**Tesis para optar el título profesional de Ingeniero de Software**

**Autor**

CÁCERES NEYRA, Peter Diego Arturo

**Asesor**

LA SERNA PALOMINO, Nora Bertha

**Lima – Perú**

**2014**

**FICHA CATALOGRÁFICA**

CÁCERES NEYRA, Peter Diego Arturo

IMPLEMENTACION DE UNA HERRAMIENTA INFORMATICA PARA LA ESTIMACION DE PROYECTOS DE DESARROLLO DE SOFTWARE EN EL AMBITO ACADEMICO UTILIZANDO EL CONJUNTO DE TENICAS TUPUY

Ingeniería de Software

(Lima, Perú 2013)

Tesis, Facultad de Ingeniería de Sistemas e Informática, Pregrado, Universidad Nacional Mayor de San Marcos

Formato 21 x 29.7 cm Páginas

DEDICATORIA

El presente trabajo está dedicado a mis padres….

AGRADECIMIENTOS

Agradezco a mis padres por los principios y valores inculcados en mi vida, a todos mis maestros que compartieron conmigo sus conocimientos y experiencias, a mi Universidad San Marcos por acogerme en sus aulas y por permitirme crecer como persona y profesional y a mi asesora Dra. Nora La Serna por ayudarme a materializar el presente proyecto.

**UNIVERSIDAD NACIONAL MAYOR DE SAN MARCOS**

**FACULTAD DE INGENIERIA DE SISTEMAS E INFORMÁTICA**

**ESCUELA ACADEMICO PROFESIONAL DE INGENIERÍA DE SOFTWARE**

**IMPLEMENTACION DE UNA HERRAMIENTA INFORMATIVA PARA LA ESTIMACION DE PROYECTOS DE DESARROLLO DE SOFTWARE EN EL AMBITO ACADEMICO UTILIZANDO EL CONJUNTO DE TECNICAS TUPUY**

Autor: CÁCERES NEYRA, Peter Diego Arturo

Asesor: LA SERNA PALOMINO, Nora

Título: Tesis, para optar el Título Profesional de Ingeniero de Software

Fecha: Abril del 2014

RESUMEN

….

**PALABRAS CLAVES**: Tecnicas de Estimacion, Estimacion de Software, Tupuy, Puntos de Función.

**MAJOR NATIONAL UNIVERSITY OF SAN MARCOS**

**FACULTY OF SYSTEMS ENGINEERING AND INFORMATIC**

**ACADEMIC PROFESSIONAL SCHOOL SYSTEMS ENGINEERING**

**INFORMATIVE IMPLEMENTATION OF A TOOL FOR ESTIMATING SOFTWARE DEVELOPMENT PROJECTS IN THE FIELD OF ACADEMIC SET OF USING TECHNICAL TUPUY**

Author: CÁCERES NEYRA, Peter Diego Arturo

Adviser: LA SERNA PALOMINO, Nora

Title: Thesis, to choose the Professional Title Software Engineer

Date: February 2013

ABSTRACT

….

**KEY WORDS**: Estimation Techniques, Estimating Software, Tupuy, Function Points

ÍNDICE DE CONTENIDO

[DEDICATORIA i](#_Toc387484420)

[AGRADECIMIENTOS ii](#_Toc387484421)

[RESUMEN iii](#_Toc387484422)

[ABSTRACT iv](#_Toc387484423)

[ÍNDICE DE CONTENIDO v](#_Toc387484424)

[CAPÍTULO 1: INTRODUCCIÓN 1](#_Toc387484425)

[1.1 Antecedentes 1](#_Toc387484426)

[1.2 Definición del problema 2](#_Toc387484427)

[1.3 Objetivos 2](#_Toc387484428)

[1.3.1 Objetivo principal 2](#_Toc387484429)

[1.3.2 Objetivo secundarios 2](#_Toc387484430)

[1.4 Justificación 3](#_Toc387484431)

[1.5 Alcance 3](#_Toc387484432)

[CAPÍTULO 2: MARCO TEORICO REFERENCIAL 4](#_Toc387484433)

[2.1 Estimación para Proyectos de Software 4](#_Toc387484434)

[2.2 Técnicas de Estimación 4](#_Toc387484435)

[2.2.1 Puntos de Función 4](#_Toc387484436)

[2.2.2 COCOMO II 6](#_Toc387484437)

[2.2.3 Metodología Tupuy 8](#_Toc387484438)

[REFERENCIAS BIBLIOGRÁFICAS 9](#_Toc387484439)

# CAPÍTULO 1: INTRODUCCIÓN

* 1. **Antecedentes**

Al inicio, cuando surgen los primeros programas de computadora, la actividad de la programación se orientaba a entregar una “funcionalidad” y era lo único que importaba, hacer el programa y que funcionase.

Poco a poco mientras el software evoluciona y era cada vez más complejo, se requirió no solo de la programación sino también de un proceso con actividades definidas y claras para guiar a la construcción de software.

Como todo proceso bien estructurado se requiere de una etapa de planeación donde surgen dudas ¿Cómo sabemos cuánto tiempo necesitaremos para terminar el software? O ¿Cuántas personas necesitaremos estén involucradas en el desarrollo del software?

Así es como respuesta a estas interrogantes surgen los primeros modelos y técnicas para la estimación de proyectos de software tales como el modelo COCOMO que fue publicado en 1981 por Barry W. Boehm basándose en el estudio del proceso de desarrollo de ese entonces y posteriormente se publicó una revisión de este modelo llamada COCOMO II [1].

Otra técnica también es la de Puntos de función, definida por Allan Albrecht en 1979 [2], es una técnica utilizada para medir el tamaño del software basándose en las funcionalidades ofrecidas, esta técnica sigue vigente en la actualidad debido actualizaciones que recibe de parte del IFPUG (International Function Point Users Group) [3].

* 1. **Definición del problema**

A menudo en nuestra facultad se suelen realizar proyectos de desarrollo de software en cursos en donde los alumnos deben aplicar sus conocimientos sobre el proceso de desarrollo según una metodología que se hallan fijado. Es un problema que los alumnos a lo largo del desarrollo del proyecto tengan que restringir el alcance a uno que es cada vez menor al que definieron al inicio, esto hace que su meta de presentar un software según lo planeado no se cumpla y reduzcan sus expectativas sobre un proceso de desarrollo de software confiable.

* 1. **Objetivos**
     1. **Objetivo principal**

Desarrollar una herramienta software que proporcione la estimación del tiempo y personal necesitado aplicando el conjunto de técnicas Tupuy ayudando así a los alumnos a definir el tiempo y el número de los integrantes de los equipos para sus proyectos.

* + 1. **Objetivo secundarios**
* Revisar trabajos relacionados que se orienten a la construcción de herramientas para la estimación de software.
* Definir el marco conceptual donde se utilizara la herramienta de estimación a desarrollar.
* Desarrollar los componentes del software que implementen el conjunto de técnicas Tupuy.
  1. **Justificación**

Debido a problemas en la estimación inicial para los proyectos académicos se ha decidido la implementación de una herramienta para ayudar y orientar en la estimación del tiempo y costo de los proyectos de software realizados por los alumnos, de esta manera los alumnos podrán estimar tempranamente el tiempo que le costara llevar a cabo su proyecto de software y definir cuantas personas tendrán que estar involucradas para terminar.

* 1. **Alcance**

En el presente trabajo se implementara una herramienta de software basándose en las técnicas que comprende Tupuy. Tupuy comprende 3 técnicas que son UML2FP, Use Case Precedence Diagram (UCPD) e Incremental-FP, las tres serán implementadas en la herramienta.

# CAPÍTULO 2: MARCO TEORICO REFERENCIAL

* 1. **Estimación para Proyectos de Software**

La gestión del proyecto de software comienza con un conjunto de actividades que en grupo se denominan planificación del proyecto. Antes de que el proyecto comience el gestor del proyecto y el equipo de software deben estimar el trabajo que habrá de realizarse, los recursos que se requerirán y el tiempo que transcurrirá desde el principio hasta el final. Una vez que se completen estas actividades, el equipo de software debe establecer un plan del proyecto que defina las tareas y fechas clave de la ingeniería del software, que identifique quien es responsable de dirigir cada tarea y especifique las dependencias entre tareas que pueden ser determinantes en el proceso. [4]

Aunque la estimación es tanto un arte como una ciencia, esta importante actividad no necesita realizarse en una forma improvisada. Existen técnicas útiles para la estimación de tiempo y esfuerzo. Las métricas del proceso y del proyecto ofrecen la perspectiva histórica y la energía para la generación de estimaciones cuantitativas. La experiencia puede auxiliar enormemente conforme se desarrollan y revisan las estimaciones. Puesto que la estimación coloca los cimientos para las demás actividades de planificación del proyecto, y esta proporciona al ruta para la ingeniería del software exitosa, se estaría mal aconsejando si se embarca sin ella. [4]

* 1. **Técnicas de Estimación**
     1. **Puntos de Función**

Los Puntos de Función permiten estimar el tamaño del software a partir de sus requerimientos, mientras que los Casos de Uso permiten documentar los requerimientos del software. Ambos tratan de ser independientes de las tecnologías utilizadas para la implementación. En etapas tempranas del ciclo de vida, se identifican los Actores y los Casos de Uso del sistema, y se documenta cada uno de ellos mediante una breve descripción. Aplicando el Análisis de Puntos de Función a estos Casos de Uso, se podrá obtener una estimación bastante imprecisa debida principalmente a la escasa información que se tiene sobre el software al principio de un proyecto, pero permitirá obtener una idea del esfuerzo necesario para llevar adelante el mismo, y podrá ser refinada a medida que se obtenga más información. Si se aplica nuevamente el Análisis de Puntos de Función sobre una descripción más detallada de los Casos de Uso, la estimación del tamaño y esfuerzo será más precisa que la anterior. Esta métrica se define como una métrica funcional, dado que se enfoca a la funcionalidad que el software proporciona al usuario.

Es una métrica para establecer el tamaño y complejidad de los sistemas informáticos basada en la cantidad de funcionalidad requerida y entregada a los usuarios. Los Puntos de Función miden el tamaño lógico o funcional de los proyectos o aplicaciones de software basado en los requerimientos funcionales del usuario.

Partamos de la primera definición para entender las características de la métrica:

* Tamaño: es una métrica de tamaño, no de la calidad con la que se hizo ese software, o del valor de ese producto, o del esfuerzo requerido para desarrollarlo, etc.
* Aplicaciones: mide las aplicaciones de software, no considera el hardware que utilizará, ni la administración del proyecto, ni la documentación, etc.
* Funcionalidad: se refiere a la capacidad del software para que un usuario pueda realizar transacciones (lectura, escritura etc.) y el guardar datos. Si analizamos a detalle, con estos elementos podemos describir cualquier sistema.
* Usuario: quien lo va a usar y no quien lo desarrolló o quien lo diseñó.

Así como existe el metro lineal para medir longitudes, Puntos Función es "el metro" para medir tamaño de una aplicación de software. El método de Puntos de Función fue publicado por primera vez en el año 1979 por Allan J. Albrecht [2] y se obtienen utilizando una relación empírica basada en medidas cuantitativas del dominio de información del software y valoraciones subjetivas de su complejidad. Es un método para medir el tamaño del software. Pretende medir la funcionalidad entregada al usuario independientemente de la tecnología utilizada para la construcción y explotación del software, y también ser útil en cualquiera de las fases de vida del software, desde el diseño inicial hasta la explotación y mantenimiento.

* + 1. **COCOMO II**

Los objetivos principales que se tuvieron en cuenta para construir el modelo COCOMO II fueron:

* Desarrollar un modelo de estimación de costo y cronograma de proyectos de software que se adaptara tanto a las prácticas de desarrollo de la década del 90 como a las futuras.
* Construir una base de datos de proyectos de software que permitiera la calibración continua del modelo, y así incrementar la precisión en la estimación.
* Implementar una herramienta de software que soportara el modelo.
* Proveer un marco analítico cuantitativo y un conjunto de herramientas y técnicas que evaluaran el impacto de las mejoras tecnológicas de software sobre los costos y tiempos en las diferentes etapas del ciclo de vida de desarrollo.

COCOMO II está compuesto por tres modelos denominados: Composición de Aplicación, Diseño Temprano y Post-Arquitectura. [1]

El modelo **Composición de Aplicación** se emplea en desarrollos de software durante la etapa de prototipado.

El modelo **Diseño Temprano** se utiliza en las primeras etapas del desarrollo en las cuales se evalúan las alternativas de hardware y software de un proyecto. En estas etapas se tiene poca información, lo que concuerda con el uso de Puntos Función, para estimar tamaño y el uso de un número reducido de factores de coste.

El modelo **Post-Arquitectura** se aplica en la etapa de desarrollo, después de definir la arquitectura del sistema, y en la etapa de mantenimiento

Cada uno de estos modelos está orientado a sectores específicos del mercado de desarrollo de software y a las distintas etapas del desarrollo de software.

**La sectorización de aplicaciones en COCOMO II:**

**Aplicaciones desarrolladas por Usuarios Finales:** En este sector se encuentran las aplicaciones generadas directamente por usuarios finales, mediante la utilización de generadores de aplicaciones tales como hojas de cálculo, sistemas de consultas, etc. Estas aplicaciones surgen debido al uso masivo de estas herramientas, conjuntamente con la presión actual para obtener soluciones rápidas y flexibles.

**Generadores de Aplicaciones:** En este sector están los módulos pre-empaquetados que serán usados por usuarios finales y programadores.

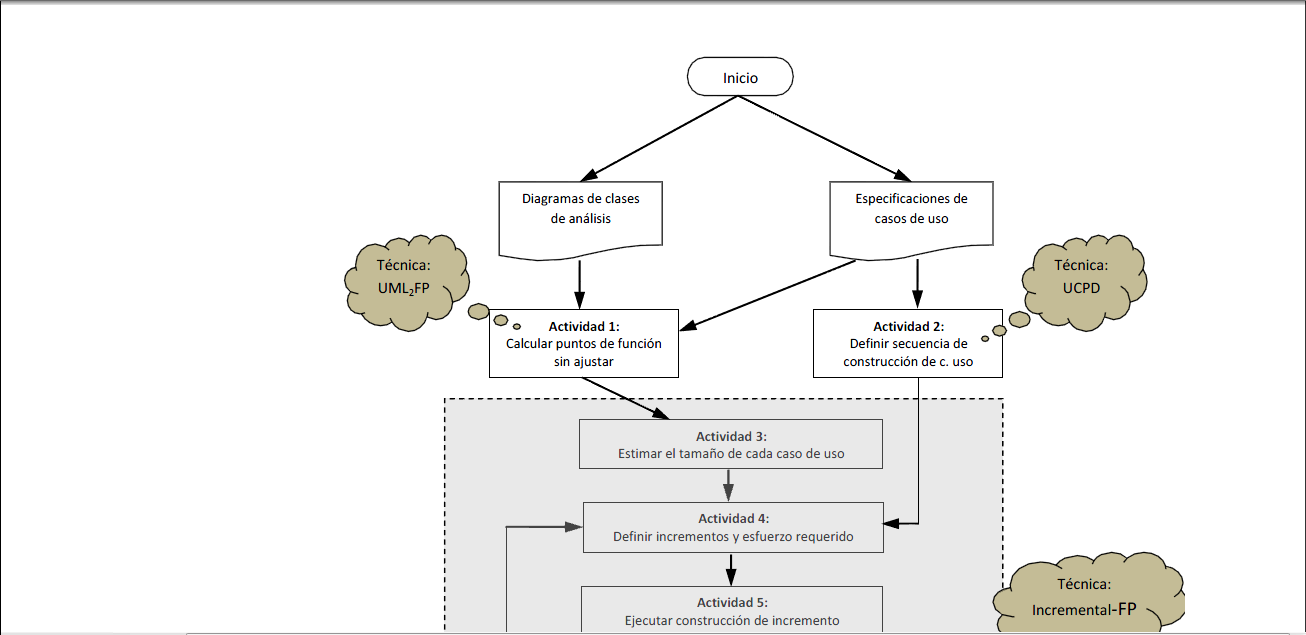
**Aplicaciones con Componentes:** Sector en el que se están aquellas aplicaciones que se resuelven por soluciones pre empaquetadas, pero son lo suficientemente simples para ser construidas a partir de componentes interoperables. Por ejemplo: interfaces gráficos, administradores de bases de datos, buscadores inteligentes de datos, … Estas aplicaciones son generadas por un equipo reducido de personas, en pocas semanas o meses.

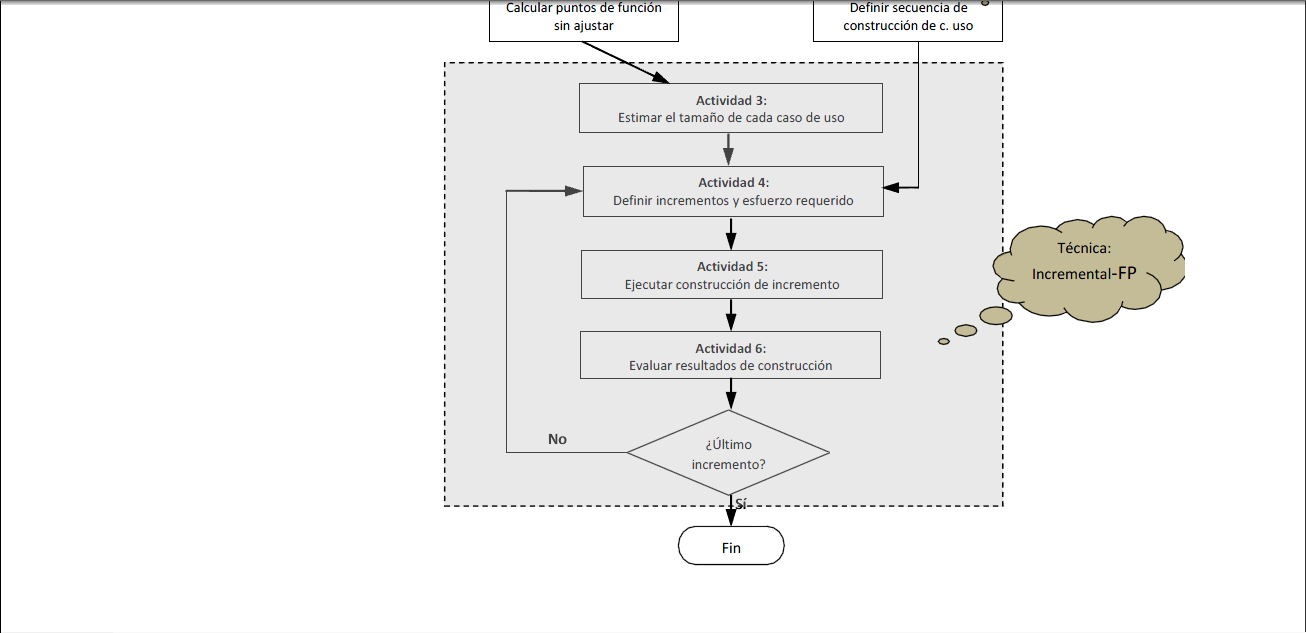
**Sistemas Integrados**: Sistemas de gran escala, con un alto grado de integración entre sus componentes, sin antecedentes en el mercado que se puedan tomar como base. Partes de estos sistemas pueden ser desarrolladas a través de la composición de aplicaciones.

**Infraestructura**: Área que comprende el desarrollo de sistemas operativos, protocolos de redes, sistemas administradores de bases de datos.

* + 1. **Metodología Tupuy**

Tupuy está conformada por tres técnicas: UML2FP, UCPD e Incremental-FP. [5] La figura 2-1 Muestra las actividades que incluye Tupuy. Además, se indica con una nube, las actividades que se hacen con cada técnica.





**Figura 2-1. Actividades de Tupuy y técnicas que incluye.**

# REFERENCIAS BIBLIOGRÁFICAS

|  |  |
| --- | --- |
| [1] | B. Boehm, Software Cost Estimation with Cocomo II, Prentice Hall PTR, 2009. |
| [2] | A. J. Albrecht, Measuring Aplication Development Productivity, 1979. |
| [3] | «IFPUG International Function Point Users Group,» 2014. [En línea]. Available: http://www.ifpug.org. [Último acceso: 10 Mayo 2014]. |
| [4] | R. S. Pressman, Ingeniería del Software: Un Enfoque Práctico, 2009. |
| [5] | J. A. Pow-Sang Portillo, «Tecnicas para la Estimación y Planificación de Proyectos de software con Ciclos de Vida Incremental y Paradigmas Orientado a Objetos,» 2012. |