**Trabajo Práctico Integrador 2**

**Sistema Maxi Kiosco Ilolay**

Universidad: Universidad Católica de Santiago del Estero – DAR

Carrera: Ingeniería en Informática

Materia: Ingeniería de Software

Profesores: Marcela Vera, Juan Carlos Ramos

Fecha de entrega: 10/11/2016

Alumnos: Giorgina Castagno

Miguel Delpuppo

Camila Kopech

Wendy Sclerandi

Índice

[1. Estimación COCOMO II 3](#_Toc466510472)

[2. Métricas de Calidad 5](#_Toc466510473)

[3. Gestión de la Configuración 7](#_Toc466510474)

[3.1. Elementos que se incluirán en la gestión de la configuración 7](#_Toc466510475)

[3.2. Herramientas automatizadas 7](#_Toc466510476)

[3.3. Plan de gestión de cambios 7](#_Toc466510477)

[3.4. Estrategia de Gestión de Versiones 8](#_Toc466510478)

[Bibliografía 9](#_Toc466510479)

# 1. Estimación COCOMO II

Teniendo en cuenta los requisitos funcionales documentados en la Especificación de Requerimientos, podemos clasificar al proyecto como orgánico. Esto se debe a que consideramos que el proyecto es relativamente sencillo de realizar y sus requerimientos no son muy cambiantes. Además, se cuenta con profesionales que tiene experiencia en entornos similares.

Punto de objeto

El punto de objeto es una medida de software indirecta que se calcula utilizando cantidades del número de pantallas, reportes y componentes. Cada instancia de objeto se clasifica en uno de tres niveles de complejidad, como se muestra en la siguiente tabla:

|  |  |  |  |
| --- | --- | --- | --- |
| Tipo de objeto | Complejidad simple | Complejidad media | Complejidad difícil |
| Pantalla | 1 | 2 | 3 |
| Reporte | 2 | 5 | 8 |
| Componentes requeridos |  |  | 10 |

Las cantidades de cada objeto que se utilizarán para realizar las estimaciones serán las siguientes:

|  |  |
| --- | --- |
| Pantallas | 11 |
| Reportes | 3 |
| Componentes Requeridos | 1 |

A cada objeto se le asigna un nivel de complejidad, según sus funcionalidades correspondientes:

Pantalla:

* Complejidad Simple:
  + Login
  + Alta de productos
  + Baja de productos
  + Modificaciones de productos
  + Alta de proveedores
  + Baja de proveedores
  + Modificaciones de proveedores
* Complejidad Media:
  + Alta de pedidos
  + Modificaciones de pedidos
  + Consulta de reportes
  + Formulario de ventas

Reporte:

* Complejidad Simple:
  + Impresión de facturas
  + Impresión de tickets
* Complejidad Media:
  + Impresión de reportes

Componentes requeridos:

* Complejidad Difícil:
  + Bases de datos para la persistencia de los datos

A continuación, teniendo la complejidad de los objetos y la cantidad de cada uno, se pueden listar en la siguiente tabla de ponderación de la complejidad:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Tipo de objeto | Simple | Cantidad | Media | Cantidad | Difícil | Cantidad | Total |
| Pantalla | 1 | 7 | 2 | 4 | 3 | 0 | 15 |
| Reporte | 2 | 2 | 5 | 1 | 8 | 0 | 9 |
| Componentes Requeridos |  |  |  |  | 10 | 1 | 10 |
| Total de números de Puntos de Objeto | | | | | | | 34 |

La siguiente tabla muestra la tasa de productividad para puntos de objetos, necesaria para realizar los cálculos de estimación.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Experiencia/Capacidad del desarrollador | Muy Baja | Baja | Nominal | Alta | Muy Alta |
| Productividad (NOP/mes) | 4 | 7 | 13 | 25 | 50 |

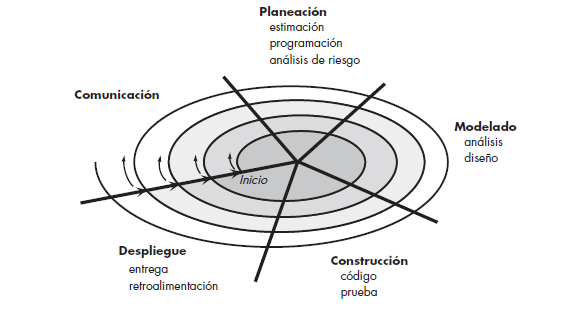
Por último, se detallan los cálculos realizados para la estimación de esfuerzo, tiempo, personal y costo:

Se consideran a 2 programadores de experiencia media

El sueldo de los desarrolladores es de $10000 por mes.

# 2. Métricas de Calidad

Las etapas correspondientes a cada iteración del ciclo de vida en espiral son las siguientes:



Comunicación

En la primera iteración de esta etapa no se pueden aplicar métricas ya que sólo se conversa con el cliente sobre los objetivos del software, sus restricciones y riesgos. Sin embargo, en las demás iteraciones se pueden aplicar métricas para evaluar el nivel de aceptación del usuario sobre el incremento entregado en cada iteración.

Planeación: En esta fase se crea el plan para el desarrollo incremental y se define el equipo de trabajo. El plan de desarrollo consiste en la definición de tareas, actividades y resultados durante el desarrollo.

La métrica que se utilizará será Puntos de Función ya que permite evaluar las funcionalidades externas que proporciona el sistema. Se tiene en cuenta el número de entradas de usuario, número de salidas de usuario, número de peticiones de usuario, número de archivos y número de interfaces externas.

Modelado: Se crean todos los modelos de análisis y diseño.

* Métricas para los modelos de análisis:
  + Especificidad (ausencia de ambigüedad)
  + Grado de Validación de Requerimientos:
    - Requerimientos validados sobre no validados
    - Requerimientos validados sobre el total de requerimientos.
* Métricas para los modelos de diseño:
  + Nivel de acoplamiento entre módulos.

Construcción: En esta fase se llevan a cabo la codificación y las pruebas.

Las métricas adecuadas para utilizar en esta etapa son:

* Líneas de código por hora de programador.
* Productividad diaria de los programadores.
* Cantidad de errores por KLDC.

Despliegue: Esta fase implica las actividades de entrega, pruebas de usabilidad y retroalimentación.

Las métricas a aplicar en esta fase son:

* Tiempo medio de respuesta
* Número de medio de transacciones por hora.
* Tiempo medio entre fallas/caídas.
* Cantidad de fallas por día.

A continuación, se especifican cuatro métricas de las mencionadas anteriormente:

Métrica: Cantidad de requerimientos válidos por el usuario sobre requerimientos totales.

* Objetos medidos: Requerimientos funcionales y no funcionales.
* Atributos medidos: Cantidades.
* Cómo medir:
* Valor de medida: Porcentaje de requerimientos validados.
* Indicador: 🡪 Los riesgos están altamente reducidos.

Métrica: Cantidad de fallas por día.

* Objeto medido: Fallas
* Atributos medidos: Cantidades.
* Como medir:
* Valor de medida: Cantidad
* Indicador: 🡪 el software tiene es confiable.

Métrica: Cantidad de líneas de código por hora de programador.

* Objeto medido: Líneas de código.
* Atributos medidos: Tiempo.
* Como medir:
* Valor de medida:
* Indicador: 🡪 hay buena productividad por parte de los programadores y se avanza correctamente con el proyecto.

Métrica: Productividad diaria de programadores.

* Significado: Cantidad de tareas realizadas por día por programador.
* Objeto medido: Tareas.
* Atributos medidos: Tiempo.
* Como medir:

* Valor de medida: Tareas.
* Indicador: Se evaluará teniendo en cuenta la cantidad de tareas que es capaz de realizar un programador diariamente.

# 3. Gestión de la Configuración

## 3.1. Elementos que se incluirán en la gestión de la configuración

Los elementos que se incluirían en la gestión de la configuración son:

* Documento de especificación de requerimientos.
* Diseño de la base de datos.
* Diseño de Interfaz.
* Casos de prueba.
* Manual de usuario.
* Manual técnico incluyendo el diccionario de datos.
* Hardware requerido.
* Diagramas de clases.
* Diagramas de secuencia.
* Diagramas de despliegue.
* Código fuente.

## 3.2. Herramientas automatizadas

Las herramientas a utilizar para llevar a cabo la gestión de la configuración son:

Control de versiones:

La herramienta a utilizar será Git. La misma persigue los siguientes objetivos:

* Velocidad.
* Diseño sencillo.
* Fuerte apoyo al desarrollo no lineal (miles de ramas paralelas).
* Completamente distribuido.
* Capaz de manejar grandes proyectos de manera eficiente.

Es por esto que facilita la organización del control del versionado de fuentes y hace que la integración al proyecto principal no resulte tan engorrosa de realizar y se ejecute en un corto tiempo.

Control de cambios:

La herramienta utilizada para esta tarea será Redmine. La misma se encarga de la administración de proyectos y seguimiento de incidencias.

Con esto aseguramos que los costos y beneficios del cambio sean adecuados y que el cambio se implemente correctamente.

## 3.3. Plan de gestión de cambios

Para gestionar un cambio se comienza por la generación de una propuesta de cambio. Una vez recibida la misma, se la evalúa en cuanto a esfuerzo técnico, efectos secundarios, y otros aspectos relevantes, y se la documenta junto con los demás cambios requeridos.

A continuación, se procede a una etapa de revisión en la que la Autoridad de Control de Cambio evalúa la aceptación o rechazo de la modificación solicitada. Si la misma es rechazada, se finaliza el proceso. Por el contrario, si el cambio es aprobado, se procede a la elaboración de la Orden de Cambio de Ingeniería para cada cambio, y a su implementación. Para finalizar el proceso, es necesaria una verificación para comprobar que el cambio fue aplicado correctamente, lo cual se realiza en una auditoría.

## 3.4. Estrategia de Gestión de Versiones

La estrategia de versionado a utilizar será la convención <id><9>.<9>.<9>.<9>, siendo el id la identificación de la versión. Tomando como ejemplo Id 1.2.3.4, los números corresponden a los siguientes aspectos:

* 1: *reléase*: se incrementa cuando se produce una modificación importante del producto.
* 2: *nivel*: se incrementa cuando se produce una modificación leve en el producto.
* 3: *versión*: se incrementa cuando se corrige un fallo o problema.
* 4: *instalación*: se incrementa cuando se realiza una modificación debido al entorno donde se instalará.

La escogimos por ser considerada una nomenclatura clara y de utilización masiva. Además es completa porque permite visualizar en cada cambio su dimensión e importancia.

# Bibliografía

Pressman, R. (2010). *Ingeniería del Software: un enfoque práctico. Séptima Edición.* Mc Graw-Hill.

Sommerville, I. (2010). *Ingeniería de Software. Novena Edición.* Pearson Educación.