información. Las palabras claves a la hora de medir son: señalar, predecir, evaluar y mejorar. Las mediciones de software han venido estudiándose desde hace varios años y precisamente son conocidas como métricas de software.

Es muy común asociar la palabra métrica con los términos medida y medición, aunque son completamente diferentes, es por ello que, existe una confusión en la utilización de los términos métricas de software y mediciones de software, sin embargo, en la literatura los términos métrica, medida o medición son usados como sinónimos[15].

Existen varias razones para medir un producto.

1. Para indicar la calidad del producto.
2. Para evaluar la productividad de la gente que desarrolla el producto.
3. Par evaluar los beneficios en términos de productividad y de calidad, derivados del uso de nuevos métodos y herramientas de la ingeniería de software.
4. Para establecer una línea de base para la estimación
5. Para ayudar a justificar el uso de nuevas herramientas o de formación adicional.

Además se conocen varios tipos de métricas de software de las cuales se da una breve descripción a continuación.

- **Métricas técnicas:** Se centran en las características de software por ejemplo: la complejidad lógica, el grado de modularidad. Mide la estructura del sistema, el cómo está hecho.

- **Métricas de calidad:** proporcionan una indicación de cómo se ajusta el software a los requisitos implícitos y explícitos del cliente. Es decir cómo voy a medir para que el sistema se adapte a los requisitos que pide el cliente.

- **Métricas de productividad:** Se centran en el rendimiento del proceso de la ingeniería del software. Es decir que tan productivo va a ser el software que se va a diseñar.

- **Métricas orientadas a la persona**: Proporcionan medidas e información sobre la forma que la gente desarrolla el software de computadoras y sobre todo el punto de vista humano de la

efectividad de las herramientas y métodos. Son las medidas que se deben hacer al personal que hará el sistema.

- **Métricas orientadas al tamaño:** Permiten conocer en qué tiempo se va a terminar el software y cuántas personas se van a necesitar. Son medidas directas al software y el proceso por el cual se desarrolla, si una organización de software mantiene registros sencillos, se puede crear una tabla de datos orientados al tamaño.

- **Métricas orientadas a la función:** Son medidas indirectas del software y del proceso por el cual se desarrolla. En lugar de calcularlas las LDC, las métricas orientadas a la función se centran en la funcionalidad o utilidad del programa.

Las métricas orientadas a la función fueron el principio propuestas por Albercht, quien sugirió un acercamiento a la medida de la productividad denominado método del punto de función. Los puntos de función que obtienen utilizando una función empírica, basando en medidas cuantitativas del dominio de información del software y valoraciones subjetivos de la complejidad del software.

1.7 Estándares de calidad.

Las métricas son utilizadas en varios estándares ó modelos de procesos, para dar una mejoría en la calidad de software a nivel de empresa u organización, entre estos estándares reconocidos internacionalmente se encuentran:

- **CMMi**: Modelo de Capacidad de Madurez Integrado. Modelo de mejora de procesos organizado en áreas de proceso que determinan el nivel de capacidad o madurez de una organización.
- **ISO 9003-04**: Guía específica sobre la implantación del modelo de calidad ISO 9001 para las organizaciones de software. Complementa otros estándares como el ISO/IEC 19207 o ISO/IEC 15504.[16]
- **SPICE (ISO/IEC 15504):** Modelo similar al CMM que se utiliza para la mejora de procesos y medir la capacidad (Propuesta europea 2005)[16].
- **IDEAL:** Es un modelo propuesto por el Instituto de Ingeniería de Software (SEI), para evaluación de los procesos de software.[16]

- **IEEE 12207**: Estándar que incluye una serie de procesos definidos para el ciclo de vida de los productos software, incluyendo aquellos asociados a los procesos de calidad.[16]
- **CMM**: el Modelo de Capacidad y Madurez, es un modelo de evaluación de los procesos de una organización. Este modelo establece un conjunto de prácticas o procesos clave agrupados en Áreas Clave de Proceso (KPA - Key Process Área). Para cada área de proceso se define un conjunto de buenas prácticas que serán: definidas en un procedimiento documentado, provistas (la organización) de los medios y formación necesarios, ejecutadas de un modo sistemático, universal y uniforme (institucionalizadas), medidas y verificadas.[17]

Para lograr buenos resultados a nivel de las empresas desarrolladoras de software, primero es necesario, que cada uno de sus ingenieros alcancen resultados de excelencia y luego que sus grupos de trabajo también lo hagan. En el caso del desarrollo de software de alta calidad, que responda a las expectativas de los usuarios, es fundamental definir procesos tanto personales como para el trabajo en equipo.

Se pueden establecer 3 niveles sobre los cuales se puede desarrollar el proceso de mejora continua:

- Nivel Organizacional (Capacidades de Dirección)
- Nivel del Equipo (Ejecución del Equipo)
- Nivel del Individuo (Conocimientos y Disciplina Personal)

En cuanto al nivel del Individuo y al de Equipo, se explica dos de los más importantes procesos, el PSP y el TSP respectivamente.

1.8 PSP.

"El PSP es un método basado en el "proceso", el cual aplica las definiciones de procesos y principios con base en las tareas personales."[19]

Este modelo es una continuación de la CMM, ya que al igual que éste, trata de demostrar que es más productivo trabajar con prácticas de ingeniería de software y también es benéfico para su mantenimiento. También es importante que en este método los ingenieros entiendan que si se usa este tipo de direccionamiento, los métodos serían perfeccionados y se entendería que hay mecanismos que podrían mejorar el desarrollo.

El **figura 1** se refleja la relación ascendente que se alcanza desde el nivel personal hasta el organizacional con el PSP, TSP, CMMI.

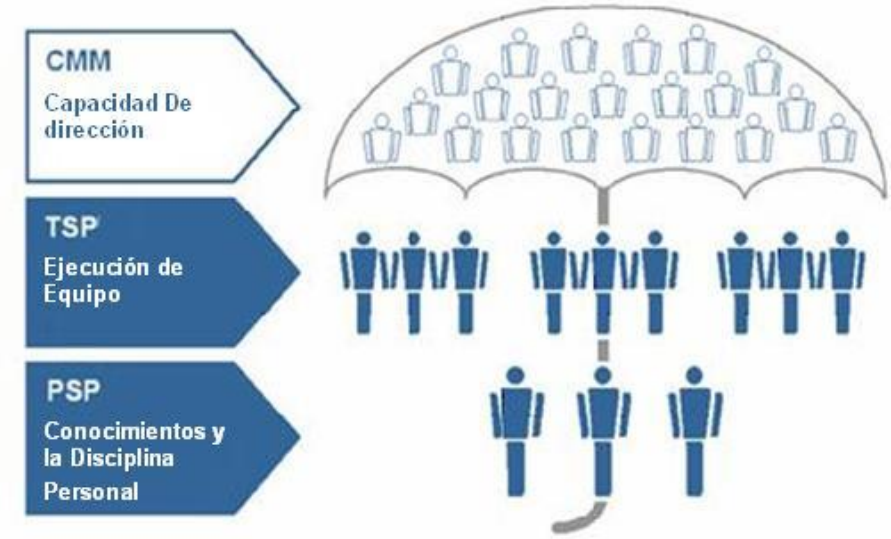

Figura 1: Relación entre CMMI, TSP y PSP.

El PSP considera ayudas para desarrollar las habilidades y hábitos necesarios para planificar, guiar y analizar proyectos complejos desde una perspectiva personal. Así como en CMM existen cinco niveles de capacidades a nivel organizacional, en PSP se plantean cuatro niveles de evolución a nivel personal.[20]

El PSP es el que enseña a las personas que practican la ingeniería de software a:

- Manejar la calidad de los proyectos.
- Mejorar la aproximación y la planificación.
- Reducir defectos en los productos y tareas.

El PSP puede ser utilizado en:

- Desarrollo de pequeños programas.
- Pruebas de los sistemas utilizados.
- Mantenimiento de sistemas.
- Mantenimiento de software en equipos grandes.
- Verificación y validación de técnicas.

1.8.1 Principios de PSP.

PSP mantiene los siguientes principios:

- La calidad de un sistema de software, está determinada por la calidad de sus peores componentes.
- La calidad de un componente de software, está gobernada por el individuo que lo desarrolla.
- La calidad de un componente de software, está gobernada por la calidad de los procesos usados para desarrollarlo.
- La clave para la calidad son las habilidades individuales del desarrollador, compromiso y disciplina personal de proceso.
- Como un profesional del software, se debe ser responsable del proceso personal.
- Cada persona debe medir, monitorear y analizar su trabajo.
- Cada persona debe aprender de sus variaciones de desempeño.
- Cada persona debe incorporar lecciones aprendidas en sus prácticas personales.

1.8.2 Niveles del PSP.

En la **figura 2** se presenta un esquema del proceso personal de desarrollo de software. Es un proceso evolutivo: se parte del estado actual del desarrollador, se van agregando elementos, dando lugar a nuevos procesos, y así se continúa hasta llegar a la última etapa en la cual se repite el proceso personal cíclicamente.[24]

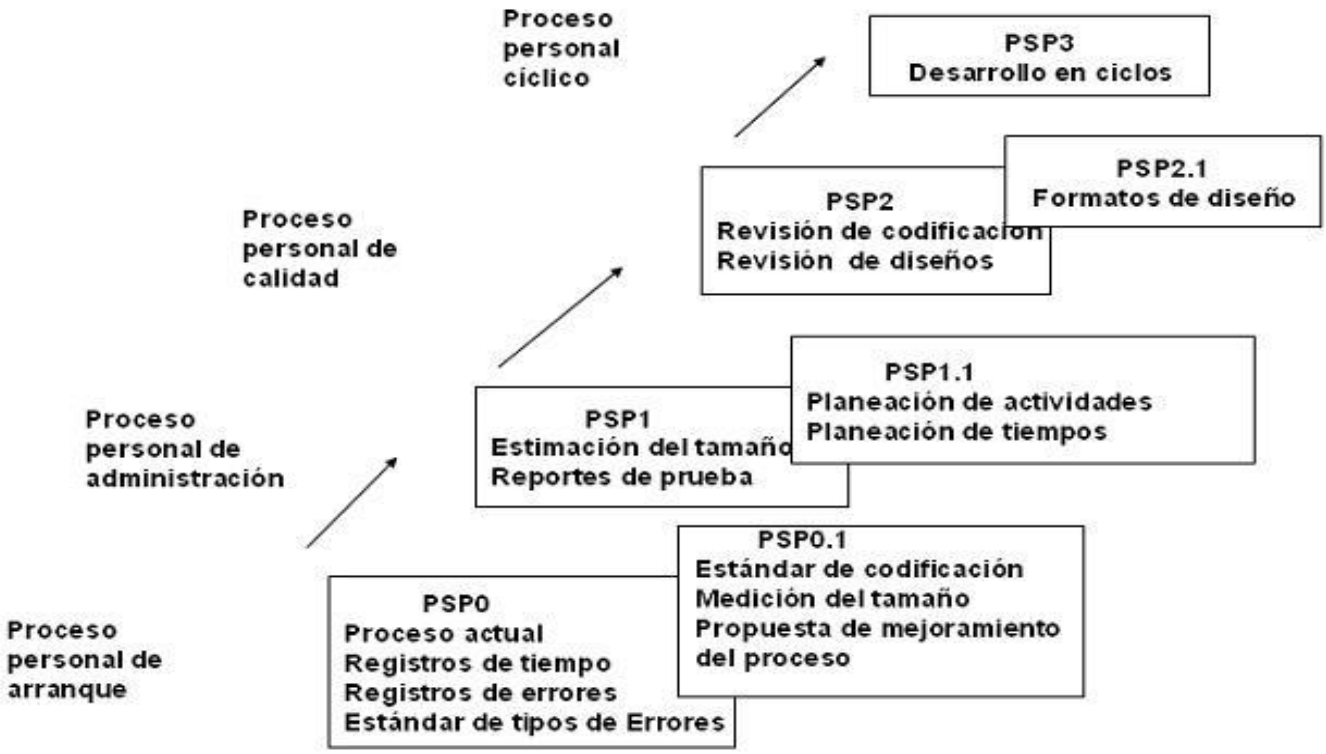

Figura 2: La evolución de PSP.

El propósito de PSP es ayudar a mejorar las habilidades de ingeniería de software. El mismo constituye una herramienta poderosa que puede ser empleada de varias formas. Por ejemplo, ayuda a administrar el trabajo personal, evaluar los talentos y formar habilidades. Contribuye a la realización de mejores planes, para monitorear el desempeño de forma precisa y medir la calidad de los productos. Independientemente de que se diseñen programas, se desarrolle requerimientos, se escriba la documentación o se de mantenimiento al software existente, el PSP ayuda a hacer mejor el trabajo.

En lugar de usar un enfoque para cada trabajo, es necesario un arreglo de herramientas y métodos así como las habilidades expertas para usarlas apropiadamente. El PSP brinda los datos y técnicas de análisis necesarios para determinar qué tecnologías y métodos trabajan mejor para cada ingeniero.

El PSP también brinda un marco de trabajo para comprender por qué se cometen errores y cómo es mejor encontrarlos, arreglarlos y prevenirlos. Es posible determinar la calidad de las revisiones, los tipos de defectos que comúnmente se obvian y los métodos de calidad más efectivos.

Después de conocer PSP, se podrá decidir qué métodos emplear para usar y cuándo usarlos. También se conocerá cómo definir, medir y analizar el propio proceso. Entonces a medida que se

obtenga experiencia, se reforzarán los procesos para tomar ventaja de cualquier nueva herramienta y métodos de desarrollo.

Como el PSP está enfocado a tareas como la predicción, la verificación de calidad y productividad; las cuales están diseccionadas como procesos individuales, ya que al principio solo se tiene apuntado hacia una meta, pero para que esto realmente funcione en un proceso grande, se necesita un conjunto de este tipo de actividades, lo cual lleva de la mano a la explicación del TSP.

1.9 TSP.

El TSP es el desarrollo "real" del proceso, ya que es el modelo que permite ver cómo usar el PSP en un proyecto de desarrollo y cómo el creador y su equipo pueden desarrollar las mejores prácticas para aumentar y perfeccionar lo que se hizo en el PSP.[19]

La **figura 3** muestra la relación que existe entre PSP y TSP y las características que los distingue:

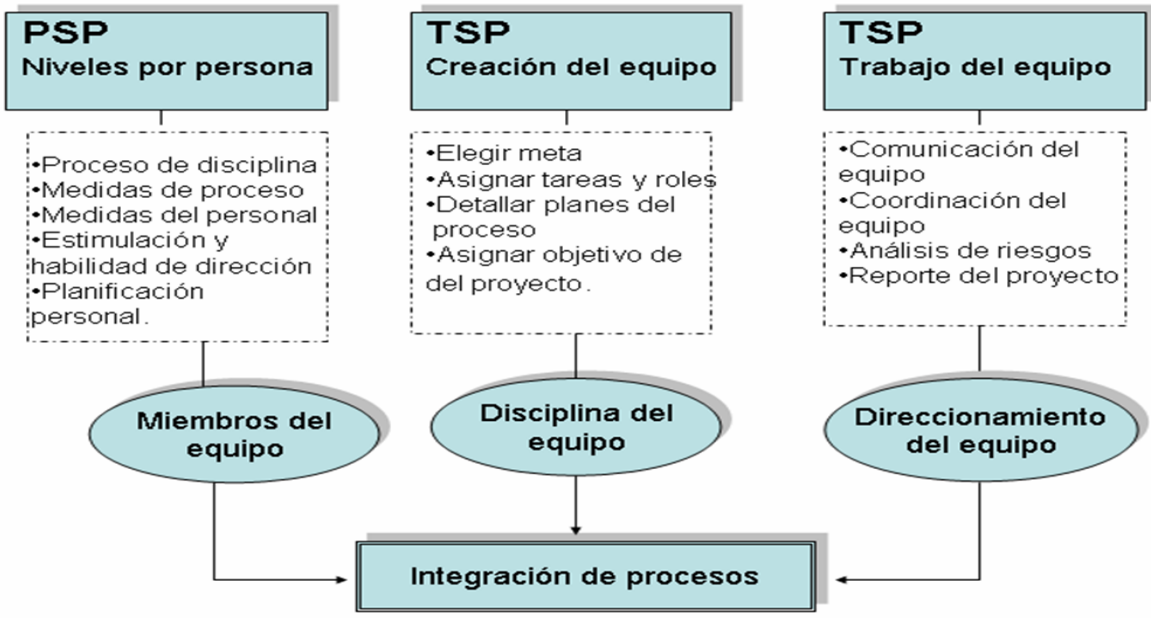

Figura 3: Estructura del TSP.

Esto quiere decir que el TSP no funcionaría (o al menos no de una manera exitosa) sin el PSP, ya que éste es dependiente de la buena realización de las tareas individuales para que al momento de concatenarlo, de un buen resultado como un equipo. Se puede decir que el TSP se divide en dos componentes: un equipo el cual se encarga de construir el proceso durante el proyecto, y el otro es un equipo que se encarga del funcionamiento y del mantenimiento de los procesos.

Algunas de las características más importantes del TSP son:

- Establecer metas.
- Crear control.
- Mantener la calidad en cualquier etapa de los procesos.
- Afrontar los problemas como un equipo.
- Mantener el control dentro de la organización.
- Crear y asegurar software seguro y de alta calidad.

La recompensa de trabajar con este tipo de modelos es que mejora la calidad de los procesos y reducen los costos, esto gracias a la generación mínima de errores y el poco tiempo en que estos procesos se realizan. También estos procesos permiten ser modificados fácilmente y sobre todo, es factible tener un buen mantenimiento.[19]

El éxito de estos modelos, no solo implican la buena calidad que generan, también una herramienta clave que manejan son las visiones a largo plazo, ya que de eso se trata. Al crear modelos los cuales permiten estructurar de manera organizada procesos desde lo particular a lo general, no solo se manejan manejas sus funciones para la resolución de tareas presentes, ya que el mundo de la tecnología está evolucionando rápidamente, y los usuarios que buscan este tipo de modelos, entienden que si no buscan metas y soluciones a largo plazo, simplemente serán desplazados por nuevos usuarios, o por usuarios que generan una proyección futura.

1.10 TSPi.

TSPi es una versión a escala reducida del TSP por lo que es más flexible.

Principios básicos del TSPi.

1- El aprendizaje es más efectivo cuando se sigue un proceso bien definido y se tiene una rápida retroalimentación.[18]

- Los formularios y los guiones de TSPi proporcionan una armazón repetitiva, medible y definida para hacer ingeniería de software en equipo.
- TSPi proporciona una rápida retroalimentación, porque el equipo desarrolla el producto en varios ciclos cortos de desarrollo y evalúa el resultado después de cada ciclo.

2- Un trabajo productivo en equipo requiere una combinación de metas específicas, un ambiente de trabajo que lo soporte, una dirección capaz y un liderazgo.

- La meta del equipo es elaborar un producto que funcione.
- El TSPi proporciona un ambiente con soporte adecuado, donde uno de los miembros del equipo será el líder y el instructor proporciona el entrenamiento.

3- Cuando se han enfrentado a problemas de proyectos reales y han estado guiados hacia soluciones efectivas, se aprecian los beneficios de estas prácticas sólidas de desarrollo.

- Sin la guía precisa del TSPi, se gasta mucho tiempo en definir las propias prácticas, métodos y roles.

4- La instrucción es más efectiva cuando tiene lugar en alguien dotado con un conocimiento anterior.

- Experiencia con equipo de software
- Cursos de equipos de software.

Diseño del TSPi.

TSPi propone 7 decisiones principales de diseño:

1. Proporciona una armazón simple fundamentada en el PSP.
2. Desarrolla productos en varios ciclos.
3. Establece métricas estándar para calidad y ejecución.
4. Proporciona métricas precisas para equipos y estudiantes.
5. Usa roles y evaluaciones para el equipo.
6. Requiere disciplina en el proceso.
7. Proporciona orientación en los problemas de trabajo en equipo.

En la **figura 4** se presenta un esquema de la estructura y del flujo de TSPi. Para usar una estrategia de desarrollo cíclica se requiere definir el producto básico que se ha de generar con el primer ciclo. Se recomienda que sea una versión pequeña del producto, pero que incluya alguna funcionalidad que de valor al negocio. Para decidir el contenido y tamaño de cada ciclo se debe tener en cuenta que debe producir una versión que pueda probarse y que represente un subconjunto de la versión final del producto, que debe ser lo suficientemente pequeño para que pueda desarrollarse y probarse en el

tiempo disponible y que la combinación de los productos obtenidos en cada ciclo debe dar el producto final deseado.[18]

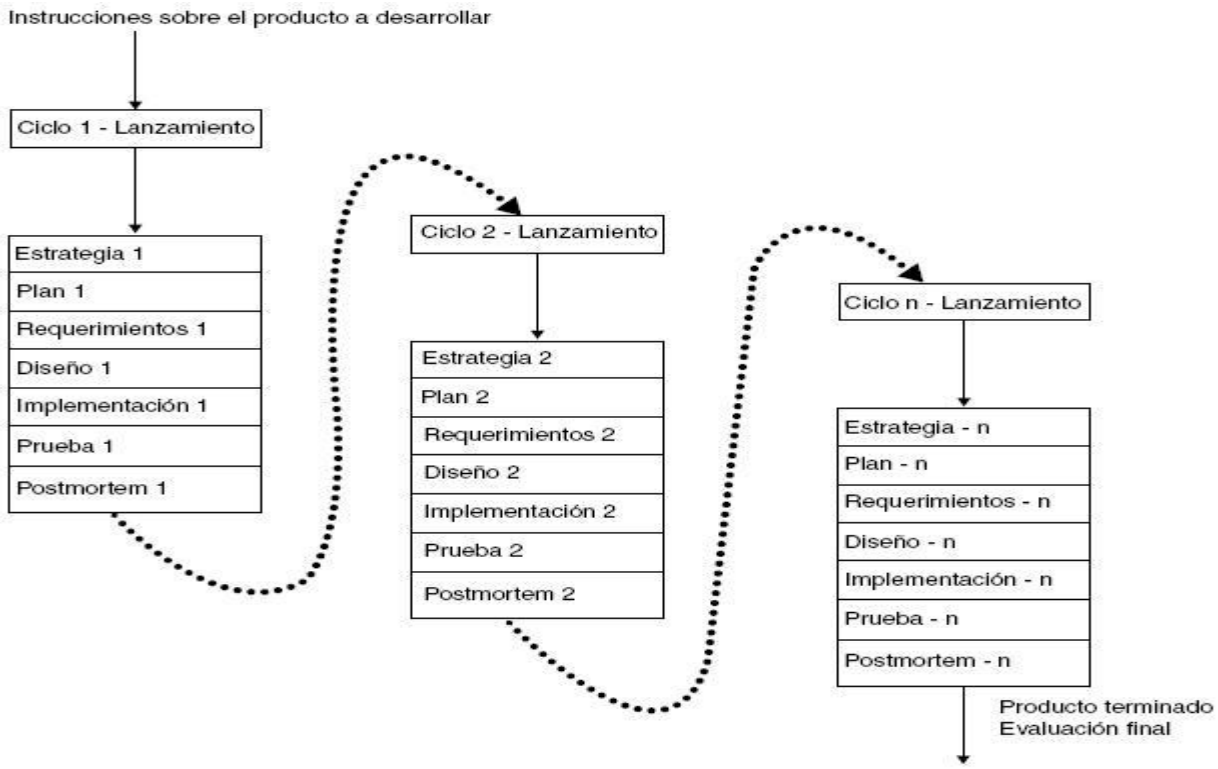

Figura 4: Estructura y flujo de TSPi.

1.11 Conclusiones.

En este capítulo se realizó un estudio del PSP y TSP, como parte de la fundamentación teórica y se llegó a la conclusión de que cuando se aplica PSP y TSP en el desarrollo de proyectos de software se obtiene un producto con mayor calidad, es decir hay una reducción significativa en los errores que presenta el software, hay una reducción en el tiempo que toma para desarrollar el software, es más fácil y rápida la estimación del tiempo que se necesita para programar. En caso de ser necesario realizar algún cambio es fácil hacerlo. Los costos de desarrollo se reducen en gran proporción, se tiene control total sobre el proceso de desarrollo y la productividad de los ingenieros aumenta, tanto personal como colaborativamente.

Se realizó además un análisis para verificar en qué medida proponer mejoras en cuanto a los aspectos medibles y que puedan contribuir paso a paso en la calidad de los productos de software que se obtendrán a partir de su aplicación.

CAPÍTULO 2: PROPUESTA DE APLICACIÓN

2.1 Introducción.

El capítulo se inicia con un una breve descripción de los proyectos donde se realizó una encuesta para conocer en qué medida en la facultad 4 se aplica el PSP y TSP, la cual demostró que en cierta mediada en dichos proyectos se aplican algunos elementos de dichos procesos y luego del análisis de la misma se hace un propuesta de aplicación de PSP y TSP con el objetivo de realizar mediciones para obtener datos reales sobre los proyectos que permitan realizar mejores estimaciones y por ende una mejor planificación de los mismos.

2.2 Proyectos de software desarrollados en la facultad 4.

A continuación se resume en qué consiste cada proyecto seleccionado para realizar una encuesta que ayudará a obtener resultados según los objetivos de esta tesis de curso. Los proyectos seleccionados son: Banco, Aduana: Gestión Integral de Recursos Humanos, Aduana: Información Adelantada, Aduana: Gestión de los Procesos Aduanales, ResiSoft, SIGEP y MINFAR

2.2.1 Banco.

En la actualidad la consolidación gradual de la economía cubana va en aumento, por lo que es una necesidad estratégica la modernización del sistema bancario cubano, para el mejoramiento de las operaciones en las entidades bancarias y financieras del país, abarcando al Banco Central y banca comercial. Actualmente en el país existe en explotación un sistema para realizar las operaciones bancarias (SABIC), que cuenta con un fuerte módulo contable y a la vez permite realizar un número significativo de funcionalidades bancarias; sin embargo existe un grupo de operaciones que no son permisibles por este sistema y otras que difieren en los datos que manejan los disímiles bancos, debido a las diferentes versiones existentes entre ellos (MS-DOS, Visual FoxPro y SQL).

2.2.2 Aduana: Gestión Integral de Recursos Humanos.

Este proyecto pretende proporcionar a la Aduana General de la República (AGR) una aplicación que resuelva los problemas que existen en la gestión de sus recursos humanos. Esta aplicación estará compuesta por los siguientes módulos en su primera versión:

- Selección de Candidatos.
- Gestión de la Estructura y Composición.

- Control de los Trabajadores.
- Gestión de las Nóminas de Pago.

2.2.3 Aduana: Información Adelantada.

Este proyecto pretende proporcionar a la AGR y a otras Aduanas de Latinoamérica y el Caribe una aplicación que gestione la información adelantada de todos los pasajeros y las cargas que deben arribar al país de la aduana en cuestión, ya sea al viajar por vía aérea o por vía marítima. El sistema debe facilitar el manejo de dicha información y proporcionar las facilidades necesarias para que la información llegue en el tiempo requerido, así como proporcionar con rapidez la respuesta a la información recibida. En sentido general el principal objetivo del proyecto es garantizar que la información adelantada tanto de carga como de pasajeros se encuentre en el lugar deseado y de la manera más correcta posible para que pueda ser utilizada por los organismos correspondientes de ese país y los remitentes de la misma.

2.2.4 Aduana: Gestión de los Procesos Aduanales.

Con este proyecto se pretende informatizar todos los procesos de la AGR cumpliendo con las especificaciones del país sobre los Medios de Transporte Internacional, Importaciones y Exportaciones con y sin carácter comercial, Bultos Postales y Viajeros; a través del desarrollo de nuevas versiones de los sistemas de Ingresos Comerciales, Despacho No Comercial, Tablas de Control, Medios de Transporte Internacional y Enfrentamiento de forma tal que se ajuste a la nueva arquitectura del Sistema Único de Aduanas, así como la realización de la primera versión del Manifiesto.

2.2.5 ResiSoft.

El proyecto de informatización de la residencia, se constituye con el objetivo de informatizar los procesos propios de la residencia de la Universidad de las Ciencias Informáticas a través del diseño del producto “ResiSoft” que es un sistema informático basado en una arquitectura Orientada a Servicios, utilizando para su desarrollo herramientas basadas en Software libre. Dicho sistema garantizará la alta integridad, disponibilidad y confiabilidad de la información que se gestiona en la residencia universitaria, así como una elevada eficiencia en los procesos de gestión de la información con el correspondiente ahorro de materiales de oficina. De esta manera se debe lograr una mejora en el funcionamiento eficiente de la organización.

2.2.6 SIGEP: Automatización de las Instalaciones Penitenciarias.

En aras de mitigar algunos de los problemas actuales del Sistema Penitenciario Venezolano, se opta por la implantación de un sistema informático que soporte las decisiones estratégicas del Ministerio del Interior y Justicia y de la Dirección General de Custodia y Rehabilitación del Recluso, que contribuya a garantizar el respeto a los derechos de los internos, su actividad de rehabilitación y reinserción en la sociedad.

2.2.7 MINFAR.

Este proyecto constituye la automatización de los procesos que se realizan en las Fuerzas Armadas Revolucionarias de Cuba (FAR) creando un ERP para las FAR.

2.3 Análisis de los resultados de la encuesta.

La planificación del proyecto de software le permite al planificador hacer estimaciones razonables de recursos, costos y tiempo. Estas estimaciones se hacen dentro de un marco de tiempo limitado al comienzo de un proyecto de software, y deberán actualizarse regularmente en la medida en que progresa el proyecto. Además las estimaciones deberán definir los escenarios del mejor y peor caso, de modo que los resultados del proyecto puedan limitarse.

Para realizar una planificación adecuada, en los proyectos de software, es necesario contar con una base histórica de información real, que permita realizar buenas estimaciones. PSP provee una estructura bien definida para tareas pequeñas y la medición de éstas. El primer paso en el proceso de PSP es la planeación, en la cual es necesario, llevar un resumen del plan para registrar todos los datos de esta fase. Se debe registrar cada dato que interviene en el desarrollo del programa, es decir, el tiempo en cada proceso y los defectos que se encuentran durante todo el proceso de desarrollo e inclusive desde la planeación. Al final del trabajo, durante la fase de postmortem, se mide el tamaño del programa y se incorporan estos datos en el formato del resumen del proyecto.

Para obtener datos hay que medir. En las encuestas (Ver anexo 5) realizadas en los Grupos de Proyectos, quedó demostrado que de una forma u otra, en todos se realizan mediciones, pero los datos de las mismas, no se guardan en un repositorio de información central con vistas a analizar la información, así como estimar, planificar, tomar decisiones, darle seguimiento y control a los proyectos productivos de la facultad. Además, las mediciones varían de un proyecto a otro condicionado por el grado de madurez y desarrollo que han alcanzado; tomando como ejemplo, el proyecto SIGEP que se considera uno de los mejores desarrollados en la UCI, y que aunque no se recogieron las mediciones a

diario, sí se tenía un control sobre los tiempos, tamaños, y otras medidas que eran de conocimiento de los desarrolladores del proyecto, evidenciando así la utilidad que proveían las mediciones para comparar el avance, estancamiento, cumplimientos y atrasos. Que propiciaron tener bien documentada la información para próximas utilidades y consultas a niveles superiores. No siendo de esta forma para el proyecto de Banco, al cual resulta de interés alertarlos sobre la necesidad de hacer las mediciones correctas desde su fase de inicio.

En la figura que se muestra a continuación (**ver figura 5**), se resaltan los resultados por proyectos en cuanto a las mediciones hechas hasta el momento, destacando que la mayoría, respondieron a la encuesta realizada, que sí realizaban las mediciones correspondientes para tener una idea de cómo marcha el desarrollo del proyecto, lo cual será llevado a profundas reflexiones a lo largo de un análisis de las otras respuestas, evidenciándose grandes contradicciones, en este caso la mayoría de los proyectos afirman que hacen mediciones. Unido a ello, la encuesta demostró también, que en dichos proyectos se aplican elementos de PSP y TSP; pero de forma no organizada y controlada según los niveles de PSP y TSP.

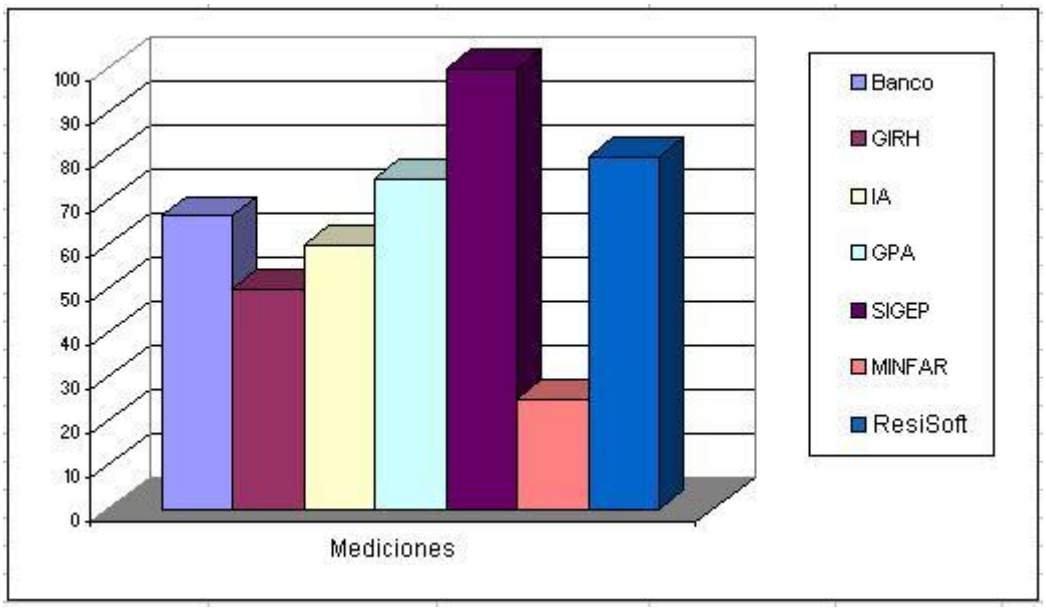

Figura 5: Mediciones en los proyectos.

Estos elementos no están al mismo nivel en todos los proyectos, como es el caso de la planificación de la actividades con un 100 por ciento de realización lo que conlleva a analizar que los proyectos sí pueden estarse planificando, pero que estas planificaciones solo se están quedando en el documento y viéndose la utilidad cuando de una forma u otra se le hace presión al proyecto en cuanto a la existencia de los cronogramas, pero las medidas en sí no se están recogiendo y una de las importancias que se le atribuye a esta acción es que el proyecto en caso de atraso no puede

justificarse porque no recogió las medidas pertinentes ni aplicó métricas para tener buenas bases y buenos argumentos.

En la siguiente figura (**ver figura 6**) se puede visualizar mejor el estado de las actividades de planificación en los proyectos a un nivel de 100% de realización.

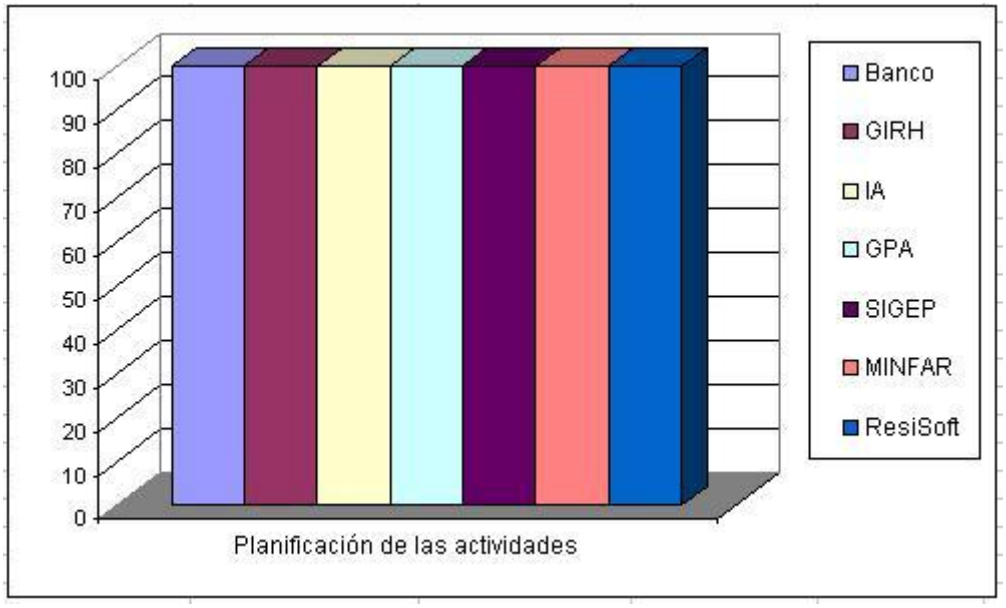

Figura 6: Planificación de las Actividades.

Si se establece una comparación entre las respuestas dadas sobre la afirmación relacionada a la planificación de las actividades, y las respuestas con respecto a la gestión de la planificación del tiempo, es evidente que existe una contradicción. Se considera que recoger las medidas de tiempo, es una de las métricas triviales que se debe tener en un proyecto, y sin embargo, según muestra la **figura 7** solo el proyecto SIGEP está por encima del 60 por ciento respondiendo que sí gestiona la planificación de los tiempos de desarrollo. No se tienen entonces los resultados reales en las planificaciones y se deja de dar la importancia debida a la gestión de las mismas.

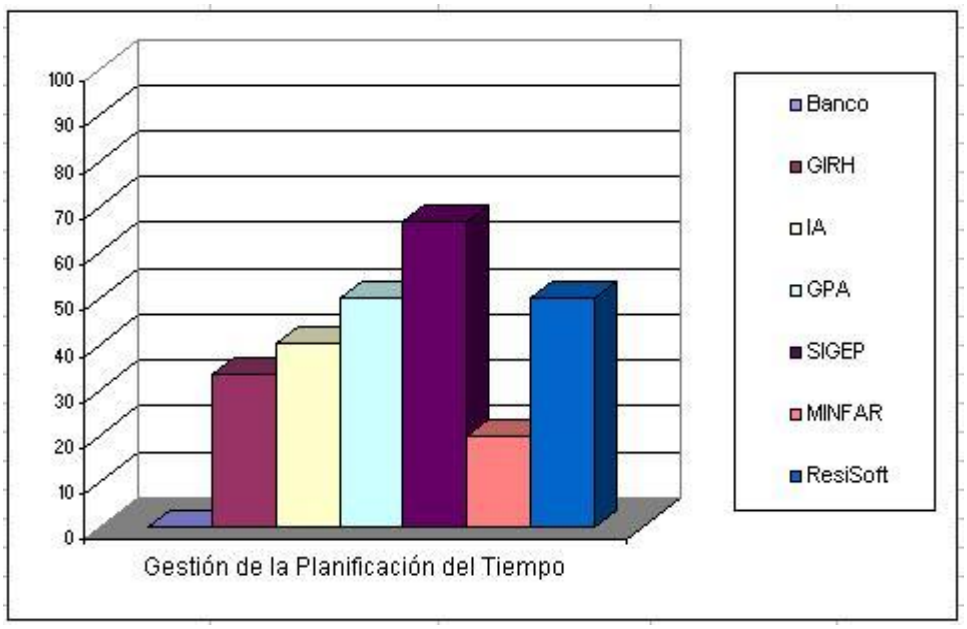

Figura 7: Gestión de la Panificación del Tiempo.

Un elemento de PSP que se relaciona directamente con la calidad del software es: el Reporte de Pruebas; pero en algunos casos no se hacen todas las pruebas necesarias (**ver figura 8**). Sin las pruebas no se puede hacer un buen reporte de éstas. Además de esto tampoco se lleva un registro de defecto eficiente lo que no permite tener una idea de los fallos del producto y como corregirlos.

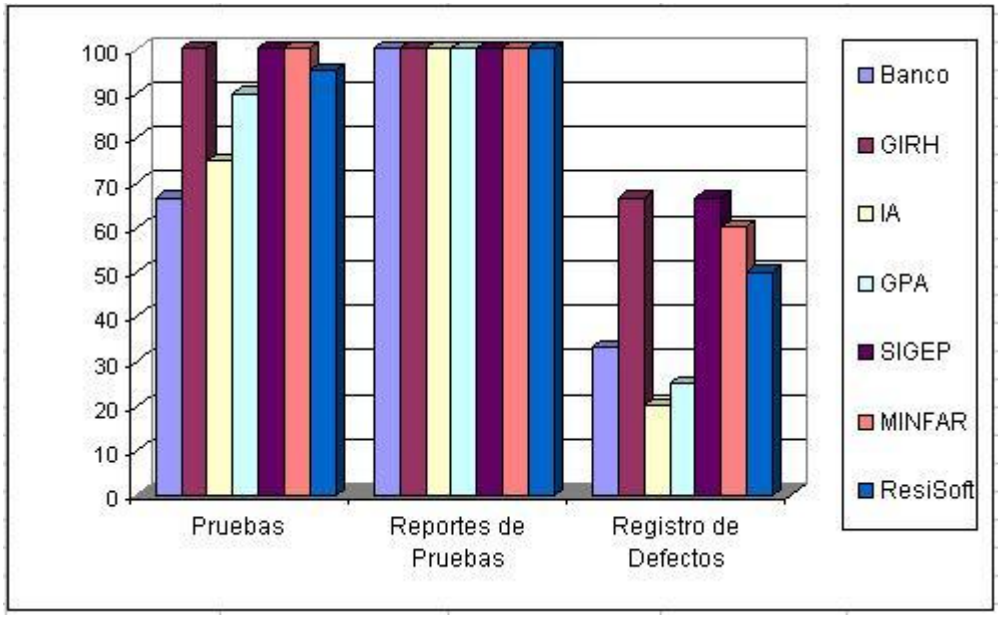

Figura 8: Prueba y Reportes de Pruebas.

Otro elemento importante de PSP que está presente de forma irregular en dichos proyectos es el relacionado con la estimación y la medición del tamaño del software como se muestra el la **figura 9**. En la mayoría de los proyectos se estima el tamaño pero éste no se mide lo que trae como consecuencia que en futuras estimaciones no se cuente con un valor real para guiarse.

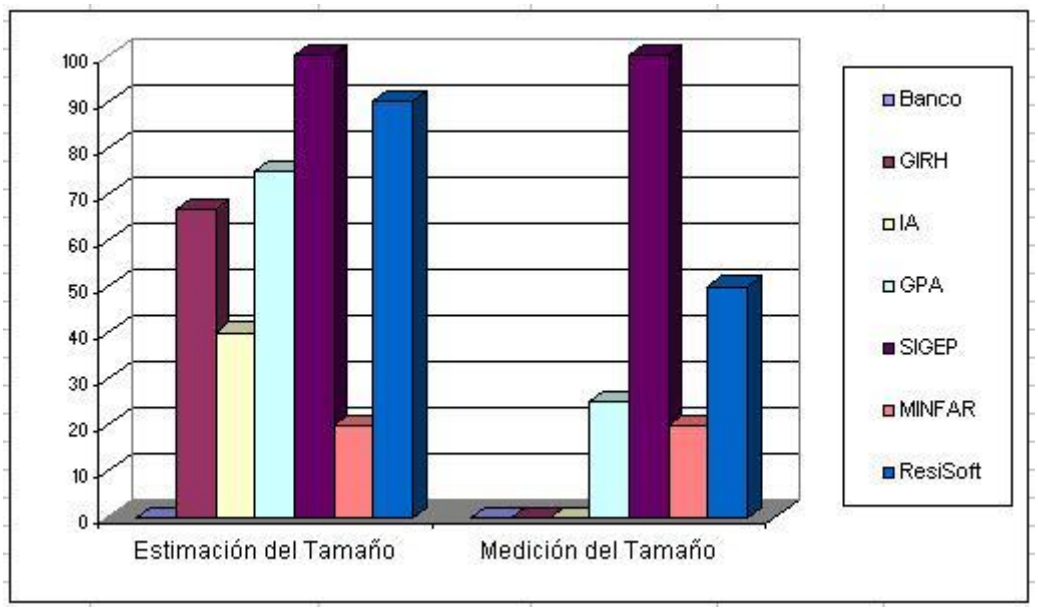

Figura 9: Estimación y Medición del Tamaño.

Además la encuesta realizada demostró con datos estadísticos que no existe concordancia en cuanto a la Planificación de Proyectos (**ver figura 10**), pues no se puede realizar una buena planificación si el nivel de todos los elementos que son necesarios para hacer una buena estimación no son concretos y reales como es el caso de que se realizan todos los reportes de pruebas pero no se realizan todas las pruebas, se estima el tamaño pero no se mide por lo que no se sabe el valor real del mismo, se planifican la actividades pero no se gestiona dicha planificación y el tiempo real no se mide.

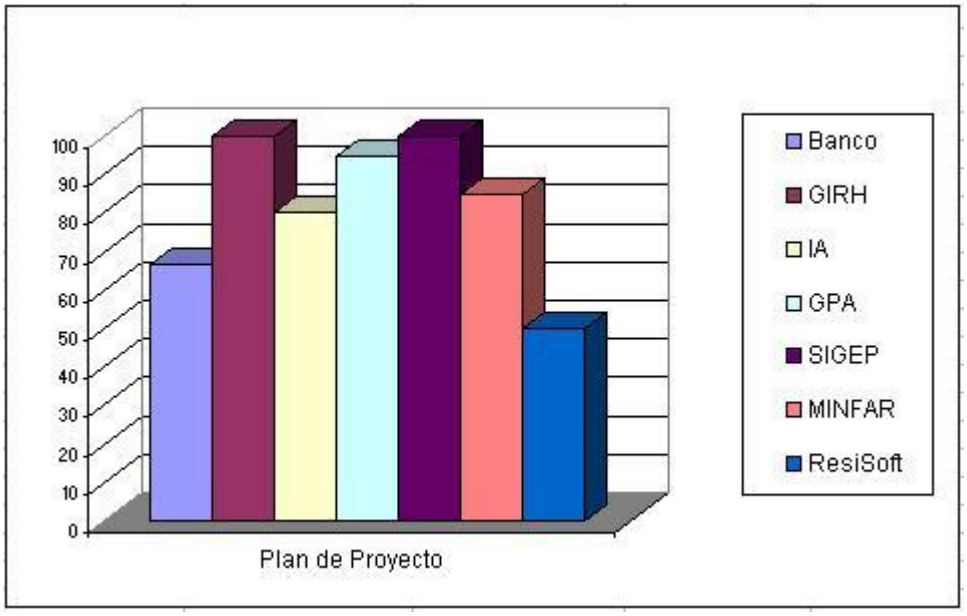

Figura 10: Plan de Proyecto.

TSPi propone 8 procesos que aplicados de manera cíclica permiten a los equipos de ingenieros de software alcanzar productos de alta calidad en el tiempo estimado. TSPi provee una guía en la que se presenta el objetivo, el criterio de entrada, las etapas (con las actividades que involucra cada una de ellas) y el criterio de salida de cada proceso. El criterio de entrada es todo aquello que necesita el proceso para llevarse a cabo y el criterio de salida es el producto o los productos generados por el proceso. Asimismo, la guía proporciona unas formas para el registro de todos los datos relevantes del proyecto. Estos datos sirven para evaluar el desempeño del equipo, de cada uno de sus integrantes y definir posibles mejoras al proceso de desarrollo.

Mientras que de estos 8 procesos hay algunos como la Definición de Requisitos, y la Implementación del Producto que están presentes en todos los proyectos con un 100% en cada caso, existes otra no menos importantes como son el Lanzamiento del Proyecto, la Estrategia de Desarrollo, el Plan del Proyecto y el Postmortem como se muestra en la **figura 11** que varían de un proyecto a otro hasta llegar en algunos proyectos a no existir lo cual influye de forma negativa en el rendimiento del equipo y por ende en la calidad del producto.

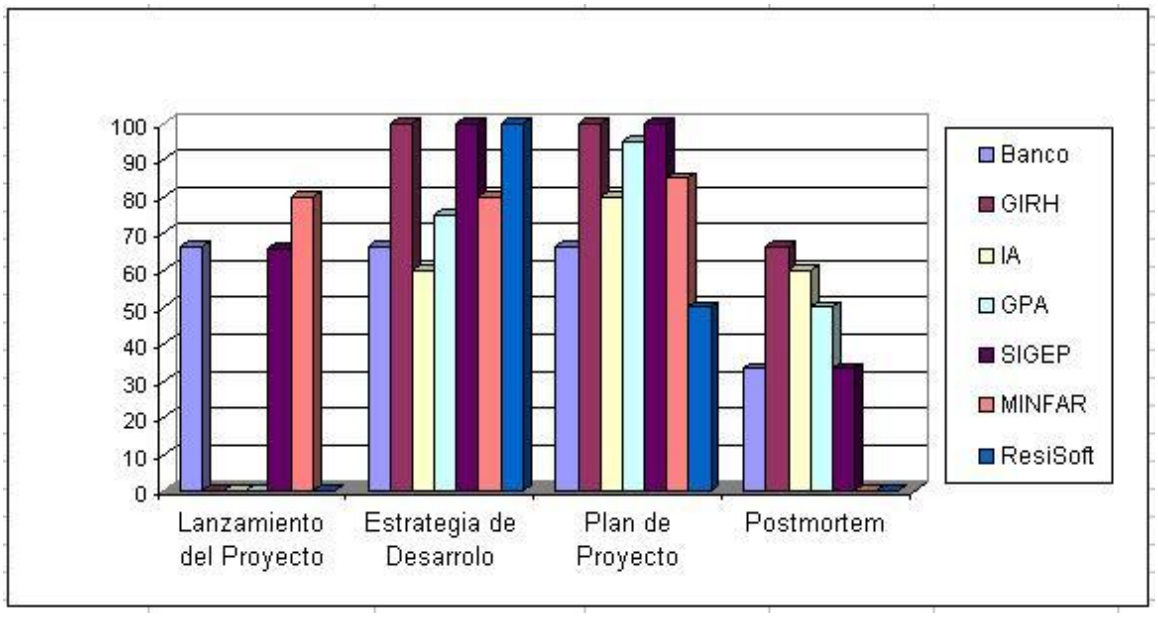

Figura 11: Procesos de TST presentes en los proyectos.

Para que un equipo funcione correctamente hay que asignar roles a cada miembro del equipo para que cada miembro sepa su responsabilidad y la metas que tiene que cumplir. TSP propone los siguientes roles: Líder de Equipo, Jefe de Desarrollo, Jefe de Planificación, Jefe del Proceso de Calidad y Jefe de Soporte. Aunque en los proyectos de la facultad la mayoría de estos roles están presentes como se muestra en la **tabla1** se da el caso de que el Líder de Equipo tiene que hacer las funciones de otros roles como el de Jefe de Planificación.

Tabla 1: Roles de TSP por proyectos.

	Banco	GIRH	IA	GPA	SIGEP	MINFAR	RESISOT
Líder de Equipo	**x**	**x**	**x**	**x**	**x**	**x**	**x**
Jefe de Desarrollo	**x**	**x**	**x**	**x**	**x**		**x**
Jefe de Planificación.	**x**	**x**	**x**	**x**	**x**		**x**
Jefe del Proceso de Calidad	**x**			**x**			**x**

Jefe de Soporte	**x**		**x**			**x**	

2.4 Propuesta de Aplicación de TSP y PSP.

PSP0 es el proceso de desarrollo en sí, incluyendo mediciones básicas que permiten identificar aspectos a mejorar y llevar un seguimiento del progreso alcanzado. Se registra el tiempo y los defectos. Para esto último se requiere definir un estándar de tipos de defectos o se puede tomar el propuesto por PSP (Ver anexo 1).

PSP0.1 se obtiene agregando al PSP0 la estandarización del código, medidas de tamaño y una propuesta para el mejoramiento del proceso. Es importante registrar los problemas encontrados en el proceso y sugerencias para mejorarlo.[24]

La Línea Base del Proceso Personal (PSP0 y PSP0.1), brinda una introducción al PSP y establece una base inicial de tamaños históricos, tiempo y datos de defecto. El principal objetivo de PSP0, es proporcionar un marco de trabajo para escribir su primer programa PSP y para obtener datos en su trabajo. Los datos de los primeros programas PSP brindan una línea base comparativa para determinar el impacto de los métodos PSP en su trabajo. En la **tabla 2** se muestra el proceso PSP0 de manera simplificada. Los guiones lo conducen a través de los pasos del proceso, los registros permiten guardar los datos del proceso y el resumen del plan brinda una forma conveniente de registrar y reportar los resultados.

Tabla 2: Línea Base de PSP0 y PSP0.1.

Paso	Fase	Descripción
1	Planificación	Se planifica el trabajo y se documenta el plan.
2	Diseño	Se diseña el programa.
3	Codificación	Se implementa el diseño.
4	Compilación	Se compila el programa, se reparan y registran los defectos encontrados.
5	Prueba	Se prueba el programa, se reparan y registran los defectos encontrados.
6	Postmortem	Se registra el tiempo actual, defectos y tamaño de datos en el plan

El PSP1 introduce la Estimación de Tamaño. El objetivo de PSP1 es entablar un procedimiento ordenado y repetido para la estimación de tamaño de software y como los demás niveles de PSP viene acompañado por un conjunto de guiones y formularios.[24]

El objetivo de la Estimación de Tamaño es hacer mejores planes que permitan tomar decisiones importantes antes de realizar el producto y tener una idea del tiempo y los recursos necesarios para su desarrollo.

Por todo lo antes expuesto y tomando en cuenta la necesidad de contar con un repositorio de información que contenga datos reales con vistas a analizar la información y así estimar, planificar, tomar decisiones, darle seguimiento y control a los proyectos productivos de la faculta se decidió en una primera etapa aplicar hasta el nivel de PSP1 a los proyectos con el objetivo de crear dicho repositorio que permitan realizar buenas estimaciones y obtener un producto con una mayor calidad en un futuro no muy lejano, ajustado lo más posible a la realidad.

2.4.1 Registro de Tiempo.

El principal elemento que mide PSP es el tiempo pues del tiempo que demore en hacerse un software dependerá el costo y los esfuerzos necesarios para realizarlo, por tanto lo primero que se debe hacer es **registrar el tiempo utilizado en cada actividad o fase** para obtener datos de cómo se trabaja realmente, la forma y el procedimiento utilizados para reunir los datos no es tan importante mientras que los datos sean exactos. Para la realización del mismo se debe utilizar el formato de Registro de Tiempo propuesto por PSP.

El registro de tiempos es necesario para:

- Saber cómo se distribuye el tiempo en la fase de desarrollo.
- Servirá para armar la base datos de tiempos que se utilizará para mejorar futuras planificaciones.

La lógica que rige la administración del tiempo es:

- Cómo se utiliza el tiempo esta semana será muy similar a la próxima.
- Para hacer un plan realista, se debe conocer cómo se utiliza el tiempo.
- Para comprobar que el plan es realista, se debe documentar la distribución del tiempo y ver qué se está haciendo.

- Para hacer un plan seguro, se debe identificar donde se falló y decidir cómo corregir esa situación.
- Cuando se desarrolla, se debe registrar el tiempo consumido en cada fase de desarrollo.
- Para administrar el tiempo, se debe elaborar el plan y cumplirlo.

El registro de tiempo contiene: (**Ver anexo 4**)

- El proyecto o los programas en los que se está trabajando.
- Las fases del proceso para las tareas.
- La fecha y el tiempo en que se inició y culminó el trabajo en la tarea.
- Cualquier interrupción de tiempo.
- La variación de tiempo trabajado en la tarea.
- Comentarios.

2.4.2 Registro de Defectos.

La calidad es el elemento más crítico en un software, el cual no se puede dejar para el final y tiene que estar desde la misma fase de inicio y un elemento crítico que contribuye a la mala calidad del software son los defectos por lo que se propone **registrar los datos de los defectos inyectados en cada actividad o fase** con el objetivo de mejorar la forma de encontrarlos y corregirlos utilizando estos datos. Para que un software cumpla con las necesidades de los usuarios debe funcionar y para que funcione hay que eliminar los defectos. Para la realización del mismo se debe utilizar el formato de Registro de Defecto propuesto por PSP.

El Registro de Defectos.

- Permite registrar cada defecto encontrado y corregido en el proceso de desarrollo. Servirá como información para elaborar el proceso de depuración (**Ver anexo 2**).
- El Registro del Tiempo del proceso de depuración de defectos, permite conocer la distribución de tiempos.

2.4.3 Resumen del Plan de Proyecto.

El Plan de Proyecto indica todas las actividades y el orden en el que deben realizarse pero este plan puede no estar ajustado a las necesidades reales del proyecto, por lo que se debe hacer **el Resumen del Plan de Proyecto** con el objetivo de obtener las estimaciones y los datos reales que conforman al proyecto en toda su amplitud para que al final de cada actividad o fase se realicen las comparaciones

necesarias y exista un histórico de todos los proyectos realizados. Esta actividad es esencial, ya que es un respaldo para cada proyecto que se desarrolla. En él se pueden encontrar los datos que serán útiles para el siguiente proyecto parecido que se desarrolle. Es importante que los datos se escriban con claridad y con precisión para que cada fase de desarrollo sirva para tener un margen de comparación con proyectos futuros. Para la realización del mismo se debe utilizar el formato de Resumen del Plan de Proyecto propuesto por PSP.

2.4.4 Medición del Tamaño.

Aunque PSP mide en líneas de código (LOC) en la facultad no existe una cultura creada para medir en LOC y los integrantes del proyecto no poseen experiencias como programadores, es decir, y a modo de ejemplo se puede notar que, en lo que alguien hace en 30 LOC, otro puede implementar también en 50, lo que conlleva a analizar entonces, que si se toma el tamaño en LOC, no se podrá contar con medidas fiables para estimar los tamaños de software y analizar en qué medida se puede estar avanzando sobre la planificación.

Se propone entonces partir de **medir el tamaño del software mediante Puntos de Casos de Uso** con el objetivo de obtener los valores reales del tamaño para compararlos con los valores de la estimación y que permitan hacer ajustes en la misma si es necesario y hacer mejores estimaciones en futuros proyectos. Es importante destacar que medir en LOC resultaría ser lo más óptimo.

2.4.5 Estimación del Tamaño.

Si se conoce el tamaño se pueden estimar los esfuerzos, el tiempo y el costo del software y como el tamaño se mide en Puntos de Casos de Uso se debe **estimar el tamaño del software mediante Puntos de Casos de Uso** con el objetivo de tomar decisiones importantes antes de llevarlo a cabo y así tener una idea general de los costos y el esfuerzo que se necesitarán para el desarrollo del software.

Como el PSP está enfocado a tareas como la predicción, la verificación de calidad y productividad, las cuales están diseccionadas como procesos individuales, ya que al principio solo se tiene apuntado hacia una meta, pero para que esto realmente funcione en un proceso grande, se necesita un conjunto de este tipo de actividades, lo cual lleva de la mano a la aplicación del TSPi. El TSPi es el desarrollo “real” del proceso, ya que es el modelo que permite ver cómo usar el PSP en un proyecto de desarrollo y cómo el líder y su equipo pueden desarrollar las mejores prácticas para aumentar y perfeccionar lo

que se hizo en el PSP por lo que se proponen los siguientes procesos de TSP para guiar a los equipos durante el desarrollo del producto para obtener una mayor calidad.

2.4.6 Lanzamiento del Proyecto.

Cuando se va a empezar un nuevo proyecto lo primero que se debe hacer es **el Lanzamiento del Proyecto** en el cual se debe crear el equipo de trabajo, determinar las relaciones de trabajo, los roles de los miembros, quién va a ocupar cada uno de ellos y se debe llegar a un acuerdo sobre los objetivos. Se deben fijar objetivos ambiciosos pero realistas, de tal manera que motiven a los integrantes del grupo. Para fijar objetivos con estas características, es importante contar con experiencias previas. Cuando no se cuente con esta experiencia, TSPi sugiere adoptar, como objetivos básicos, elaborar un producto de calidad, llevar a cabo un proyecto productivo y bien administrado, y terminar el proyecto a tiempo.

2.4.7 Estrategia de Desarrollo.

Otro proceso que se debe tener en cuenta por su gran importancia pues fija el camino a seguir hasta la terminación del producto es: **la Estrategia de Desarrollo**. En esta etapa se debe idear una estrategia para realizar el trabajo, crear un diseño conceptual del producto, y hacer una estimación preliminar del tamaño del producto y del tiempo de desarrollo. El tiempo estimado debe corresponder al tiempo disponible. De no ser así, se debe revisar y ajustar la estrategia hasta que los tiempos coincidan. TSPi propone un proceso cíclico en el que cada ciclo toma la versión del sistema generada en el ciclo anterior y produce una versión aumentada. La cantidad total de ciclos depende del tamaño del sistema y del tiempo disponible.

2.4.8 Plan de Proyecto.

La planificación da una idea del alcance, el esfuerzo y los costos del producto y qué se debe hacer y cuándo es por ello que otro de los 8 procesos propuesto por TST y que tiene gran importancia es **el Plan del Proyecto.** El Plan de Proyecto indica todas las actividades y el orden en el que deben realizarse[18]. Proporciona el marco y el contexto de trabajo. Una vez definido el plan, se puede trabajar de manera más eficiente, pues se sabe qué hacer y cuándo hacerlo. Las actividades planeadas entre los miembros del equipo deben ser distribuidas de manera equilibrada. Es decir, el trabajo asignado a cada miembro del equipo se debe poder realizar en el mismo período de tiempo. De

esta manera no se ocasionan esperas inútiles para algunos miembros, mientras otros están saturados de trabajo.

2.4.9 Posmorten.

Una buena calidad solo se puede lograr con una mejora continua es por ello que en el proceso de **el Postmortem** se debe revisar todo el trabajo hecho por los ingenieros y todos los datos recolectados durante los procesos previos. Posteriormente se debe hacer un análisis para identificar los puntos en los cuales se puede mejorar el proceso, lo cual provee un medio para aprender y mejorar. Se debe comparar lo planeado con lo hecho. Además, se deben detectar oportunidades para mejorar y decidir cambios en las prácticas para el siguiente ciclo o proyecto.

Para determinar en qué áreas se puede mejorar el proceso se deben realizar revisiones. Algunas de las más importantes son:

- Revisión de calidad. Se debe comparar la calidad planeada a nivel personal y a nivel de equipo con la calidad producida. Deben plantearse las siguientes tres preguntas. ¿Qué se puede aprender de esta experiencia? ¿Qué objetivos se pueden plantear para el siguiente desarrollo? ¿Dónde se podría modificar al proceso para mejorarlo?
- Evaluación de roles. Se deben evaluar los diferentes roles del equipo. Deben plantearse las siguientes tres preguntas. ¿Qué sirvió? ¿Dónde hubo problemas? ¿Qué se puede mejorar?

Como se expuso anteriormente para que un equipo funcione correctamente es necesario asignar roles a cada miembro del equipo por lo que se proponen los siguientes roles paras los equipos de desarrollo de los proyectos de la facultad 4.

2.4.10 Líder de Equipo.

El **Líder de Equipo** debe lograr un equipo efectivo. El equipo debe culminar todo su trabajo planificado, cada uno debe participar en las reuniones del equipo y todos deben trabajar duro por terminar las tareas en tiempo. Otra forma de medir el trabajo del Líder es mediante las evaluaciones colectivas del equipo. Debe formar y mantener un equipo efectivo. Motivar a todos los miembros para trabajar agresivamente en el proyecto. Resolver todos los problemas que los miembros del equipo le planteen y debe ser bueno como facilitador en las reuniones del equipo.

2.4.11 Jefe de Desarrollo.

Otro de los roles más importantes es **el Jefe de Desarrollo**. El cual debe establecer como la meta principal la construcción de un producto funcional y de alta calidad. Lograr que el equipo realice un producto superior. Usar completamente los conocimientos y habilidades de los miembros del equipo.

2.4.11 Jefe de Planificación.

El **Jefe de Planificación** por su gran contenido de trabajo y su gran responsabilidad no debe ser el mismo Líder del Equipo aunque debe tener una gran relación con el mismo. Debe guiar al equipo en la producción de un plan detallado y chequear con precisión el progreso contra este plan. Debe producir un plan completo, preciso y veraz para el equipo y para cada miembro del mismo. Reportar exactamente la situación del equipo cada semana.

2.4.13 Jefe de Proceso y la Calidad.

Como la calidad en una parte importante del producto se debe seleccionar un **Jefe de Proceso y Calidad** el cual debe tener como la meta principal asegurarse de que el equipo use de manera adecuada el TSPi para construir un producto libre de defecto. Debe asegurarse de que todos lo miembros del equipo reporten exactamente y adecuadamente el uso de los datos del proceso TSPi, de que el equipo fielmente siga el TSPi y realice un producto de calidad, de que todas las inspecciones del equipo sean adecuadamente moderadas y registradas y todas las reuniones del equipo sean exactamente documentadas y los reportes se coloquen en el libro del proyecto.

2.4.14 Jefe de Soporte.

No se puede producir un software si no se cuenta con los medios y las herramientas necesarias para realizar su trabajo es por ello que el **Jefe de Soporte** debe asegurarse que el proyecto es adecuadamente soportado y controlado, que el equipo este dotado con las herramientas y métodos para soportar su trabajo, de que todos los cambios no autorizados son hechos sobre productos de la línea base, de que todos los problemas y riesgos del equipo sean registrados en el sistema de control de problemas y reportados cada semana y que el equipo enfrente sus metas de reutilización para el ciclo de desarrollo.

El éxito de TSPi, se basa en que la mayoría de los sistemas que se desarrollan actualmente requieren del trabajo de equipos, no de programadores individuales. Además, los equipos en los

cuales cada miembro cuenta con buenas prácticas de trabajo, y trabajan cooperativa y eficazmente juntos, producen resultados de alta calidad. Cuando un equipo encuentra condiciones adecuadas, sus miembros realizan un trabajo superior, son más productivos y disfrutan su trabajo.

2.4.15 Métricas.

Para complementar la propuesta anterior se debe utilizar las siguientes métricas para realizar la medición de los proyectos y poder conformar el repositorio de información con vista a poder analizar la información y así estimar, planificar, tomar decisiones, darle seguimiento y control a los proyectos productivos de la facultad.

Tabla 3: Métricas Bases.

Métrica	Descripción
Tiempo	Tiempo real dedicado por la persona en cada una de las tareas ejecutadas como parte del proyecto. Se expresa en minutos
Tiempo Estimado	Tiempo estimado por la persona en cada una de las tareas que debe ejecutar como parte del proyecto. Se expresa en minutos
Tamaño	Tamaño estimado Se expresa en Casos de Uso.
Tamaño Estimado	Tamaño real Se expresa en Casos de Uso.
Defectos Removidos Estimados	Estimación de los defectos a eliminar
Defectos Removidos	Defectos eliminados reales
Defectos Inyectados Estimados	Estimación de defectos a inyectar
Defectos Inyectados	Defectos inyectados reales
Total de defectos inyectados por cada Caso de Uso.	El valor ideal para esta métrica es 0.
Defectos inyectados en la fase de diseño por cada Caso de Uso.	El valor ideal para esta métrica es 0.

Defectos inyectados en la fase de código por cada Caso de Uso	El valor ideal para esta métrica es 0.
Defectos removidos en la fase de revisión de diseño por cada Caso de Uso.	Lo ideal es que el valor de esta métrica tienda a aumentar pero se recomienda que no se analice de forma aislada sino con respecto a los valores de defectos inyectados en la propia fase y en fases anteriores, ya que se desea, que en el desarrollo de esta fase se eliminen la mayor cantidad posible de los defectos inyectados en las fases que le anteceden y en la propia fase.
Defectos removidos en la fase de revisión de código por cada Caso de Uso.	Lo ideal es que el valor de esta métrica tienda a aumentar pero se recomienda que no se analice de forma aislada sino con respecto a los valores de defectos inyectados en la propia fase y en fases anteriores, ya que se desea, que en el desarrollo de esta fase se eliminen la mayor cantidad posible de los defectos inyectados en las fases que le anteceden y en la propia fase.
Defectos removidos en la fase de compilación por cada Caso de Uso.	El valor ideal para esta métrica es 0, si los defectos inyectados en codificación no persisten hasta fases posteriores.
Defectos removidos en la fase de prueba por cada Casos de Uso.	Para esta métrica el valor ideal es 0, si no persisten defectos al concluir esta fase. Lo deseado es que todos los errores se hubieran sido detectados y eliminados antes de llegar a esta fase.
Defectos inyectados para una fase	Este tipo de gráfico no posee una métrica directamente asociada. Pero si permite valorar en que fases del proceso se están inyectando la mayoría de los defectos y cual requiere más atención y revisión
Defectos removidos para una fase	Este tipo de gráfico no posee una métrica directamente asociada. Pero si permite valorar en que fases del proceso se están eliminando la mayoría de los defectos. Para una mejor comprensión de este gráfico debe considerarse el análisis realizado sobre el gráfico anterior y el de "Defectos removidos en prueba".
Defectos removidos en la fase de compilación	Es de esperarse que en la medida que disminuyan los defectos en la fase de compilación disminuyan en la fase de prueba.

contra defectos removidos en la fase de prueba, por cada Caso de Uso.	
Defectos inyectados por fases a la fecha	Permite saber cuales son las fases de mayores problemas en desarrollo de un proyecto en cuanto a la inyección de defectos.
Defectos removidos por fases a la fecha	Permite saber cuales son las fases de mayores problemas en desarrollo de un proyecto en cuanto a la eliminación de defectos.

Combinando las métricas bases anteriores se obtuvieron las métricas derivadas siguientes.

Tabla 3: Métricas Derivadas.

Métricas	Fórmula	Descripción
Razón del Costo de Planificación(RCP)	Tiempo Estimado / Tiempo	Métrica que indica la calidad de la planificación. Indica el grado en que se están cumpliendo las metas planificadas. Idealmente el RCP debe ser 1. Preferiblemente ligeramente mayor que 1.0, para contar con una holgura de tiempo moderada para posibles riesgos. Si es menor que 1.0 es que se está gastando más tiempo que lo planificado en los proyectos. Si son substancialmente mayores que 1 los planes están siendo muy conservadores.
Error Estimando Tiempo	(Tiempo - Tiempo Estimado) / Tiempo Estimado	Permite apreciar el margen de error en la estimación de tiempo. Esta métrica constituye un indicador de la calidad en la estimación de tiempo. Entre más cercano a 0 sea este valor, mejor será la estimación de tiempo. Valores mayores que 0 indican que el tiempo real está siendo mayor que el tiempo estimado y nuestra estimación está siendo demasiado irreal. Valores menores que 0 indican que la estimación está siendo muy conservadora.

Densidad Defectos Estimada	Defectos Removidos Estimados / Tamaño Estimado	Revela la cantidad estimada de defectos que se eliminan por líneas de código. La tendencia en esta métrica es ir disminuyendo paulatinamente.
Densidad Defectos	Defectos Removidos / Tamaño	Revela la cantidad real de defectos que se eliminan por Casos de Uso. La tendencia en esta métrica es ir disminuyendo paulatinamente
Estimación de defectos inyectados por horas	60*Defectos Inyectados Estimados / Tamaño Estimado	Representa la cantidad estimada de defectos a inyectar por hora en el desarrollo de un proyecto. Esta métrica también puede ser evaluada a nivel de fases. Donde se tendrían en cuenta la estimación de defectos a inyectar para esa fase y su tiempo estimado. El objetivo o valor ideal para esta métrica es 0, La tendencia debe ser a ir disminuyendo gradualmente.
Defectos inyectados por horas	60* Defectos Inyectados / [Tamaño]	Representa la cantidad de defectos inyectados por hora en el desarrollo de un proyecto. Esta métrica también puede ser evaluada a nivel de fases. Donde se tendrían en cuenta los defectos a inyectar para esa fase y su tiempo. El objetivo o valor ideal para esta métrica es 0, La tendencia debe ser a ir disminuyendo gradualmente.
Defectos removidos por hora	Defectos removidos/h = 60*Defectos removidos en <fase>/tiempo de <fase>	El valor ideal para esta métrica en cada una de las fases es 0 siempre que se cumpla que no existen defectos inyectados.
Influencia en Defectos removidos	DRL se calcula como: (Defectos/horas) en la <fase> /	Esta métrica debe ser más alta para las fases de revisión que para el resto de las fases.

	(defectos/horas) en prueba de unidad.	
Cobertura de las pruebas	X = A / B A –Número de casos de pruebas que han sido realmente ejecutados, y que representan el escenario de operación durante las pruebas. B – Número de casos de pruebas a ejecutar requeridos para cubrir los requisitos	Esta métrica responde a cuántos casos de pruebas requeridos han sido ejecutados durante las pruebas. Contar el número de casos de pruebas que han sido ejecutados durante las pruebas y se deben comparar con el número de casos de pruebas requeridos para obtener una adecuada cobertura de pruebas. 0 <= X <= 1 Mientras más cercano al 1, mejor cobertura.
Madurez de las pruebas	X=A / B A – Número de casos de pruebas que han obtenido un resultado satisfactorio al ser ejecutados o durante su operación. B – Número de casos de pruebas a ejecutar para cubrir los requisitos	Esta métrica responde a si está bien probado el producto. Contar el número de casos de pruebas que han obtenido un resultado satisfactorio de los casos realmente ejecutados y se deben comparar con el número total de casos de pruebas requeridos para cubrir los requisitos. 0 <= X <= 1 Mientras más cercano al 1, mejor.

2.5 Conclusiones.

Después de haber analizado los resultados de la encuesta y teniendo en cuenta la necesidad de los proyectos por mejorar su calidad se elaboró la propuesta anterior con el objetivo de obtener datos reales que permitan hacer una mejor planificación la cual será sometida a su validación y aprobación

mediante el método del Juicio del Experto, dicho procedimiento se explica mas detalladamente en el próximo capítulo.

CAPÍTULO 3: EVALUACIÓN TÉCNICA

3.1 Introducción.

Para aceptar y validar la propuesta del presente trabajo se conformó un Panel de Expertos que emitió su criterio. Este Panel estuvo compuesto por jefes de diferentes proyectos en la Universidad de Ciencias Informáticas. Este proceso de validación se llevó a cabo por el Método Delphi, que es una técnica de investigación social que tiene como objeto la obtención de una opinión grupal a partir de un grupo de expertos.

3.1 Proceso de selección de expertos.

En la planificación del criterio de expertos se parte de la concepción inicial del problema y la selección de los expertos como pasos previos fundamentales para la aplicación del criterio.

Los expertos seleccionados son un grupo de personas con un gran conocimiento sobre el tema, que pueden emitir un criterio concluyente de cualquier problema de la estimación y la planificación de proyectos y emitir valoraciones importantes con un alto nivel de conocimiento. En este proceso de selección se tuvieron en cuenta pasos que se justifican a continuación.

3.1.1 Selección de expertos que conforman el panel.

Para escoger el grupo de expertos que integraron el panel se tuvo en cuenta los conocimientos que tienen acerca del tema. El listado de expertos se realizó con personas que están vinculadas directamente a la producción de software en la UCI. Todas las personas seleccionadas para integrar el panel cuentan con los conocimientos necesarios para emitir una. Se escogieron un total de cinco expertos entre los cuales se encuentran Jefes de Planificación y Líderes de Proyectos (ver anexo 6).

Los expertos seleccionados son personas serias, honestas, sinceras, responsables, creativas y presentas gran capacidad de análisis. Estas cualidades han permitido que las opiniones brindadas sean confiables y válidas para valorar el trabajo realizado.

3.1.2 Confirmación de la participación de los expertos.

Una vez confeccionado el panel se invitó a cada uno de ellos de forma personal para que participaran en el proceso de validación y aceptación de la propuesta. Se les detalló de una forma clara y precisa los objetivos de este proceso explicando en que consistía el trabajo en general, la propuesta a evaluar, la realización del cuestionario, así como el plazo de entrega.

Una vez recibida la respuesta positiva, se estableció el listado final de los expertos, informando a cada especialista su inclusión en el proceso a evaluar y las instrucciones necesarias para contestar las preguntas. De esta forma culmina el proceso de selección, logrando la participación de los cinco expertos que conformarán el panel.

3.2 Elaboración del cuestionario.

Para la presentación del cuestionario se tuvo en cuenta realizar preguntas con un enfoque investigativo y que se centraran principalmente en los principios básicos que debe cumplir la propuesta presentada, además de permitir que las respuestas fueran abiertas, y en todos lo casos con posibilidad de emitir su criterio personal en cuanto a su experiencia en el proyecto al que pertenece.

Para el procesamiento y análisis de la información se tiene en cuenta el tipo de pregunta si es abierta o cerrada y se valora desde el punto de vista cualitativo y cuantitativo respectivamente. El análisis cualitativo es fundamentalmente para las preguntas de tipo abiertas, se leen detalladamente cada una de las respuestas y se resumen los elementos más comunes y esenciales. Lo cuantitativo en general es para las preguntas que son de tipo cerrada están asociadas a valores numéricos.

El cuestionario establece catorce preguntas, que permitieron ver la posibilidad real de que pueda ser aplicada la propuesta definida en este trabajo, según las características actuales de la UCI. Además de brindar su efectividad en caso de ser establecida y una evaluación general del proceso, teniendo en cuenta una serie de requisitos establecidos en la pregunta número dos (Ver Anexo 7)

.

Se estableció una escala del uno al cinco, siendo uno el valor de menor importancia y cinco el de mayor. Estas preguntas proporcionan una mayor riqueza en las respuestas que son brindadas por los expertos debido a que se les dio la posibilidad que expresaran su opinión y dieran sugerencias al trabajo propuesto. (Ver Anexo 7)

En este proceso los expertos recibieron de forma personal los cuestionarios y se les pidió que completaran esta tarea en un mínimo de tiempo para analizar las respuestas y permitirle que hicieran preguntas en caso que hubiesen surgido dudas referentes al cuestionario.

3.3 Resultados de la evaluación.

La **tabla 5** muestra los criterios emitidos a la pregunta número dos por los diferentes expertos que conforman el panel.

Tabla 4 Valores emitidos por los expertos.

	Criterios													
Expertos	**a)**	**b)**	**c)**	**d)**	**e)**	**f)**	**g)**	**h)**	**i)**	**j)**	**k)**	**l)**	**m)**	**n)**
Experto 1	5	5	5	3	5	4	4	5	5	5	5	5	5	5
Experto 2	5	3	5	3	5	5	4	5	5	5	5	5	5	5
Experto 3	4	4	3	3	5	5	4	5	4	3	4	5	4	4
Experto 4	5	5	4	4	5	3	4	4	5	5	5	4	5	5
Experto 5	5	5	5	5	5	2	4	5	4	5	5	5	5	4

La suma de rangos que se obtiene a partir de los valores ya definidos para cada pregunta se denota por Sj y se representa por

$S_j = \sum_{i=1}^{m}$ (Rij el rango asociado a la evaluación del experto “i” a la pregunta “j”).

El valor de Sj es utilizado para comparar la importancia de diferentes respuestas, de modo que un mayor valor significará una mayor importancia. Además se empleará para buscar el coeficiente de concordancia.

Se define también la media de la suma de rangos de cada pregunta “j” denotada por $\overline{S}$, la que se calcula según la fórmula:

$$\overline{S} = \frac{\sum_{j=1}^{n} S_j}{n} = \frac{m(n+1)}{2} \qquad (1)$$

Con los valores de Sj y $\overline{S}$ S es posible calcular el coeficiente de concordancia Kendall (denotado por k):

$$k = \frac{12\sum_{j=1}^{n}(S_j - \overline{S})^2}{m^2(n^3 - n) - m\sum_{i=1}^{m} T_i} = \frac{12\sum_{j=1}^{n} S_j^2 - n\overline{S}^2}{m^2(n^3 - n) - m\sum_{i=1}^{m} T_i} \qquad (2)$$

En la fórmula anterior Ti representa el resultado de los rangos iguales (llamados también ligas), que ofreció el experto “i” para las preguntas, que se calcula como sigue:

$$T_i = \frac{\sum_{i=1}^{l}(t^3 - t)}{12}$$, donde l y t representan los siguientes aspectos:

l: número de grupos con rangos iguales para el experto “i”.

t: número de observaciones dentro de cada uno de los grupos para el experto “i”.

Los valores del coeficiente “k” deben oscilar entre 0 y 1 (0 < k < 1), si k alcanza el valor uno (k = 1) entonces existe una concordancia total de criterios, mientras mayor sea el valor de k, es decir, cuanto más se acerque a uno, mayor será la concordancia entre los expertos.

Luego se aplica la Prueba de Significación de Hipótesis, planteándose la hipótesis nula y la alternativa de la siguiente forma:

H0: los expertos no están de acuerdo con la propuesta, k= 0

H1: los expertos si están de acuerdo con la propuesta, k □ 0

Se determina Chi-cuadrado calculado como:

$$X^2_{cal} = m\,(n-1)\,k \qquad (3)$$

Por otra parte, se busca el Chi-cuadrado tabulado en la tabla del percentil de la distribución Chicuadrado con un nivel de significación yn - 1 grados de libertad, representado por: $X^2_{tab} = X^2_{\alpha;n-1}$

Se compara X^2_{cal} y X^2_{tab}, si se obtiene que $X^2_{cal} > X^2_{tab}$ entonces se rechaza H0 y se considera válida la hipótesis alternativa H1.

3.3.1 Resultados obtenidos

Después de aplicar la fórmula de la suma de rangos que se obtiene a partir de los valores que les fueron otorgados a cada pregunta por parte de los expertos se obtuvieron los siguientes valores que se muestran el la **tabla 5**:

Tabla 5 Valores de suma de rangos.

Preguntas	**a)**	**b)**	**c)**	**d)**	**e)**	**f)**	**g)**	**h)**	**i)**	**j)**	**k)**	**l)**	**m)**	**n)**
Sj	24	22	22	18	25	19	20	24	23	23	24	24	24	23

La comparación de los valores obtenidos para Sj conduce a que el mayor resultado obtenido corresponde a la pregunta e), esta es la pregunta que tuvo mayor importancia por parte del panel de expertos y de esta manera en orden descendente se determina la importancia que le fue otorgada a los incisos por los expertos.

Ahora se calcula la media $\overline{S}$ de la suma de rangos de cada inciso "j" por la fórmula (1) y se obtiene que para m = 5 (números de expertos) y n = 14 (número de cuestiones):

$$\overline{S} = \frac{5 * (14+1)}{2} = 37.5$$

Con el valor obtenido de Sj y $\overline{S}$, y los cálculos parciales auxiliares que siguen, se determina el coeficiente "k" de Kendall aplicando la fórmula (2) de la siguiente forma:

$$\sum_{j=1}^{n} S_j = 576+484+484+324+400+361+625+576+529+529+576+576+576+529 = 7145$$

$$\sum_{i=1}^{m} T_i = 110.5+110.5+35+52+84.5 = 392.5$$

El coeficiente k resulta:

$$k = \frac{12 * 7145 - 14 * 1406.25}{25 * (2744 - 14) - 5 * 392.5} = \frac{12 * 7145 - 19687.5}{68250 - 1962.5} = 0.99$$

Se obtiene que 0 < k < 1, estando k bastante cerca de 1 por tanto es aceptable, por lo que están básicamente de acuerdo los expertos.

Para la Prueba de Significación de Hipótesis se calcula X^2_{cal} por la fórmula (3) como sigue:

$$X^2_{cal} = 5 * (14 - 1) * 0.99 = 64.35$$

Buscando en la tabla del percentil de la distribución Chi-cuadrado con un nivel de significación α = 0,05 y 6 grados de libertad el X^2_{tab} obteniendo $X^2_{0.05:6} = 12, 592$.

Como 64.35 > 12,592 entonces se rechaza la hipótesis nula (H0) de que los expertos no están de acuerdo con la propuesta y se considera válida la hipótesis alternativa (H1) de que los expertos sí están de acuerdo con la propuesta.

Con lo visto hasta el momento durante todo el proceso de validación por parte de los expertos y analizando los resultados que arrojaron sus respuestas en los cuestionarios que les fueron aplicados el panel de expertos estuvo de acuerdo con la propuesta planteada.

3.4 Conclusiones.

Los resultados obtenidos en el criterio de expertos permiten arribar a la siguiente conclusión:

Con la aplicación de la propuesta se obtienen datos que permitirán realizar mejores estimaciones logrando una mejor planificación, obteniendo como producto final un software con una mayor calidad, libre de defectos, entregado en la fecha pactada con el cliente y que este a la altura de los requisitos de calidad de la UCI y las necesidades del cliente.

CONCLUSIONES GENERALES

Una vez estudiado y analizado los niveles del PSP y TSP y las métricas que se utilizan en el PSP y TSP y basado en los resultados obtenidos en la encuesta realizada en los proyectos productivos de la facultad 4, se arriba a las siguientes conclusiones:

- Se demostró que en todos los proyectos, se realizan mediciones, pero los datos de estas mediciones no se guardan en un repositorio de información central, con vistas a analizar la información y así estimar, planificar, tomar decisiones, darle seguimiento y control a los proyectos productivos de la facultad.
- Se demostró que en dichos proyectos se aplican elementos de PSP y TSP pero los mismos no se aplican de forma organizada y controlada.
- Se realizó una propuesta de medición para los proyectos de la facultad 4 basados en PSP y TSP.

RECOMENDACIONES

Al finalizar este trabajo quedan algunas recomendaciones que pueden servir de punto de partida para una posterior mejora de la propuesta presentada.

- Realizar un estudio a los proyectos productivos de las demás facultades para ver el grado de utilización de PSP y TST en los mismos y la aplicabilidad de la propuesta.

- Continuar con las investigaciones para aplicar niveles superiores de PSP y TSP y añadir nuevos elementos a la propuesta, que permitan obtener mejoras en futuras versiones, logrando adecuarla cada vez más a las necesidades de los proyectos.

REFERENCIAS BIBLIOGRÁFICAS

1. Productivo, C.e.d.t.d.a.c.p.d.p., *PSP y TSP*. 2007.
2. Pressman, R.S., *Ingeniería de Software. Un enfoque práctico.* 2001.
3. Pressman, R.S., *Ingeniería del Software. Un Enfoque Práctico. Capítulo 5: Planificación de proyectos de software.* Quinta Edición ed. 2002.
4. Daniele, M., *Métricas de Software.* 2007.
5. Engineers, T.I.o.E.a.E., *IEEE Standard Glossary of Software Engineering Terminology, IEEE Std. 610-1990* Vol. IEEE Standards Software Engineering, Volume 1. 1999.
6. Pressman, R.S., *Ingeniería del Software. Un Enfoque Práctico. Planificación de proyectos de software*. Cuarta Edición ed. 1998.
7. Grupo de Calidad del Software, I.y.G., *Calidad y Gestión del Software.* 2007-2008.
8. deMarco, T., *CONTROLLING SOFTWARE PROJECTS: MANAGEMENT, MEASUREMENT AND ESTIMATION* 1982.
9. CAPUCHINO, A.M.M.S., *"Control y gestión de proyectos software*. Vol. Unidad 4: Estimación de Proyectos Software". 1996.
10. Pressman, R.S., *Ingeniería del Software. Un Enfoque Práctico. Capítulo 4: Proceso del Software y Métricas del Proyecto*. Quinta Edición. ed. 2002.
11. ESCORIAL, J.S., *"Calidad de Software: Medidas del Proceso".* 2006.
12. HERNÁNDEZ, S.E.B., *"Métricas de estimación de tamaño: Puntos de Caso de Uso".* 2002.
13. WIKIPEDIA1. *"The Free Enciclopedia"*. 2006.
14. Brito, D.R. and H.F. Díaz, *Análisis del Método de Estimación empleado para el desarrollo del proyecto SIGEP*. 2007, Univercidad de las Ciencias Informáticas (UCI): Ciudad de La Habana. p. 89.
15. ZUSE, H., *"History of Software Measurement".* 1995.
16. Rúa, E.H.C., *Mejora de la calidad del software en el entorno de las microempresas del TI.* 2006-2007.
17. Agustín, G.C., *Para Comprender el Modelo de Madurez de Capacidad del Software.* 1998.
18. Buemo, A.W.R.y.S.G., *Ingeniería de software: el proceso para el desarrollo de software. Capítulo 12*. 2007.
19. Humphrey, W.S., *Introducción al PSP y TSP.* 2005.
20. Luna, D.A.A., *"Personal Software Process and Team Software Process".* Instituto Tecnológico y de Estudios Superiores de Monterrey. Campus Monterrey, 2006.
21. Humphrey, W.S., *"PSP. A Self – Improvement Process for Software Engineers"*. 2005.

22. Informáticas, U.d.l.C., *Gestión de Software: MANUAL DEL ESTUDIANTE.* Curso 2007-2008.
23. Humphrey, N., *Personal Software Process for Module-Level Development*. 1995.
24. Humphrey, W.S., *A discipline for software engineering*. SEI Series in Software Engineering. 2004.

BIBLIOGRAFÍA

1. Jacobson, Booch, Rumbaugh, Addison – Wesley / Object Technology Series, “El Proceso Unificado de Software”, 2000. Capítulo 12 pp 303-326.
2. Watts S. Humphrey, “PSP. A Self – Improvement Process for Software Engineers”, 2005.
3. Watts S. Humphrey, “Introducción al Proceso Software Personal”, 2001.
4. Tele formación/Bibliografía/Temas/PSP/PSP_for_Eng_Student_V4.1
5. U de Talca, “Estimación de Costos y Esfuerzo”, [Disponible en: http://ghostamd.googlepages.com/Estimacionesfuerzo.pdf]
6. Tom deMarco, “Controlling Software Projects: Management, Measurement and Estimation”, 1982.
7. Humphrey, Nicholas, “Personal Software Process for Module-Level Development”, 1995, [Disponible en: www.sei.cmu.edu/str/descriptions/psp_body.html]
8. ZUSE. H, “History of Software Measurement”, 1995.
9. CAPUCHINO, A. M. M. S, “Control y gestión de proyectos software”, 1996, Unidad 4: “Estimación de Proyectos Software".
10. Gonzalo Cuevas Agustín, “Para Comprender el Modelo de Madurez de Capacidad del Software”, 1998.
11. Roger S. Presuman, “Ingeniería del Software. Un Enfoque Práctico. Planificación de proyectos de software”, 1998, Cuarta Edición.
12. The Institute of Electrical and Electronics Engineers, “IEEE Standard Glossary of Software Engineering Terminology, IEEE Std. 610-1990”, 1999, IEEE Standards Software Engineering, Volume 1
13. Watts S. Humphrey, “Introducción al Proceso Software Personal”, 2001.
14. Roger S. Presuman, “Ingeniería de Software. Un enfoque práctico”, 2001.
15. HERNÁNDEZ, S. E. B, "Métricas de estimación de tamaño: Puntos de Caso de Uso", 2002.
16. Roger S. Presuman, “Ingeniería del Software. Un Enfoque Práctico.Capítulo 5: Planificación de proyectos de software”, 2002, Quinta Edición.
17. Roger S. Presuman, “Ingeniería del Software. Un Enfoque Práctico. Capítulo 4: Proceso del Software y Métricas del Proyecto”, 2002, Quinta Edición.
18. Hichem Labdelaoui, “Métodos de Estimación de Tamaño Funcional Software Aplicación a Enfoques de desarrollo”, 2003.
19. Humphrey, W.S, “A discipline for software engineering”, 2004, SEI Series in Software Engineering.

20. Watts S. Humphrey, “Introducción al PSP y TSP”, 2005.
21. Watts S. Humphrey, “PSP. A Self – Improvement Process for Software Engineers”, 2005.
22. ESCORIAL, J. S, “Calidad de Software: Medidas del Proceso”, 2006.
23. Dennia A. Aguilar Luna, “Personal Software Process and Team Software Process”, 2006, Instituto Tecnológico y de Estudios Superiores de Monterrey, Campus Monterrey.
24. WIKIPEDIA1, "The Free Enciclopedia", 2006.
25. Edgar Henrry Caballero Rúa, “Mejora de la calidad del software en el entorno de las microempresas del TI.”, 2006-2007, Resumen del trabajo tutelado curso 2006-2007.
26. Dayami Rodríguez Brito, Haydee Fernández Díaz, “Análisis del Método de Estimación empleado para el desarrollo del proyecto SIGEP”, 2007, Ciudad de La Habana, Universidad de las Ciencias Informáticas (UCI).
27. Alfredo Weitzenfeld Ridel y Silvia Guardati Buemo, “Ingeniería de software: el proceso para el desarrollo de software. Capítulo 12”, 2007.
28. Pedro Concepción, “Planificación de Proyectos de Software”, 2007, [Disponible en: http://www.getec.etsit.upm.es/articulos/gproyectos/art4.htm]
29. Marcela Daniele, “Métricas de Software”, 2007, [Disponible en: http://dc.exa.unrc.edu.ar/nuevodc/materias/ingenieria/2007/TEORIA/TEORIA%202-Metricas-Estimacion-2007.pdf]
30. Cybertécnica: empresa de tecnología dedicada al constante perfeccionamiento del proceso productivo, PSP y TSP, 2007,[Disponible en: http://www.cybertecnica.com/section23_es.htm]
31. Yeleny Zulueta Véliz, “Introducción de técnicas del Personal Software Process desde los primeros años en la formación del ingeniero informático”, 2007, [Disponible en: http://www.inf.udec.cl/revista/]
32. Grupo de Calidad del Software, Ingeniería y Gestión, “Calidad y Gestión del Software”, 2007-2008, [Disponible en: http://www.calidaddelsoftware.com/]
33. Universidad de las Ciencias Informáticas, “Gestión de Software: MANUAL DEL ESTUDIANTE”, curso 2007-2008.

GLOSARIO DE TÉRMINOS

Ámbito del proyecto: Se definen los objetivos del proyecto, identifica funciones primordiales que debe llevar a cabo el software e intenta limitar esas funciones de manera cuantitativa.

Calidad de software: Concordancia con los requisitos funcionales y de rendimiento explícitamente establecidos, con los estándares de desarrollo explícitamente documentados, y con las características implícitas que se espera de todo software desarrollado profesionalmente.

Defecto: Errores introducidos al software en toda su fase de desarrollo.

Estándar: Que posee el tamaño, la forma o cualquier otra característica que sigue al modelo. Se aplica a lo que se produce en serie. Que sigue una tendencia muy extendida. Aquello que se considera modelo.

Experto: Persona que aporta conocimientos o experiencia específica con respecto a una organización, proceso, actividad o materia que se vaya a auditar.

Gestión: Actividades coordinadas para dirigir y controlar una organización.

Método heurístico: Método o procedimiento mediante el cual se puede deducir o inducir la verdad.

Modelo empírico: Modelo de regresión que relaciona esfuerzo con tamaño o funcionalidad.

Modelo: Objeto de imitación. Objeto, construcción u otro objeto con un diseño del que se reproducen más iguales. Esquema teórico de un sistema o de una realidad compleja que se elabora para facilitar su comprensión y estudio.

Planificación: Es la actividad fundamental del gestor de proyecto que comprende la formulación de lo que hay que realizar para obtener una finalidad que será precisamente la del sistema que estamos planificando (Decidir + Hacer).Control: Es inspección, fiscalización, intervención. Esta actividad no se centra solamente en realizar planes, sino controlar su ejecución y puesta en práctica.

Preguntas abiertas: permiten emitir una opinión propia del encuestado.

Preguntas cerradas: son aquellas que se le asignan valores numéricos.

Proceso: Cualquier actividad, o conjunto de actividades, que utiliza recursos para transformar entradas en salidas.

Procesos de Gestión: Aquellos que ofrecen salidas que regulan o determinan lineamientos para otros procesos de la organización.

Productividad: Relación entre la cantidad de bienes y servicios producidos y la cantidad de recursos utilizados

Proyecto: Proceso único consistente en un conjunto de actividades coordinadas y controladas con fechas de inicio y de finalización, llevadas a cabo para lograr un objetivo conforme con requisitos específicos.

Puntos de caso de uso: Representa una suma ponderada del número de actores y el número de casos de uso de la especificación.

Puntos de función: Miden la aplicación desde una perspectiva del usuario, dejando de lado los detalles de codificación. Se define como una función comercial del usuario final.

ANEXOS

Anexo #1: Estándar de Tipos de Defecto.

Tipos de defectos:

Número Tipo	Nombre del tipo	Descripción
10	Documentación	comentarios, mensajes
20	Sintaxis	ortografía, puntuación, tipos, formatos de instrucción
30	Construcción, paquete	Gestión de cambios, librerías, control de versiones
40	Asignación	declaración, nombres duplicados, ámbito, limites
50	Interfaz	Llamadas de rutinas y referencias, I/O, formatos
60	Comprobación	mensajes error, comprobaciones no adecuadas
70	Datos	estructura, contenido
80	Funciones	lógica, punteros, bucles, recursión, cálculos, defectos de la función
90	Sistema	configuración, tiempos, memoria
100	Entorno	diseño, compilación, probar, y otros problemas del sistema de soporte

Anexo #2: Cuaderno Registro Defectos.

Ingeniero _____ Fecha _____

Líder de Equipo _____ #Programa _____

Fecha	Número	Tipo	Inyectado	Eliminado	Tiempo corrección	Defecto corregido

Descripción: _____

Fecha	Número	Tipo	Inyectado	Eliminado	Tiempo corrección	Defecto corregido

Descripción: ____________________

Fecha	Número	Tipo	Inyectado	Eliminado	Tiempo corrección	Defecto corregido

Descripción: ____________________

Fecha	Número	Tipo	Inyectado	Eliminado	Tiempo corrección	Defecto corregido

Descripción: ____________________

Fecha	Número	Tipo	Inyectado	Eliminado	Tiempo corrección	Defecto corregido

Descripción: ____________________

Fecha	Número	Tipo	Inyectado	Eliminado	Tiempo corrección	Defecto corregido

Descripción: ____________________

Fecha	Número	Tipo	Inyectado	Eliminado	Tiempo corrección	Defecto corregido

Descripción: ____________________

Instrucciones para el Cuaderno Registro Defectos

Propósito	Esta tabla mantendrá los datos de cada defecto encontrado y corregido. Utiliza estos datos para completar el Resumen del Plan de Proyecto.
General	Anota todos los defectos de revisiones, compilaciones, y pruebas en este cuaderno. Anota cada defecto de forma separada y completa. Si necesitas mas espacio, usa otra hoja.
Cabecera	Indica los siguientes datos: - nombre - la fecha de hoy - el número del programa
Fecha	Anota el día que se encontró el defecto.
Número	Anota el número de defecto. En cada programa, suele ser un número en secuencia comenzando por el 1 (ó 001, etc.).
Tipo	Anota el tipo de defecto, según la lista de defecto del anexo 1 (también resumida en la esquina superior izquierda del cuaderno). Utiliza el criterio del ingeniero para seleccionar que tipo aplicar.
Inyectado	Anota la fase donde se introdujo el defecto. Usa el criterio del ingeniero.
Eliminado	Anota la fase donde se eliminó el defecto. Normalmente, seria la fase donde se encontró y corrijo el defecto.
Tiempo corrección	Estima el tiempo que llevó corregir el defecto. Se puede usar un cronómetro o estimarlo.
Defecto corregido	Si se ha introducido este defecto mientras se corregía otro defecto, se debe anotar el número del defecto incorrectamente corregido. Si no se puede identificar el número de defecto, se debe anotar una X en la casilla.
Descripción	Se debe hacer una descripción breve y clara del defecto para recordar después, el error que causó el defecto y por qué se hizo.

Anexo #3: Resumen del Plan de Proyecto.

Ingeniero		Date	
Programa		Programa #	
Líder del Equipo		Lenguaje	

Tiempo en Fase (min.)	Plan	**Actual**	**Hasta Hoy**	**Hasta hoy %**
Planificación				
Diseño				
Código				
Compilación				
Pruebas				
Post-mortem				
Total				

Defectos Inyectados	**Actual**	**Hasta Hoy**	**Hasta hoy %**
Planificación			
Diseño			
Código			
Compilación			
Pruebas			
Total Desarrollo			

Defectos Eliminados	**Actual**	**Hasta Hoy**	**Hasta Hoy %**
Planificación			
Diseño			
Código			
Compilación			
Pruebas			
Total Desarrollo			
Después Desarrollo			

Instrucciones para el Resumen del Plan de Proyecto PSP

Propósito	Este formulario trata de los datos estimados y reales de los proyectos de una forma cómoda y fácilmente recuperable.
Cabecera	Introduce los siguientes datos: - nombre y la fecha de hoy. - el nombre y el número de programa. - el nombre del líder del equipo - el lenguaje que se utilizará para escribir el programa
Tiempo en Fase	- Debajo de Plan, se introduce la estimación inicial del tiempo de desarrollo total. - Debajo de Actual, se introduce el tiempo actual en minutos gastados en cada fase de desarrollo. - Debajo de Hasta Hoy, se introduce la suma del tiempo actual y del tiempo 'Hasta Hoy' de los programas desarrollados últimamente. - Debajo Hasta Hoy %, se introduce el % de tiempo Hasta Hoy de cada fase.
Defectos Inyectados	- Debajo Actual, se indica el número de defectos inyectados en cada fase. - Debajo Hasta Hoy, se indica la suma de la cantidad actual de defectos inyectados en cada fase y los valores Hasta Hoy de los últimos programas. - Debajo Hasta Hoy %, se indica el % de los defectos inyectados Hasta Hoy por fase.
Defectos Eliminados	- Debajo Actual, se introduce el número de defectos eliminados en cada fase. - Debajo Hasta Hoy, se introduce la suma del número de defectos actuales eliminados en cada fase y el valor de Hasta Hoy de los últimos programas. - Debajo Hasta Hoy %, se indica el % de los defectos Hasta Hoy eliminados por fase. - Después del desarrollo, se anotan los defectos encontrados durante el uso del programa, reutilización o modificación.

Anexo #4: Cuaderno Registro Tiempos.

Ingeniero _____ Fecha _____

Líder de Equipo _____ Programa # _____

Fecha	Inicio	Fin	Tiempo Interrupción	Delta	Fase	Comentarios

Instrucciones para Cuaderno de Registro Tiempos.

Propósito	Esta tabla se usa para anotar el tiempo gastado en cada fase del proyecto. Estos datos se usan para completar el Resumen Plan de Proyecto.
General	- Se anota todo el tiempo que gastes en el proyecto. - Se anota el tiempo en minutos. - Se debe ser lo mas preciso posible. Si se necesita más espacio, usa otra hoja.
Cabecera	Indica los siguientes datos: - nombre - la fecha de hoy - el nombre del líder de proyecto - el número del programa - si se esta trabajando en una tarea que no es de programación, se anota la descripción del trabajo en el campo # Programa.
Fecha	Se anota la fecha de inicio.
Ejemplo	10/18/93
Inicio	Se anota la hora de comienzo a trabajar en la tarea.
Ejemplo	8:20
Stop	Se anota la hora que se para el trabajo.
Ejemplo	10:56
Interrupción Time	Se anota los tiempos de las interrupciones y los que no se dedican a la tarea y la razón de la interrupción. Si se tienen varias, se anota el tiempo total.
Ejemplo	37 - descanso
Delta Time	Se anota el tiempo de reloj que se ha dedicado a la tarea, quitando el tiempo de las interrupciones.
Ejemplo	Desde 8:20 a 10:56, menos 37 minutos son 119 minutos.
Fase	Se anota el nombre o la etiqueta de la fase o paso en el que se trabaja.
Ejemplo	planificación, código, pruebas, etc.
Comentarios	Se anota cualquier comentario que se considere importante que pueda ayudar a recordar cualquier circunstancia inusual de la actividad.
Ejemplo	Se tenía un problema con el compilador y se necesitó ayuda.
Importante	Es importante anotar todo el tiempo de trabajo. Si se olvida anotar el comienzo, el fin, o el tiempo de interrupción de la tarea, rápidamente se anota una estimación del tiempo.

Anexo #5: Encuesta Realizada a los proyectos.

Este cuestionario tributará a la tesis "Propuesta de medición a los proyectos de la facultad 4 basados en PSP y TSP". Para realizar el cuestionario no necesita dar constancia de su nombre, los autores les brindan confidencialidad.

Datos de la Persona Entrevistada:

1. ¿Qué rol desempeña?

   _____

Datos del Proyecto:

1. Proyecto al que pertenece:

   ____ Banco:
   ____ Aduana: Gestión Integral de Recursos Humanos.
   ____ Aduana: Información Adelantada.
   ____ Aduana: Gestión de los Procesos Aduanales.
   ____ RESISOFT
   ____ SIGEP: Automatización de las Instalaciones Penitenciarias.
   ____ MINFAR.

2. ¿Cuántos años lleva usted desarrollando software?

   _____ Menos de 3 años.

   _____ De 3 a 5 años.

   _____ Más de 5 años.

Datos sobre medición en los proyectos:

1. ¿En su proyecto se realizan mediciones?

____ Si.

____ No.

____ No se.

2 ¿En su proyecto se registra el tiempo?

____ Si.

____ No.

____ A veces.

____ No se.

En caso que su respuesta sea SI. ¿Qué registra?

____ Tiempo real dedicado a cada fase.

____ Tiempo Real a la Fecha por fase.

____ % del Tiempo Real a la Fecha por fase.

____ Tiempo Total Planificado.

____ Otras.

3 ¿En su proyecto se realiza el registro de defecto?

____ Si.

____ No.

____ A veces.

____ No se.

En caso que su respuesta sea SI. ¿Qué registra?

____ Cantidad Total de Defectos inyectados por fases o tareas.

____ Cantidad Total de Defectos eliminados por fases o tareas.

____ Otras.

4 En su proyecto se aplican estándares en las etapas de:

____ Análisis.

____ Diseño.

____ Codificación o Implementación.

5 ¿En su proyecto se realiza la medición del tamaño?

____ Si.

____ No.

____ A veces.

____ No se.

6 ¿En su proyecto se realiza la estimación del tamaño?

____ Si.

____ No.

____ No se.

En caso que su respuesta sea SI. ¿Qué Métodos utiliza?

____ Empíricos.

____ Heurísticos.

____ Juicio del experto.

____ Puntos de Casos de Uso.

____ Puntos de Función.

____ Otros. ¿Cuáles?

7 ¿En su proyecto se realizan pruebas?

____ Si.

____ No.

____ A veces.

____ No se.

En caso que su respuesta sea SI. ¿Realiza un reporte de los resultados de las pruebas?

____ Si.

____ No.

____ No se.

8 ¿En su proyecto se planifican las actividades?

____ Si.

____ No.

____ A veces.

____ No se.

9 ¿Planifica el tiempo que le dedica a cada actividad?

____ Si.

____ No.

____ A veces.

____ No se.

10 ¿En su proyecto se gestiona la planificación del tiempo?

____ Si.

____ No.

____ A veces.

____ No se.

11 ¿Realiza la revisión del código?

____ Si.

____ No.

____ A veces.

____ No se.

12 ¿Realiza la revisión del diseño?

____ Si.

____ No.

____ A veces.

____ No se.

13 De los siguientes roles. ¿Cuales están presentes en su proyecto?

____ Líder de Equipo.

____ Jefe de Desarrollo.

____ Jefe de Planificación.

____ Jefe del Proceso y la Calidad.

____ Jefe de Soporte.

14 De las siguientes actividades. ¿Cuáles se realizan en su proyecto?

____El Lanzamiento del Proyecto.

____La Estrategia de Desarrollo.

____ El Plan del Proyecto.

____ Definición de los Requisitos.

____ El Diseño con Equipos.

____Implementación del Producto.

____Pruebas.

____El Postmortem.

15 En su proyecto:

____ Se forma y mantiene un equipo efectivo.

____Se motiva a todos los miembros para trabajar agresivamente en el proyecto.

____Se resuelven todos los problemas que los miembros del equipo te plantean.

____Se mantiene al Líder de Equipo completamente informado del progreso del equipo.

____Se usa completamente los conocimientos y habilidades de los miembros del equipo.

____Todos los miembros del equipo reportan exacta y adecuadamente el uso de los datos del proceso.

____Todas las reuniones son reportadas con exactitud y los reportes se colocan en el libro del proyecto.

____El equipo está dotado con las herramientas y métodos para soportar su trabajo. 94.7%

____Todos los problemas y riesgos del equipo son registrados en el sistema de control de problemas y reportados cada semana.

Anexo #6: Datos de los miembros del Panel de Expertos.

Expertos	Año vinculados a proyectos	Graduado de:	Rol que ocupa en el proyecto.
Experto 1	3	Ingeniero Informático	Líder de Proyecto
Experto 2	3	Ingeniero Informático	Líder de Proyecto
Experto 3	2	Ingeniero Informática	Jefe de Modulo
Experto 4	2	Ingeniera Informática	Líder de Proyecto.
Experto 5	4	Ingeniera Informática	Líder de Proyecto.

Anexo #7: Encuesta realizada al Panel de Expertos.

Cuestionario

1. ¿Cree usted que la propuesta está a la altura de las posibilidades y necesidades de los Proyectos Productivos de Software?

__Si __No ¿Por qué?

2. En la escala del 1 al 5, otorgue una evaluación al la propuesta según los siguientes criterios:

a) __Satisfacción de las necesidades en los Proyectos Productivos.

b) __Adaptabilidad a los Proyectos Productivos.

c) __ Repercusión en los Proyectos Productivos.

d) __ Posibilidad de aplicación.

e) __ Posibilidad obtener datos reales para hacer una mejor planificación.

2.1 En la escala del 1 al 5, otorgue una evaluación a las actividades propuestas según su importancia para la planificación de proyectos

f) __Registro de Tiempo.

g) __Registro de Defectos.

h) __ Resumen del Plan de Proyecto.

i) __Estimación del tamaño.

j) __ Lanzamiento del Proyecto

k) __ Estrategia de Desarrollo.

l) __ Plan de Proyecto.

m) __ Postmortem

n)__ Utilización de métricas.

3. ¿Qué argumentos usted expondría a favor de la aplicación de la propuesta y cuáles estarían en contra?

4. Realice alguna observación sobre la propuesta que es por su parte objeto de su evaluación.

1