**Implementación del Sistema de gestión de boletos para autobuses intercantonales e interprovinciales (SGA).**

**Autores.**

Docente: Franklin Ricardo Parrales Bravo, Doctor en ingeniería Informática, Máster en Inteligencia Artificia Universidad Politécnica de Madrid, 2019, Máster en Ingeniería Informática, Universidad Complutense de Madrid, 2015. Ingeniero en Ciencias Computacionales, Escuela Superior Politécnica del Litoral, 2010.

Estudiantes:

Sergio Vicente Campoverde Leiva

Kevin Alexander Arévalo Saldaña

Ronny Elian Gómez Peñafiel

Ángel Adrián Chavarrea Jaramillo

Johann Donato Lucio Vera

Kevin Andrés Romero Mero

Kevin Bryan Sotomayor Carrera

David Fernando Zorrilla Baque

**Resumen.**

En el presente documento se realiza una breve descripción de las herramientas que se utiliza para la implementación del sistema de gestión de boletos para los autobuses intercantonales e interprovinciales (SGA).

Para mantener una organización adecuada y asignar diferentes tareas a realizar a cada integrante del equipo de desarrollo, se utilizó la metodología ágil Scrum.

El lenguaje de programación que se utilizó para la implementación del sistema es C#, con el IDE de Visual Studio 2019, para realizar las interfaces graficas de usuario se hizo uso de paquetes NuGet los cuales son: font Awesome sharp version 5.15.4 y Guna.UI2.WinForms.

Para realizar la gestión de la base de datos se utilizó MySQL.

Con la intensión por determinar cuánto dinero, esfuerzo, recursos y tiempo tomaría construir el sistema de software, se procedió a realizar los cálculos mediante el modelo COCOMO también conocido como COCOMO 81’ y COCOMO II.

**Abstract.**

This document provides a brief description of the tools used for the implementation of the ticket management system for intercantonal and interprovincial buses (SGA).

To maintain an adequate organization and assign different tasks to each member of the development team, the agile Scrum methodology was used.

The programming language that was used for the implementation of the system is C#, with the IDE of Visual Studio 2019, to make the graphical user interfaces, NuGet packages were used, which are: font Awesome sharp version 5.15.4 and Guna. UI2.WinForms.

MySQL was used to manage the database.

With the intention of determining how much money, effort, resources and time it would take to build the software system, the calculations were made using the COCOMO model, also known as COCOMO 81' and COCOMO II.

**Introducción.**

El sistema de software será realizado por los estudiantes de sexto semestre de la universidad de Guayaquil como guía principal estará a cargo el Ing. Franklin Parrales docente de la asignatura construcción de software.

En el presente artículo se hablará sobre la implementación de un sistema de software que tiene como nombre: Sistema de gestión de boletos de autobuses intercantonales e interprovinciales(SGA) este sistema será implementado para ser utilizado en el terminal de la ciudad de Guayaquil con el objetivo de optimizar el tiempo al momento de realizar las compras de boletos y agilizar el servicio a los clientes de las diferentes cooperativas de transporte, el cual será de mucha importancia tanto para la parte empresarial como para la comunidad, porque el sistema estará operativo las 24 horas del día, los 7 días a la semana por lo que los clientes se podrán acercar a realizar la compra de su boleto a cualquier hora del día.

En la sección 1 se tratará sobre las herramientas utilizadas para el desarrollo del sistema y la metodología Scrum que se utiliza para llevar una organización dentro del grupo de desarrollo, para lo cual se asignan tareas a cada integrante que serán entregadas dentro tiempo propuesto por el líder del grupo estas entregas serán conocidas como sprint.

En la sección 2 se aborda lo concerniente a la estimación aproximada que se tomará en el desarrollo del sistema de software, para lo cual se hace uso del modelo COCOMO el mismo que será evaluado en la parte de la conclusión para ver si la estimación dada tiene similitud con el tiempo real de desarrollo.

**Sección 1.**

**Herramientas para el desarrollo del sistema.**

Según para la empresa NeoAttack (2020) menciona que:

MySQL es un sistema de gestión de bases de datos que cuenta con una doble licencia. Por una parte, es de código abierto, pero por otra, cuenta con una versión comercial gestionada por la compañía Oracle. Actualmente, es la base de datos de código abierto más famosa y utilizada en el mundo entero. (pág. 1)

Para Terry Glee (2022) en su artículo manifiesta que:

Visual Studio 2019 es entorno de desarrollo que se puede usar para editar, depurar y compilar código y, después, publicar una aplicación. Aparte del editor y el depurador estándar que proporcionan la mayoría de IDE, Visual Studio incluye compiladores, herramientas de finalización de código, diseñadores gráficos y muchas más características para facilitar el proceso de desarrollo de software.

**Metodología Scrum.**

Según Manuel Trigas (2012): Scrum al ser una metodología de desarrollo ágil tiene como base la idea de creación de ciclos breves para el desarrollo, que comúnmente se llaman iteraciones y que en Scrum se llaman "Sprints". (pág. 34)

Las principales razones del uso de un ciclo de desarrollo iterativo e incremental de tipo Scrum para la ejecución de este proyecto son:

* Sistema modular. Las características del sistema del “Sistema de gestión de boletos de autobuses” permiten desarrollar una base funcional mínima y sobre ella ir incrementando las funcionalidades, modificando el comportamiento o apariencia de las ya implementadas.
* Entregas frecuentes y continuas al cliente de los módulos terminados, de forma que puede disponer de una funcionalidad básica en un tiempo mínimo y a partir de ahí un incremento y mejora continua del sistema.
* Previsible inestabilidad de requisitos. Es posible que el sistema incorpora más funcionalidades de las inicialmente identificadas.
* Es posible que durante la ejecución del proyecto se altere el orden en el que se desean recibir los módulos o historias de usuario terminadas. Esta técnica permitirá una mejor organización del equipo de trabajo a lo largo del desarrollo del sistema de software.

**Personas y roles del proyecto.**

|  |  |  |
| --- | --- | --- |
| **Nombre** | **Contacto** | **Rol** |
| Ángel Chavarrea | angel.chavarreaj@ug.edu.ec | Coordinador |
| Sergio Campoverde | sergio.compoverdel@ug.edu.ec | Gestor de producto |
| Kevin Arévalo | kevin.arevalos@ug.edu.ec | Desarrollador |
| Ronny Gómez | ronny.gomezpe@ug.edu.ec | Desarrollador |
| Kevin Sotomayor | kevin.sotomayorc@ug.edu.ec | Desarrollador |
| Kevin Romero | kevin.romerome@ug.edu.ec | Desarrollador |
| David Zorrilla | david.zorrillab@ug.edu.ec | Desarrollador |
| Johann Lucio | johann.luciov@ug.edu.ec | Desarrollador |

**Pila del sprint:** Cada sprint es una reunión en la que se realiza la entrega de las asignaciones dadas en la reunión.

**Sección 2.**

**Método COCOMO.**

El Modelo Constructivo de Costes (Constructive Cost Model) es un modelo matemático de base

empírica utilizada para la estimación de costes de software. Se publicó por primera vez en

1981 por Barry Boehm. (Luque, 2015, pág. 18)

Para hacer uso del método COCOMO primero se debe encontrar KLOC. Por lo que primero se debe encontrar los puntos de función y multiplicarlos por el factor de correlación.

A continuación, se realiza las operaciones:

**Puntos de función.**

Para Sánchez (2011)en su informe menciona que: Los Puntos Función proporcionan una medida objetiva, cuantitativa y auditable del tamaño de las aplicaciones, desde el punto de vista de los requisitos especificados por el usuario final de la aplicación. (pág. 8)

Para encontrar los puntos de función del sistema se debe tener en cuenta el número de entradas externas, salidas externas, consultas externas, archivos lógicos externos y las interfaces externas.

A cada elemento se le asigna una complejidad funcional de BAJA (B), MEDIA (M), o ALTA (A) según la cantidad asociada de Tipos de Registros (Record Element Types - RET) y Tipos de Elementos de Datos (Data Element Types - DET).

Los RET son subgrupos de archivos lógicos internos basados en la visión lógica o del usuario de los datos. Los archivos lógicos internos a los que no se les puede identificar subgrupos se considera que tienen un RET. Los DET son campos de datos reconocibles por el usuario. Cada campo en el que el archivo lógico interno es un DET, con las siguientes excepciones:

Se cuenta un FTR por cada Archivo Lógico Interno (ILF) leído, creado, o actualizado, y por cada archivo de Interfaz Externa (EIF) leído durante el procesamiento de la Entrada Externa. Se cuenta un DET por cada campo agregado, modificado, o eliminado en un Archivo Lógico Interno (ILF) debido al procesamiento de la Entrada Externa.

La tabla 1 permite obtener la cantidad de entradas externas que se tendrá en el sistema, en donde los DET es la cantidad de elementos que se identifican con la complejidad que tienen asignada en la lista de los casos de uso los cuales son tomados desde los requisitos solicitados por el cliente. Los FRT es la puntuación que se asigna dependiendo la complejidad y la cantidad de elementos que se identifica como entrada externa.

La Tabla 2 permite obtener la cantidad de salidas externas. La tabla 3 y la tabla 4 permiten obtener el número de consultas que realizara el usuario. La tabla 5 Perite obtener la cantidad de archivos lógicos internos y la tabla 6 permite obtener la cantidad de interfaces de interfaces externas.

|  |  |  |  |
| --- | --- | --- | --- |
| DET | 1 a 4 | 5 a 15 | 16 o más |
| FRT |
| 0 a 1 | B | B | M |
| 2 | B | M | A |
| 3 o más | M | A | A |

*Tabla 1: Matriz de complejidad de entradas externas.*

|  |  |  |  |
| --- | --- | --- | --- |
| DET | 1 a 5 | 6 a 19 | 20 o más |
| FRT |
| 0 a 1 | B | B | M |
| 2 a 3 | B | M | A |
| 4 o más | M | A | A |

*Tabla 2: Matriz de complejidad de salidas externas*

|  |  |  |  |
| --- | --- | --- | --- |
| DET | 1 a 4 | 5 a 15 | 16 o más |
| FRT |
| 0 a 1 | B | B | M |
| 2 | B | M | A |
| 3 o más | M | A | A |

*Tabla 3: Matriz de complejidad de consultas internas.*

|  |  |  |  |
| --- | --- | --- | --- |
| DET | 1 a 5 | 6 a 19 | 20 o más |
| FRT |
| 0 a 1 | B | B | M |
| 2 a 3 | B | M | A |
| 4 o más | M | A | A |

*Tabla 4: Matriz de complejidad de consultas externas.*

|  |  |  |  |
| --- | --- | --- | --- |
| DET | 1 a 19 | 20 a 50 | 51 o más |
| FRT |
| 0 a 1 | B | B | M |
| 2 | B | M | A |
| 6 o más | M | A | A |

*Tabla 5: Matriz de complejidad de archivos lógicos internos*

|  |  |  |  |
| --- | --- | --- | --- |
| DET | 1 a 19 | 20 a 50 | 51 o más |
| FRT |
| 0 a 1 | B | B | M |
| 2 | B | M | A |
| 6 o más | M | A | A |

*Tabla 6: Matriz de complejidad de interfaces externas.*

Luego de clasificar los elementos dependiendo su complejidad con la ayuda de las tablas se procede a asignar la totalidad de cada ítem como se muestra a continuación:

Entradas externas: 24

Salidas externas 10

Consultas externas: 4

Archivos lógicos Internos(tablas) 9

Interfaces externas: 1

Una vez obtenidos los valores se ubican en la tabla 7, en la columna contador para ser multiplicados con los parámetros de medida, después se realiza la sumatoria de la columna total como se realiza a continuación.

**Factores funcionales de peso.**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Factores de peso** | | | | |
| Factores Funcionales de peso | **Parámetros de medida (1)** | | | Contador | Total |
| Simple | Media | Compleja |
| **N.º Entrada de usuario** | 7 | 10 | 15 | 24 | 240 |
| **N.º Salida de usuario** | 5 | 7 | 10 | 10 | 70 |
| **N.º Consulta de usuario** | 3 | 4 | 6 | 4 | 16 |
| **N.º Archivo lógico Interno** | 4 | 5 | 7 | 9 | 36 |
| **N.º Interfaces externas** | 3 | 4 | 6 | 1 | 4 |
| **Total** | 366 | | | | |

*Tabla 7: Factores funcionales de peso.*

Al obtenerel total de los factores funcionales de peso se realiza el cálculo de los puntos de función para ello se hará uso la siguiente formula:

**Fórmula:**

**PF =**

**Aplicando la fórmula se tiene:**

**= 295**

Si se desea calcular el tiempo aproximado que durara el proyecto a partir de los puntos de función, tomando en cuenta que el número de desarrolladores que son 8.

Estimación para la duración del proyecto.

El desarrollo del proyecto durara 295 horas entre los 8 desarrolladores.

Se procede a calcular el tiempo en días y meses.

El tiempo estimado de desarrollo seria de 2 meses trabajando los 8 desarrolladores de lunes a viernes 8 horas al diarias.

Para encontrar los KLOC (miles de líneas de código) se multiplica los *Pf* (puntos de función) por la tabla 8, que se conoce como factores de correlación por lenguaje de programación. Para este ejercicio se utiliza el factor promedio del lenguaje de programación C# el cual tiene un valor de 58.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Lenguaje** | **Líneas de código/Puntos de función.** | | | |
| **….** | **Pro** | **Med** | **Min** | **Máx.** |
| **ASP** | 56 | 50 | 32 | 106 |
| **Assember** | 209 | 203 | 91 | 320 |
| **C** | 148 | 107 | 22 | 704 |
| **C++** | 59 | 53 | 20 | 178 |
| **C#** | 58 | 59 | 51 | 66 |
| **FoxPro** | 36 | 35 | 34 | 38 |
| **J2EE(Java)** | 57 | 50 | 50 | 67 |
| **Java** | 55 | 53 | 9 | 214 |
| **JavaScript** | 54 | 55 | 45 | 63 |
| **JSP** | 59 | - | - | - |
| **.NET** | 60 | 60 | 60 | 60 |
| **Perl** | 57 | 57 | 45 | 60 |
| **PL/SQL** | 47 | 39 | 16 | 78 |

*Tabla 8: Tabla de factores QSM por lenguaje de Programación* (Ordoñez1, 2015)

Luego de seleccionar el factor de correlación se procede a realizar la siguiente operación.

**COCOMO I.**

Para Aguilera Ángel, Clemente Jesús, Gil Eduardo y Hernández Ricardo (2017) mencionan que:

El modelo de costos de construcción (o COCOMO para abreviar, COnstructive COst MOdel para abreviar) es un modelo matemático basado en la experiencia que se utiliza para estimar los costos de software1. Incluye tres submodelos, cada uno de los cuales proporciona niveles cada vez más altos de detalle y aproximación a medida que avanza el proceso de desarrollo de software: básico, intermedio y detallado. (pág. 4)

Por un lado, COCOMO define tres modos de desarrollo o tipos de proyectos:

• **Orgánico:** proyectos relativamente sencillos, (menores de 50 KLOC) líneas de código, en los cuales se tiene experiencia de proyectos similares y se encuentran en entornos estables.

**• Semi-acoplado:** proyectos intermedios en complejidad y tamaño (menores de 300 KLOC), donde la experiencia en este tipo de proyectos es variable, y las restricciones intermedias.

**• Empotrado:** proyectos bastante complejos, en los que apenas se tiene experiencia y se engloban en un entorno de gran innovación técnica. Además, se trabaja con unos requisitos muy restrictivos y de gran volatilidad (mayores de 300 KLOC).

**COCOMO Básico.**

Para Gómez, López, Migani y Otazú (2000) mencionan que:

El Modelo Básico de COCOMO’81 estima el esfuerzo y el tiempo empleado en el desarrollo de un proyecto de software dando uso dos variables predictivas denominadas factores de costo (cost drivers): el tamaño del software y el modo de desarrollo. (pág. 9)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Modo** | **a** | **b** | **c** | **d** |
| Orgánico | 2.5 | 1.05 | 2.50 | 0.38 |
| Semi-acoplado | 3.0 | 1.12 | 2.50 | 0.35 |
| Empotrado | 3.6 | 1.20 | 2.50 | 0.32 |

*Tabla 9: Modelo de desarrollo básico.*

En este caso los KLOC es igual a 17 por lo que se elige el modelo orgánico.

|  |  |
| --- | --- |
| **Esfuerzo:** |  |
| **Duración:** |  |
| **Persona:** | Esfuerzo/duración |
| **Costo:** | Persona\*Salario |

*Tabla 10: Formula para modelo básico. Modo orgánico estimado por Boehm.*

Se reemplaza la formula por los datos obtenidos.

**COCOMO Intermedio.**

Aguilera Ángel, Clemente Jesús, Gil Eduardo y Hernández Ricardo

Este añade al modelo básico quince modificadores opcionales para tener en cuenta en el entorno de trabajo, incrementando así la precisión de la estimación. Para realizar este ajuste, al resultado de la fórmula general se lo multiplica por el coeficiente surgido de aplicar los atributos que se decidan utilizar. (2017, pág. 6).

Este modelo tiene tres submodelos para realizar la estimación los cuales se mencionan en la tabla 11 con su respectiva formula.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Modo** | **a** | **b** | **c** | **d** |
| Orgánico | 3.2 | 1.05 | 2.50 | 0.38 |
| Semi-acoplado | 3.0 | 1.12 | 2.50 | 0.35 |
| Empotrado | 2.8 | 1.20 | 2.50 | 0.32 |

*Tabla 11: Modelo de desarrollo intermedio.*

**Factores de costo.**

Para realizar la estimación con el modelo COCOMO intermedio se debe hacer uso de la tabla 12 la cual consta de cuatro categorías en donde se selecciona cada uno de los atributos con el que cuenta para realizar la estimación del sistema. Luego de seleccionar los atributos se realiza la sumatoria de los mismos.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Atributos** | **valor** | | | | | |
| **Muy bajo** | **bajo** | **Nominal** | **Alto** | **Muy alto** | **Extra alto** |
| **Atributos de software** | | | | | | |
| Fiabilidad | 0.75 | 0.88 | 1 | 1.15 | 1.4 |  |
| Tamaño de base de datos |  | 0.94 | 1 | 1.08 | 1.16 |  |
| Complejidad | 0.7 | 0.85 | 1 | 1.15 | 1.3 | 1.65 |
| **Atributos de hardware** | | | | | | |
| Restricciones de tiempo de ejecución |  |  | 1 | 1.1 | 1.3 | 1.66 |
| Restricciones de memoria virtual |  |  | 1 | 1.06 | 1.21 | 1.56 |
| Volatilidad de la máquina virtual | - | 0.87 | 1 | 1.15 | 1.3 |  |
| Tiempo de respuesta |  | 0.87 | 1 | 1.15 | 1.3 |  |
| **Atributos de personal** | | | | | | |
| Capacidad de análisis | 1.46 | 1.19 | 1 | 0.86 | 0.71 |  |
| Experiencia en la aplicación | 1.29 | 1.13 | 1 | 0.91 | 0.82 |  |
| Calidad de los programadores | 1.42 | 1.17 | 1 | 0.86 | 0.7 |  |
| Experiencia en la máquina virtual | 1.21 | 1.1 | 1 | 0.9 |  |  |
| Experiencia en el lenguaje | 1.14 | 1.07 | 1 | 0.95 |  |  |
| **Atributos del proyecto** | | | | | | |
| Técnicas actualizadas de programación | 1.24 | 1.1 | 1 | 0.91 | 0.82 |  |
| Utilización de herramientas de software | 1.24 | 1.1 | 1 | 0.91 | 0.83 |  |
| Restricciones de tiempo de desarrollo | 1.24 | 1.08 | 1 | 1.04 | 1.1 |  |
|  | | | | | | |

*Tabla 12: Factores de costo.*

**Multiplicador.**

El multiplicador es necesario para obtener el esfuerzo nominal, para obtener el multiplicador se debe multiplicar todos los atributos que selecciono en la tabla 12. Esto permite encontrar el factor de esfuerzo ajustado al entorno (FAE).

En este caso los KLOC es igual a 17 por lo que se elige el modo orgánico.

|  |  |
| --- | --- |
| **Esfuerzo:** |  |
| **Duración:** |  |
| **Persona:** | Esfuerzo/duración |
| **Costo:** | Persona\*Salario |

*Tabla 13: Formula para modelo intermedio. Modo orgánico.*

**COCOMO II.**

Según Marianela Acosta (2016) en su presentación de ingeniería en software menciona que:

Este modelo permite realizar estimaciones en función del tamaño del software, y de un conjunto de factores de costo y de escala. Los factores de costo describen aspectos relacionados con la naturaleza del producto, hardware utilizado, personal involucrado, y características propias del proyecto. El conjunto de factores de escala explica las economías y des economías de escala producidas a medida que un proyecto de software incrementa su tamaño. (pág. 1)

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

**Modelo de composición de aplicación.**

Se utiliza cuando el sistema es creado mediante componentes reutilizables, scripts y programación de base de datos. Este modelo utiliza como variable de medida del tamaño del producto los puntos de objeto, estos son básicamente el número de pantallas, informes y los módulos que son desarrollados en lenguajes de tercera generación en la aplicación.

|  |  |  |  |
| --- | --- | --- | --- |
| Tipo Objeto | Peso de complejidad. | | |
| Básico. | Intermedio. | Avanzado. |
| Pantalla | 1 | 2 | 3 |
| Informe | 2 | 5 | 8 |
| Componentes L3G |  |  | 10 |

*Tabla 14: Factores de peso*.

En el sistema que se encuentra en desarrollo tiene 22 pantallas 7 tipos de informe. El código que es reutilizable es de un 45%.

Para el calculo de la productividad se utiliza la tabla 15.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Experiencia/capacidad de desarrollador. | Muy baja | Baja | Normal | Alta | Muy alta |
| Madurez/capacidad del entorno. | Muy baja | Baja | Normal | Alta | Muy alta |
| Productividad (PROD) | 4 | 7 | 13 | 25 | 50 |

*Tabla 15: proporciones de productividad.*

El esfuerzo estimado de los puntos de objeto referentes a su productividad es de 2.46.

**Modelo de diseño temprano.**

Este modelo se usa en las etapas tempranas de un proyecto de software, cuando se conoce muy poco del tamaño del producto a ser desarrollado, de la naturaleza de la plataforma, del personal a ser incorporado al proyecto o detalles específicos del proceso a utilizar. Este modelo podría emplearse tanto en productos desarrollados en sectores de Generadores de Aplicación, Sistemas Integrados o Infraestructura

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Factor de  Escala Wj | Muy Bajo | Bajo | Normal | Alto | Muy Alto | Extra |
| Precedencia  PREC | Completamente  Sin procedente  (6.20) | Ampliamente  Sin precedentes  (4.96) | Algún  Precedente  (3.72) | Generalmente  Familiar  (2.40) | Ampliamente Familiar  (1.24) | Completamente  Familiar  (0.00) |
| Flexibilidad  En el  Desarrollo  FLEX | Rigurosa  (5.07) | Relajación  Ocasional  (4.05) | Alguna Relajación  (3.04) | Generalmente  (75%)  (2.03) | Alguna  Conformidad  (1.01) | Metas  Generales  (0.00) |
| Arquitectura/  Resolución  De riesgo  RESL | Poca  (20%)  (7.07) | Alguna  (40%)  (5.65) | Siempre  (60%)  (4.24) | Ampliamente  Cooperativas  (2.19) | Principalmente  (90%)  (1.41) | Completo  (100%)  (0.00) |
| Cohesión de  Equipo  TEAM | Interacciones  Difíciles  (5.48) | Interacciones  Con alguna  Dificultad  (4.38) | Interacciones  Básicamente  (3.29) | Ampliamente  Cooperativas  (2.19) | Altamente  Cooperativas  (1.10) | Interacciones  Sin Fisuras  (0.00) |
| Madurez del  Proceso  PMAT | (7.80) | (6.24) | (4.68) | (3.12) | (1.56) | (0.0) |

*Tabla 16: Factores de Escala.*

Se procede a sacar los factores de escala, estos son el resultado de sumatoria de cada uno de los factores que se encuentran marcados en la tabla 14.

B es el factor exponencial de escala.

A es una constante que captura los efectos lineales sobre el esfuerzo de acuerdo a la variación del tamaño, (A=2.94)

es el esfuerzo expresado en meses personas.

es el esfuerzo nominal.

**= A \***

Para obtener el valor nominal se hace uso de la siguiente formula:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Early Desing  Cost Drivers | Extra  Low | Very  Low | Low | Nominal | High | Very  High | Extra  High |
| RCPX | 0.73 | 0.81 | 0.98 | 1.0 | 1.30 | 1.74 | 2.38 |
| RUSE | - | - | 0.95 | 1.0 | 1.07 | 1.15 | 1.24 |
| PDIF | - | - | 0.87 | 1.0 | 1.29 | 1.81 | 2.61 |
| PERS | 2.12 | 1.62 | 1.26 | 1.0 | 0.83 | 0. 63 | 0.50 |
| PREX | 1.59 | 1.33 | 1.12 | 1.0 | 0.87 | 0.71 | 0.62 |
| FCIL | 1.43 | 1.30 | 1.10 | 1.0 | 0.87 | 0.73 | 0.62 |
| SCED | - | 1.43 | 1.14 | 1.0 | 1.0 | 1.0 | - |

*Tabla 17: Conductores de costo para diseño inicial estimados por Boehm.*

**Modelo Post-Arquitectura.**

Según Erika Meneses (2017) en su documento menciona que:

Se aplica en la etapa de desarrollo, después que se define la arquitectura del sistema y en la etapa de mantenimiento. Este modelo utiliza: Puntos Función y/o Líneas de Código Fuente para estimar tamaño, con modificadores que contemplan el reusó y el "desperdicio" (breakage). (pág. 11)

Utiliza un conjunto de 17 atributos, denominados factores de costo (personal, plataforma de desarrollo, etc.) Cinco factores que determinan un exponente que reemplazan los modos Orgánico, Semi acoplado y Empotrado del modelo COCOMO '81.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Early Deming  Cost Drivers | Very  Low | Low | Nominal | High | Very  High | Extra  High |
| RELY | 0.75 | 0.88 | 1.00 | 2.48 | 1.24 | 0.00 |
| DATA | - | 0.93 | 1.00 | 2.03 | 2.03 | 0.00 |
| CPLX | 0.75 | 0.88 | 1.00 | 2.83 | 1.41 | 0.00 |
| RUSE | - | 0.91 | 1.00 | 2.19 | 1.10 | 0.00 |
| DOCU | 0.89 | 0.95 | 1.00 | 3.12 | 1.56 | 0.00 |
| TIME | - | - | 1.00 | 1.11 | 1.31 | 1.67 |
| STOR | - | - | 1.00 | 1.06 | 1.21 | 1.57 |
| PVOL | - | 0.87 | 1.00 | 1.15 | 1.30 | - |
| ACAP | 1.50 | 1.22 | 1.00 | 0.83 | 0.67 | - |
| PCAP | 1.37 | 1.16 | 1.00 | 0.87 | 0.74 | - |
| PCON | 1.24 | 1.10 | 1.00 | 0.92 | 0.84 | - |
| AEXP | 1.22 | 1.10 | 1.00 | 0.89 | 0.81 | - |
| PEXP | 1.25 | 1.12 | 1.00 | 0.88 | 0.81 | - |
| LTEX | 1.22 | 1.10 | 1.00 | 0.91 | 0.84 | - |
| TOOL | 1.24 | 1.12 | 1.00 | 0.86 | 0.72 | - |
| SITE | 1.25 | 1.10 | 1.00 | 0.92 | 0.84 | 0.78 |
| SCED | 1.29 | 1.10 | 1.00 | 1.00 | 1.00 | - |

*Tabla 18: Conductores de costo en Post-Arquitectura. estimado por Boehm.*

**Estimación de tiempo de desarrollo.**

La estimación del tiempo de desarrollo es de 2 meses.

**Conclusión.**

Luego de realizar el desarrollo del sistema y los temas que se abordaron se puede mencionar que:

* El IDE visual studio 2019 permite agregar extensiones de aplicaciones para realizar comentarios del código de manera automática y a partir de estos generar la documentación detallada. Además, se puede agregar paquetes y librerías para un mejor diseño de interfaces graficas de usuario.
* MySQL es un sistema de gestión de base de datos muy eficiente al responder consultas concurrentes de varios usuarios.
* La metodología ágil Scrum permitió mantener la organización dentro del grupo de desarrollo, aunque en la mayoría de sprint se debía mandar a realizar correcciones porque no se cumplía con lo que se pedía.
* Al realizar la estimación con COCOMO’ 81 mediante el modelo básico modo orgánico se obtuvo datos elevados en esfuerzo, duración y coste que es una cantidad exagerada que llega a un monto total de $247,500.00.
* Mediante la estimación de COCOMO’ 81 modelo intermedio modo orgánico se obtuvo valores superiores al mencionado anteriormente esto debido a que este modelo se le agregar especificaciones de las características con las que se cuenta la empresa que desarrolla el software esto permite al dueño de la empresa evitar pérdidas, pero el coste del producto de software es muy elevado en este caso el monto total era de $617.400.
* Al realizar la estimación con COCOMO II con el modelo de diseño temprano se obtuvo resultados significativos como como el numero de personas al mes que era de 51.
* Al realizar la estimación con COCOMO II con el modelo post-arquitectura se obtuvo que la cantidad de personas a trabajar en el sistema al mes eran de 52 y al realizar la estimación del tiempo del tiempo de desarrollo se obtuvo como resultado un total de 2 meses por lo que el coste del sistema seria mas atractivo y accesible. El monto del producto de software seria de $ 93,600.00.
* El modelo de diseño temprano muestra una menor cantidad de personas al mes esto es debido a que en el modelo post-arquitectura cuenta con una cantidad mayor de factores que son esenciales para obtener mayor precisión en sus resultados. Algo principal a tener en cuenta es modo de estimación de resultados porque es aquí donde se reduce el tiempo de desarrollo debido a la cantidad de código que será reutilizable dentro del sistema.
* En este el caso el sistema que se desarrolló con 8 desarrolladores se trabajó 2 días a la semana por un lapso aproximado de 2 dos meses, por lo que el tiempo de estimación obtenido con los modelos de COCOMO II son mas precisos.
* COCOMO II permite obtener mejores resultados al momento de realizar estimaciones para un sistema que se pretenda desarrollar.

# **Bibliografía**

Acosta, M. (29 de Abril de 2016). *COCOMO II*. Recuperado el 10 de Marzo de 2022, de https://es.slideshare.net/marianela0393/cocomo-ii-61519624

Adriana Gómez, María del C.López,Silvina Migani & Alejandra Otazú. (02 de Junio de 2000). *UN MODELO DE ESTIMACION DE PROYECTOS DE SOFTWARE.* Recuperado el 09 de Marzo de 2022, de file:///C:/Users/USUARIO/Downloads/cocom0llfull.pdf

Aguilera Ángel, Clemente Jesús, Gil Eduardo y Hernández Ricardo. (2017). *Studio Document.* Recuperado el 11 de Marzo de 2022, de www.studocu.com/es-mx/document/instituto-tecnologico-superior-de-apatzingan/gestion-de-proyectos-de-software/modelo-cocomo-ensayo/10508201

Glee, T. (16 de Febrero de 2022). Visual S. 3. Recuperado el 10 de Marzo de 2022, de docs.microsoft.com/es-es/visualstudio/get-started/visual-studio-ide

Luque, J. (13 de Enero de 2015). *Universitat Oberta de Catalunya.* Recuperado el 09 de Marzo de 2022, de http://openaccess.uoc.edu/webapps/o2/bitstream/10609/40646/1/javvvi\_TFC\_0115.pdf

Meneses, E. (02 de Octubre de 2017). *TECNOLOGIAS PARA LA INTEGRACION DE SOLUCIONES.* Recuperado el 10 de Marzo de 2022, de https://www.uv.mx/personal/ermeneses/files/2017/08/Clase9-COCOMOII.pdf

NeoAttack. (27 de Agosto de 2020). *Neo Attack.* Recuperado el 10 de Marzo de 2022, de neoattack.com/neowiki/mysql/

Ordoñez1, Á. F. (2015). Mejoramiento en la productividad de software por la adaptación. Obtenido de http://scielo.senescyt.gob.ec/pdf/enfoqueute/v6n2/1390-6542-enfoqueute-6-02-00117.pdf

Sanchez, F. (2011). *Los Puntos de Funcionalidad.* infor.uva.es. Recuperado el 09 de Marzo de 2022, de https://www.infor.uva.es/~manso/calidad/PFA-CLM-2011#:~:text=Los%20Puntos%20Función%20constituyen%20una,y%20el%20coste%20del%20software.

Trigas, M. (25 de septiembre de 2012). *Openaccess.* Recuperado el 9 de Marzo de 2022, de http://openaccess.uoc.edu/webapps/o2/bitstream/10609/17885/1/mtrigasTFC0612memoria.pdf