



INSTITUTO TECNOLÓGICO SUPERIOR DE ALAMO TEMAPACHE

CARRERA:

INGENIERIA EN SISTEMAS COMPUTACIONALES

ASIGNATURA:

PROCESO PERSONAL DE DESARROLLO DE SOFTWARE (PSP)

NOMBRE DE LA PRÁCTICA:

APLICACIÓN DEL PROCESO PSP1.1

NÚMERO DE LA PRÁCTICA:

04

INTEGRANTES DEL EQUIPO:

GONZALO MARTÍNEZ SILVERIO

DOCENTE:

DRA TANIA TURRUBIATES LOPEZ

SEMESTRE

7

GRUPO:

1A



INSTITUTO TECNOLÓGICO SUPERIOR DE ALAMO TEMAPACHE

INTRODUCCIÓN:

PSP1.1 se enfoca en la mejora de la gestión de proyectos de desarrollo de software, específicamente en la planificación y seguimiento del tiempo y recursos. Los objetivos incluyen comprender nuevos elementos del proceso, aprender a utilizar plantillas de planificación de tareas y horarios, y estar preparado para aplicar el PSP1.1 en un proyecto.

El PSP1.1 introduce conceptos como el índice de desempeño de costos (CPI) y porcentajes relacionados con la reutilización de código. Además, se detalla cómo registrar y gestionar tareas y horarios para estimar la finalización del trabajo. En resumen, este tutorial proporciona herramientas y conocimientos esenciales para una gestión efectiva de proyectos de desarrollo de software.

OBJETIVO:

Familiarizar a los usuarios con las prácticas y herramientas necesarias para la planificación, seguimiento y gestión eficaz de proyectos de desarrollo de software. Esto incluye la comprensión de nuevos elementos del proceso, la utilización de plantillas de planificación de tareas y horarios, y la preparación para aplicar el PSP1.1 en un proyecto específico.

COMPETENCIA A DESARROLLAR:

- Que el alumno conozca su ritmo de trabajo y pueda hacer una evaluación del tiempo que tarda y con respecto a ello conozca su ritmo de trabajo en cada etapa.
- Planeación de tiempo y calendario

MATERIAL Y EQUIPO (REQUERIMIENTOS):

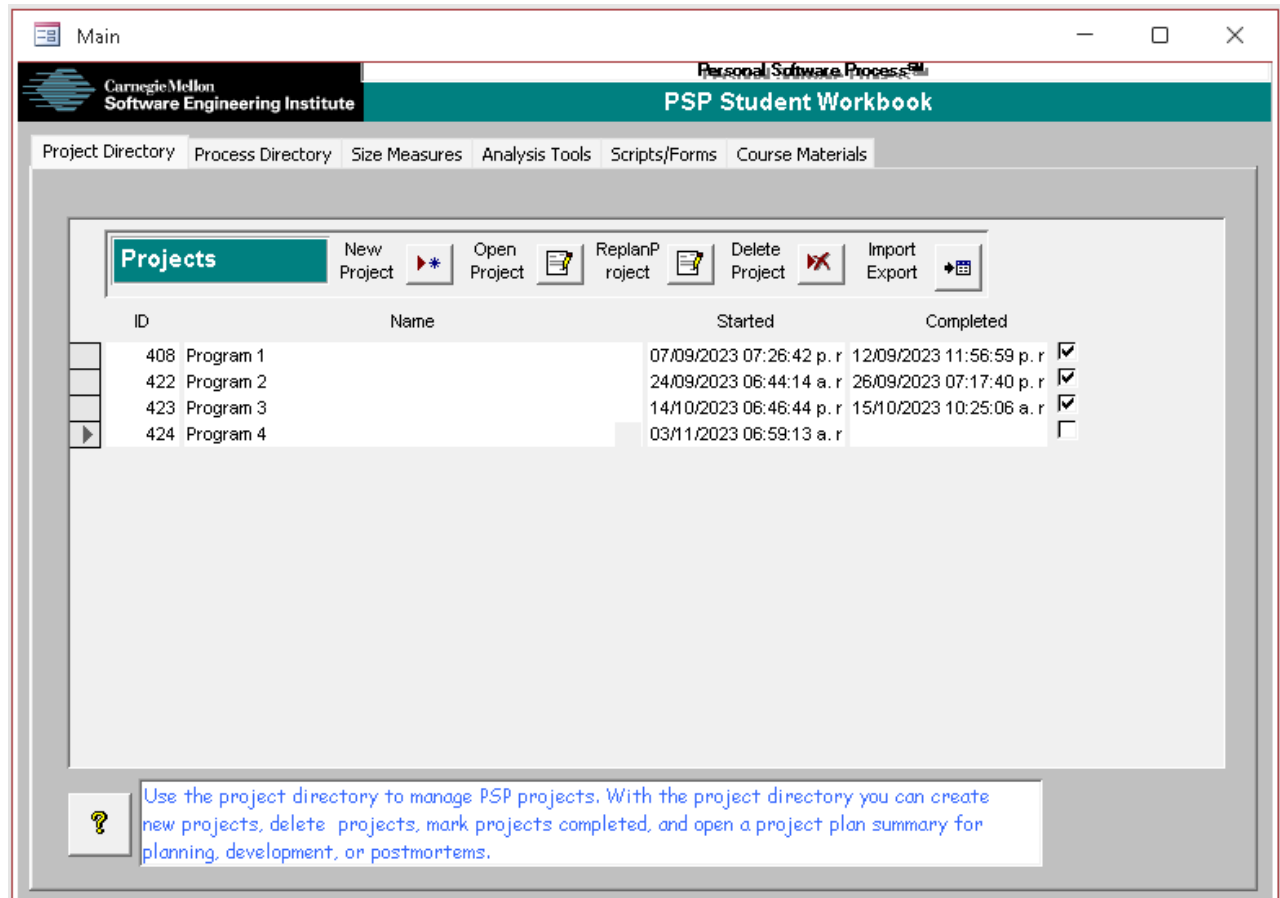
• Laptop personal.
• Microsoft Office Word.
• JDK 8u202.
• IDE NetBeans 8.2.
• Material de prácticas de PSP.



INSTITUTO TECNOLÓGICO SUPERIOR DE ALAMO TEMAPACHE

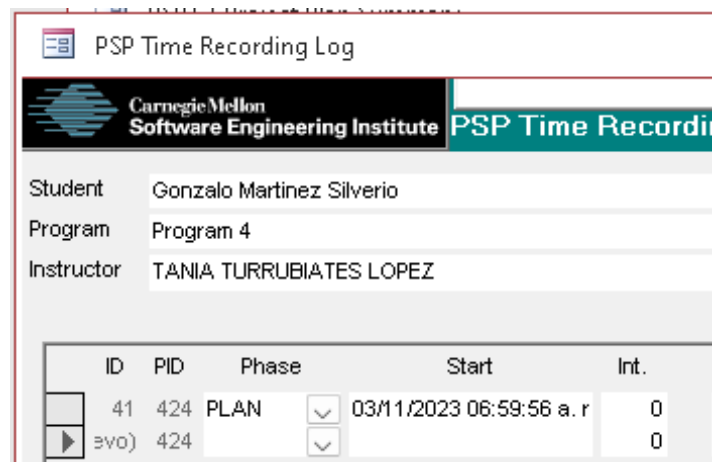
DESARROLLO:

Primero registramos el inicio de la etapa de planeación en la herramienta **PSP Student Workbook.mde**:



ETAPA DE PLANEACION:

Al igual que en el paso anterior se debe registrar en el time log el inicio de la etapa de planeación:





INSTITUTO TECNOLÓGICO SUPERIOR DE ALAMO TEMAPACHE

1.-Requerimientos del programa:

Usando PSP1.1, escriba un programa para calcular rangos de tamaño relativo para objetos muy pequeños, rangos pequeños, medianos, grandes y muy grandes utilizando la desviación estándar. Pruebe minuciosamente el programa. Pruebe el programa utilizando los datos proporcionados en tablas 1 y 2. Los valores esperados se incluyen en la tabla 3.

Class Name	Class LOC	Number of Methods
each_char	18	3
string_read	18	3
single_character	25	3
each_line	31	3
single_char	37	3
string_builder	82	5
string_manager	82	4
list_clump	87	4
list_clip	89	4
string_decrementer	230	10
Char	85	3
Character	87	3
Converter	558	10

Table 1. LOC/Method Data

Chapter	Pages
Preface	7
Chapter 1	12
Chapter 2	10
Chapter 3	12
Chapter 4	10
Chapter 5	12
Chapter 6	12
Chapter 7	12
Chapter 8	12
Chapter 9	8
Appendix A	8
Appendix B	8
Appendix C	20
Appendix D	14
Appendix E	18
Appendix F	12

Table 2. Pgs/Chapter

	VS	S	M	L	VL
LOC/Method	4.3953	8.5081	16.4696	31.8811	61.7137
Pgs/Chapter	6.3375	8.4393	11.2381	14.9650	19.9280

Table 3. Expected Values

En PSP, las tablas de tamaños relativos se utilizan para brindarle un marco para juzgar el tamaño de las piezas nuevas en sus productos planificados. Por ejemplo, si conoce los tamaños de todas las piezas de un determinado tipo desarrolladas previamente, podrá juzgar mejor el tamaño probable de una nueva pieza de ese tipo. El procedimiento de desviación estándar descrito en la siguiente sección le permite equilibrar sus estimaciones para que se ajusten más o menos a la distribución normal.

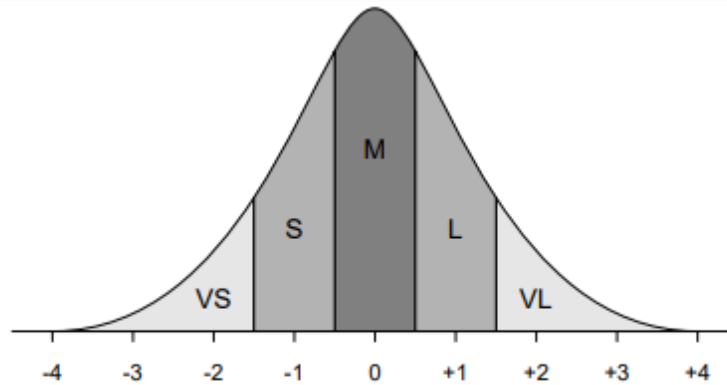


Figure 1. Ranges of standard deviations

El rango medio (M) es el área entre -0,5 desviaciones estándar y +0,5 desviaciones estándar de la media, como se muestra en la Figura 1. Suponiendo que los datos se aproximan a una distribución normal, el número probable de partes que están dentro de más o menos 0,5 La desviación estándar del valor promedio es del 38,3 por ciento. Siguiendo una lógica similar, el área de porcentajes del rango es la siguiente:

- 6,68 % debería ser muy pequeño
- 24,17% debería ser pequeño
- 38,2% debería ser medio
- 24,17% debería ser grande
- 6,68% debería ser muy grande

El método de estimación PROBE divide los datos históricos de tamaño en categorías que representan su tipo de trabajo. Una forma de hacerlo se basa en la desviación estándar. Primero, divida sus datos históricos en categorías funcionales, cada una de las cuales tenga al menos de 6 a 8 miembros (cálculo, texto y datos, por ejemplo). Para cada categoría, puede calcular los rangos de tamaño relativos para VS, S, M, L y VL siguiendo el procedimiento siguiente.

Divida los tamaños de las piezas por la cantidad de artículos en cada pieza para determinar el tamaño por artículo, si corresponde. Por ejemplo, es posible que no tenga suficientes datos sobre clases para desarrollar una tabla de tamaño relativo, pero sí tiene suficientes datos sobre métodos. En lugar de usar LOC total por clase, puede usar LOC/método.

A continuación, deberá transformar los datos con normalidad de registro. Esto es necesario porque no se pueden tener tamaños negativos y los valores más pequeños tienden a agruparse. La



INSTITUTO TECNOLÓGICO SUPERIOR DE ALAMO TEMAPACHE

transformación de datos con normalidad logarítmica le permite trazar los datos alrededor de una media de cero. Para cada valor de tamaño, X_i tome el logaritmo natural, \ln , para obtener:

$$\ln(x_i)$$

Calcule el promedio de estos n valores logarítmicos:

$$avg = \frac{\sum_{i=1}^n \ln(x_i)}{n}.$$

Calcule la varianza de estos valores:

$$var = \sigma^2 = \frac{\sum_{i=1}^n (\ln(x_i) - avg)^2}{(n-1)}.$$

Calcula la desviación estándar:

$$\sigma = \sqrt{var}.$$

Calcula los rangos logarítmicos:

$$\ln(VS) = avg - 2\sigma$$

$$\ln(S) = avg - \sigma$$

$$\ln(M) = avg$$

$$\ln(L) = avg + \sigma$$

$$\ln(VL) = avg + 2\sigma$$

Por último, convierta los valores logarítmicos naturales a su forma original calculando el antilogaritmo (calcule e elevado a la potencia del valor logarítmico) para obtener los puntos medios de los rangos de tamaño.

$$VS = e^{\ln(VS)}$$

$$S = e^{\ln(S)}$$

$$M = e^{\ln(M)}$$

$$L = e^{\ln(L)}$$

$$VL = e^{\ln(VL)}$$



2.- Estimación de tiempo:

Teniendo en cuenta que los requerimientos del proyecto son bastante específicos y su estructura es poco compleja, se estima que todas sus etapas se completarán en un tiempo de 4 horas. Sin embargo, este es un estimado y puede variar dependiendo de factores imprevistos que puedan surgir durante el desarrollo del proyecto.

3.-Registramos la conclusión de la etapa de planeación registrándolo en el Time Log:

The screenshot shows the 'PSP Time Recording Log' window. The header includes the Carnegie Mellon Software Engineering Institute logo and the title 'PSP Time Recording Log'. The student information is: Student: Gonzalo Martinez Silverio, Program: Program 4, Instructor: TANIA TURRUBIATES LOPEZ. The start date is 03-nov.-23, and the language is Java. The table below shows the recording log:

ID	PID	Phase	Start	Int.	Stop Date and Time	Delta	Comments
41	424	PLAN	03/11/2023 06:59:56 a. r	0	03/11/2023 07:20:16 a. r	20.3	Planeacion

ETAPA DE DISEÑO:

1.-Se registra el inicio de la etapa de diseño en el Time Log:

The screenshot shows the 'PSP Time Recording Log' window. The student information is: Student: Gonzalo Martinez Silverio, Program: Program 4, Instructor: TANIA TURRUBIATES LOPEZ. The table below shows the recording log:

ID	PID	Phase	Start
41	424	PLAN	03/11/2023 06:59:56 a. r
43	424	DLD	03/11/2023 07:20:17 a. r
424	424		

2.-El diseño que se presentará será el siguiente:

La clase tendrá dos métodos públicos: que toma un arreglo de cadenas como argumento y que toma un mapa con claves de cadena y valores numéricos y devuelve un nuevo mapa con claves de cadena y valores en punto flotante. Este diagrama de clases UML proporcionara una visión simplificada de la estructura del programa y sus relaciones principales. Las clases internas de Java no se han representado en el diagrama por simplicidad, ya que son parte de las bibliotecas estándar de Java.



INSTITUTO TECNOLÓGICO SUPERIOR DE ALAMO TEMAPACHE

Las clases quedarían estructuradas de la siguiente manera:

PROGRAM4
- datosLocMetodo: Map<String, Double>
- datosPagsCapitulo: Map<String, Integer>
+ main(args: String[]): void
+ calcularTamañoRelativo(datos: Map<String, ? extends Number>): Map<String, Double>

3.-Finalizamos la etapa de Diseño y la registramos en el Time Log:

ID	PID	Phase	Start	Int.	Stop Date and Time	Delta	Comments
41	424	PLAN	03/11/2023 06:59:56 a. r	0	03/11/2023 07:20:16 a. r	20.3	Planeacion
43	424	DLD	03/11/2023 07:20:17 a. r	0	03/11/2023 07:50:21 a. r	30.1	Diseño de diagrama UML

ETAPA DE CODIFICACION:

Registramos el inicio de la etapa de codificación en nuestro Time Log:

ID	PID	Phase	Start	Int.
41	424	PLAN	03/11/2023 06:59:56 a. r	0
43	424	DLD	03/11/2023 07:20:17 a. r	0
44	424	CODE	03/11/2023 07:50:22 a. r	0

Abrimos NetBeans, creamos un nuevo proyecto, creamos una clase y comenzamos a codificar:

```
Start Page x PROGRAM4.java x
Source History
1 package PROGRAM4;
2 /**
3  * INSTITUTO TECNOLÓGICO SUPERIOR DE ALAMO TEMAPACHE
4  * @author Gonzalo Martinez Silverio
5  * nm635542@gmail.com
6  */
```




INSTITUTO TECNOLÓGICO SUPERIOR DE ALAMO TEMAPACHE

```
7 import java.util.HashMap;
8 import java.util.Map;
9 import java.util.stream.DoubleStream;
10
11 public class PROGRAM4 {
12     public static void main(String[] args) {
13         // Crear un HashMap llamado datosLocMetodo para almacenar datos relacionados con LOC/Método
14         Map<String, Double> datosLocMetodo = new HashMap<>();
15         datosLocMetodo.put("cada_caracter", 18.0 / 3.0);
16         datosLocMetodo.put("lectura_cadena", 18.0 / 3.0);
17         datosLocMetodo.put("caracter_individual", 25.0 / 3.0);
18         datosLocMetodo.put("cada_linea", 31.0 / 3.0);
19         datosLocMetodo.put("caracter_solo", 37.0 / 3.0);
20         datosLocMetodo.put("constructor_cadena", 82.0 / 5.0);
21         datosLocMetodo.put("gestor_cadena", 82.0 / 4.0);
22         datosLocMetodo.put("grupo_lista", 87.0 / 4.0);
23         datosLocMetodo.put("recorte_lista", 89.0 / 4.0);
24         datosLocMetodo.put("decrementador_cadena", 230.0 / 10.0);
25         datosLocMetodo.put("Caracter", 85.0 / 3.0);
26         datosLocMetodo.put("Carácter", 87.0 / 3.0);
27         datosLocMetodo.put("Convertidor", 558.0 / 10.0);
28         // Crear un HashMap llamado datosPagsCapitulo para almacenar datos relacionados con Páginas/Capítulo
29         Map<String, Integer> datosPagsCapitulo = new HashMap<>();
30         datosPagsCapitulo.put("Prefacio", 7);
31         datosPagsCapitulo.put("Capítulo 1", 12);
32         datosPagsCapitulo.put("Capítulo 2", 10);
33         datosPagsCapitulo.put("Capítulo 3", 12);
34         datosPagsCapitulo.put("Capítulo 4", 10);
35         datosPagsCapitulo.put("Capítulo 5", 12);
36         datosPagsCapitulo.put("Capítulo 6", 12);
37         datosPagsCapitulo.put("Capítulo 7", 12);
38         datosPagsCapitulo.put("Capítulo 8", 12);
39         datosPagsCapitulo.put("Capítulo 9", 8);
40         datosPagsCapitulo.put("Apéndice A", 8);
41         datosPagsCapitulo.put("Apéndice B", 8);
42         datosPagsCapitulo.put("Apéndice C", 20);
43         datosPagsCapitulo.put("Apéndice D", 14);
44         datosPagsCapitulo.put("Apéndice E", 18);
45         datosPagsCapitulo.put("Apéndice F", 12);
46         // Calcular los tamaños relativos para datosLocMetodo y almacenar los resultados en un nuevo mapa
47         Map<String, Double> resultadoDatosLocMetodo = calcularTamañoRelativo(datosLocMetodo);
48         // Calcular los tamaños relativos para datosPagsCapitulo y almacenar los resultados en un nuevo mapa
49         Map<String, Double> resultadoDatosPagsCapitulo = calcularTamañoRelativo(datosPagsCapitulo);
50         // Imprimir los resultados para datosLocMetodo
51         System.out.println("Datos LOC/Método:");
52         for (String clave : new String[]{"VS", "S", "M", "L", "VL"}) {
53             System.out.println("LOC/Método: " + clave + " = " + resultadoDatosLocMetodo.get(clave));
54         }
55         // Imprimir los resultados para datosPagsCapitulo
56         System.out.println("\nDatos Páginas/Capítulo:");
57         for (String clave : new String[]{"VS", "S", "M", "L", "VL"}) {
58             System.out.println("Páginas/Capítulo: " + clave + " = " + resultadoDatosPagsCapitulo.get(clave));
59         }
60     }
61     // Este método calcula los tamaños relativos en base a los datos proporcionados en el mapa 'datos'
62     public static Map<String, Double> calcularTamañoRelativo(Map<String, ? extends Number> datos) {
63         // Transforma los valores en un arreglo de logaritmos naturales
64         double[] tamañosLn = datos.values().stream().mapToDouble(value -> Math.log(value.doubleValue())).toArray();
65         // Calcula la media de los logaritmos naturales
66         double media = DoubleStream.of(tamañosLn).average().getAsDouble();
67         // Calcula la varianza de los logaritmos naturales
68         double varianza = DoubleStream.of(tamañosLn).map(x -> Math.pow(x - media, 2)).sum() / (tamañosLn.length - 1);
69         // Calcula la desviación estándar (sigma) a partir de la varianza
70         double sigma = Math.sqrt(varianza);
71         // Crea un nuevo HashMap llamado rangosLn con categorías VS, S, M, L y VL basadas en la media y sigma
72         Map<String, Double> rangosLn = new HashMap<>();
73         rangosLn.put("VS", media - 2 * sigma);
74         rangosLn.put("S", media - sigma);
75         rangosLn.put("M", media);
```



INSTITUTO TECNOLÓGICO SUPERIOR DE ALAMO TEMAPACHE

```
76 rangosLn.put("L", media + sigma);
77 rangosLn.put("VL", media + 2 * sigma);
78 // Crea un nuevo HashMap llamado rangos que contiene los tamaños relativos reales
79 Map<String, Double> rangos = new HashMap<>();
80 for (Map.Entry<String, Double> entry : rangosLn.entrySet()) {
81     rangos.put(entry.getKey(), Math.exp(entry.getValue()));
82 }
83 // Retorna el mapa de tamaños relativos
84 return rangos;
85 }
86 }
```

3.-Marcamos como finalizada la etapa de codificación en el Time Log:

PSP Time Recording Log

CarnegieMellon
Software Engineering Institute

Personal Software ProcessSM

PSP Time Recording Log

Student

Gonzalo Martinez Silverio

Program

Program 4

Instructor

TANIA TURRUBIATES LOPEZ

Start Date

03-nov.-23

End Date

Language

Java

ID

424

ID	PID	Phase	Start	Int.	Stop Date and Time	Delta	Comments
41	424	PLAN	03/11/2023 06:59:56 a.r	0	03/11/2023 07:20:16 a.r	20.3	Planeacion
43	424	DLD	03/11/2023 07:20:17 a.r	0	03/11/2023 07:50:21 a.r	30.1	Diseño de diagrama UML
44	424	CODE	03/11/2023 07:50:22 a.r	0	03/11/2023 08:27:01 a.r	36.7	Codificación

ETAPA DE COMPILACION:

Se compila:

BUILD SUCCESSFUL (total time: 2 seconds)

</

Al finalizar esta etapa, debido a que el programa no recibe datos desde teclado se optó por llevar a cabo una compilación y después se realizará la prueba simultáneamente. Los de datos de la Tabla 1 y 2 están escritos en el código para que este realice los cálculos directamente.

ETAPA DE TESTEO:



INSTITUTO TECNOLÓGICO SUPERIOR DE ALAMO TEMAPACHE

Registramos el inicio de nuestra etapa de Testeo:

PSP Time Recording Log						
Carnegie Mellon Software Engineering Institute PSP Time Record						
Student		Gonzalo Martinez Silverio				
Program		Program 4				
Instructor		TANIA TURRUBIATES LOPEZ				
ID	PID	Phase	Start	Int.		
41	424	PLAN	03/11/2023 06:59:56 a.r	0		
43	424	DLD	03/11/2023 07:20:17 a.r	0		
44	424	CODE	03/11/2023 07:50:22 a.r	0		
45	424	COMPILE	03/11/2023 08:27:02 a.r	0		
46	424	UT	03/11/2023 10:51:59 a.r	0		
47	424			0		

Resultados:

Output - PSP (run)	
run:	
Datos LOC/Método:	
LOC/Método: VS = 4.395269124478683	
LOC/Método: S = 8.508138249389223	
LOC/Método: M = 16.46962095394006	
LOC/Método: L = 31.881053929269857	
LOC/Método: VL = 61.71372143193483	
Datos Páginas/Capítulo:	
Páginas/Capítulo: VS = 6.337517961211727	
Páginas/Capítulo: S = 8.439281112126054	
Páginas/Capítulo: M = 11.238069244993524	
Páginas/Capítulo: L = 14.965042481379415	
Páginas/Capítulo: VL = 19.928022473189486	

Los resultados fueron verificados y en efecto son correctos por lo tanto marcamos como finalizada la etapa de pruebas:

PSP Time Recording Log									
Carnegie Mellon Software Engineering Institute PSP Time Recording Log									
Student		Gonzalo Martinez Silverio					Start Date	03-nov.-23	
Program		Program 4					End Date		
Instructor		TANIA TURRUBIATES LOPEZ					Language	Java	
ID	PID	Phase	Start	Int.	Stop Date and Time	Delta	Comments		
41	424	PLAN	03/11/2023 06:59:56 a.r	0	03/11/2023 07:20:16 a.r	20.3	Planeacion		
43	424	DLD	03/11/2023 07:20:17 a.r	0	03/11/2023 07:50:21 a.r	30.1	Diseño de diagrama UML		
44	424	CODE	03/11/2023 07:50:22 a.r	0	03/11/2023 08:27:01 a.r	36.7	Codificacion		
45	424	COMPILE	03/11/2023 08:27:02 a.r	0	03/11/2023 08:27:15 a.r	0.2	Compilacion		
46	424	UT	03/11/2023 10:51:59 a.r	0	03/11/2023 11:04:01 a.r	12.0	Testeo		
47	424			0		0.0			



ETAPA POSMORTEM:

Registramos el inicio de la etapa:

ID	PID	Phase	Start	Int.
41	424	PLAN	03/11/2023 06:59:56 a. r	0
43	424	DLD	03/11/2023 07:20:17 a. r	0
44	424	CODE	03/11/2023 07:50:22 a. r	0
45	424	COMPILE	03/11/2023 08:27:02 a. r	0
46	424	UT	03/11/2023 10:51:59 a. r	0
47	424	PM	03/11/2023 11:05:02 a. r	0
▶ 48	424			0

En esta fase final, se lleva a cabo un análisis exhaustivo de nuestro proceso de desarrollo, centrándonos en cada etapa, para evaluar nuestro desempeño y los resultados obtenidos:

ETAPA DE PLANEACION:

Comenzamos con un análisis profundo de la descripción del problema del archivo ASGKIT PROG4, esto me permitió comprender a fondo los requisitos clave para nuestro programa. Basándonos en esta comprensión, desarrolle una estimación del tiempo necesario para completar el proyecto. Además, fue necesario establecer una estructura organizativa sólida para llevar a cabo un análisis para calcular los tamaños relativos para dos conjuntos de datos (LOC/Método y Páginas/Capítulo) utilizando estadísticas como la media y la desviación estándar. Durante esta etapa, como desarrollador tiendo a experimentar ansiedad y distracciones con facilidad. Por lo tanto, recomiendo trabajar en un entorno tranquilo (sin interrupciones), lo que contribuirá a mejorar la eficiencia y la concentración en el trabajo.

ETAPA DE DISEÑO:

En esta etapa en base a lo comprendido anteriormente se procedió a elaborar el diseño en base los requerimientos del programa, para lo cual se tuvo que desarrollar un diagrama UML. Las clases internas de Java no se han representado en el diagrama por simplicidad, ya que son parte de las bibliotecas estándar de Java.

ETAPA DE CODIFICACION:

En la etapa se procedió a implementar lo que había diseñado previamente. A pesar de no contar con un diseño detallado, la codificación transcurrió sin contratiempos. El proceso se desarrolló de manera fluida, y no se presentaron problemas significativos (solo una falta de puntuación, pero se corrigió en el instante de encontrar el fallo).



INSTITUTO TECNOLÓGICO SUPERIOR DE ALAMO TEMAPACHE

ETAPA DE COMPILACION:

En esta etapa de compilación, el código se procesó sin inconvenientes. El compilador no generó errores ni advertencias significativas. Debido a que NetBeans cuenta con un proceso de verificación de sintaxis me ayudo a no tener ningún error, además me ayudo a tener coherencia. Esto me permitió avanzar hacia la siguiente etapa con confianza en la integridad del programa.

ETAPA DE PRUEBAS:

En la etapa de pruebas, no fue necesario llevar a cabo una serie de evaluaciones exhaustivas en nuestro programa debido a que ya poseo más conocimientos en el desarrollo. Observe que el programa se comportaba de acuerdo con nuestras expectativas y cumple con los requisitos establecidos previamente. No se identificaron fallos lo que indica que el software estaba listo para su implementación y uso, lo único que se modificó fue que se agregó comentarios dentro del código sobre su funcionamiento.

EVALUACION DE RESULTADOS:

METAS ESTABLECIDAS:

En la etapa de planeación se establecieron metas principales en base a los requerimientos, las principales son:

1. El tiempo estimado para la elaboración de este proyecto fue de 4hrs (fue en menor tiempo).
2. El programa calcula los tamaños relativos de los datos de la tabla 1 y 2.
3. Obtener y mostrar los resultados de la tabla 3.

METAS NO CUMPLIDAS:

Me alegra decir que logramos todas las metas como se había planeado. Todo está funcionando como debería. Esta experiencia nos servirá de lección para futuros proyectos, ya que nos ayudará a ajustar mejor nuestras estimaciones de tiempo y mejorar la gestión del tiempo en proyectos similares.

METAS CUMPLIDAS:

- El código resultante funciona de acuerdo con los requisitos y expectativas establecidos.
- No se identificaron errores críticos ni problemas de rendimiento.
- El cálculo de rangos de tamaño relativo para objetos muy pequeños, rangos pequeños, medianos, grandes y muy grandes utilizando la desviación estándar se realizan con precisión.
- El programa utiliza los datos proporcionados en tablas 1 y 2.
- Los valores esperados son similares a la tabla 3.

En general, el proyecto se completó con éxito y todas las metas y objetivos se alcanzaron de manera satisfactoria.



Marcamos la etapa de finalización:

PSP Time Recording Log

Carnegie Mellon Software Engineering Institute

Personal Software Process

Student: Gonzalo Martinez Silverio

Program: Program 4

Instructor: TANIA TURRUBIATES LOPEZ

Start Date: 03-nov.-23

End Date:

Language: Java

ID	PID	Phase	Start	Int.	Stop Date and Time	Delta	Comments
41	424	PLAN	03/11/2023 06:59:56 a. r	0	03/11/2023 07:20:16 a. r	20.3	Planeacion
43	424	DLD	03/11/2023 07:20:17 a. r	0	03/11/2023 07:50:21 a. r	30.1	Diseño de diagrama UML
44	424	CODE	03/11/2023 07:50:22 a. r	0	03/11/2023 08:27:01 a. r	36.7	Codificación
45	424	COMPILE	03/11/2023 08:27:02 a. r	0	03/11/2023 08:27:15 a. r	0.2	Compilación
46	424	UT	03/11/2023 10:51:59 a. r	0	03/11/2023 11:04:01 a. r	12.0	Testeo
47	424	PM	03/11/2023 11:05:02 a. r	0	03/11/2023 11:35:06 a. r	30.1	POSMORTEM
						0.0	
Total						129.4	


Registro: 1 de 1

Sin filtro

Buscar


RESULTADOS:

TABLA DE REGISTRO DE DEFECTOS:


PSP Defect Recording Log

Personal Software Process

Personal Software Process


Carnegie Mellon
Software Engineering Institute

PSP Defect Recording Log

Student: Gonzalo Martinez Silverio
Program: Program 4
Instructor: TANIA TURRUBIATES LOPEZ

ID: 424

Start Date: 03-nov.-23
End Date:
Language: Java

ID	PID	Date	Type	Phase Injected	Phase Removed	Fix Time	FixDefect
16	424	03/11/2023	10-Documentation	CODE	UT	1	3
Defect Description		No se documento el codigo en la codificacion y se corrigio en la fase de testeo					
17	424	03/11/2023	20-Syntax	CODE	CODE	2	1
Defect Description		Hubo un error de puntuacion pero gracias a que netbeas muestra los errores se corrigio al momento de observar la falla.					



INSTITUTO TECNOLÓGICO SUPERIOR DE ALAMO TEMAPACHE

PROPUESTA DE MEJORA DE PROCESOS (PIP)

Carnegie Mellon Software Engineering Institute		Personal Software Process	
PSP Process Improvement Proposal			
Student	Gonzalo Martinez Silverio	Start Date	03-nov.-23
Program	Program 4	End Date	
Instructor	TANIA TURRUBIATES LOPEZ	Language	Java
UID	#¿Nor	Date	03-nov.-23
Problem Description Briefly describe the problems you encountered.			
Por error no agregue puntuacion al final de una linea para imprimir el resultado, esto generaba un error de sintaxis.			
Dentro de la etapa de codificacion no fui agregando la documentacion para que otras personas (desarrolladores) puedan comprenderlo.			
Proposal Description Briefly describe the process improvements that you propose.			
Se debe de poner mas atencion a aquellos pequeños detalles que se deben de tener en consideracion para tener unas mejores practicas.			
Other Notes and Comments Note any other comments or observations that describe your experiences or improvement ideas.			
Me comprometeré a tener mas atencion en la etapa de codificacion para evitar lo sucedió y optimizar los tiempos.			



INSTITUTO TECNOLÓGICO SUPERIOR DE ALAMO TEMAPACHE

PLANTILLA DE ESTIMACIÓN DE TAMAÑO

PSP Size Estimating Template

Carnegie Mellon Software Engineering Institute

Personal Software ProcessSM

PSP Size Estimating Template

ID 424

Student: Gonzalo Martinez Silverio Start Date: 03-nov.-23

Program: Program 4 End Date:

Instructor: TANIA TURRUBIATES LOPEZ Language: Java

Parts: Base

ID	Name	Plan				Actual			
		Base	Del.	Mod.	Add	Base	Del.	Mod.	Add
4		108	0	0	0	108	58	50	3
vo)		0	0	0	0	0	0	0	0
Base TOTAL		108	0	0	0	108	58	50	3

Parts: Added

ID	Name	Part Type	Plan			Actual	
			Items	Rel. Sz.	Size *	Items	Size *
25	Datos Tabla 1 y 2	Data	20	S	26.0	24	26
26	Metodo calculo tamaño relativo	Calc	15	VS	19.0	11	16
27	Presentacion (RESULTADOS)	IO	12	VS	13.0	10	10
vo)			0		0.0	0	0
Added Parts TOTAL			58			52	

Parts: Reused


ID	Name	Plan	Actual
3	Program 2	0	1
vo)		0	0
Reuse Object TOTAL		0	1


PROBE Calculation Worksheet

		Size	Time
Added Size (A):	A=BA+PA	58	
Estimated A&M (E):	E=BA+PA+M	58	
PROBE method used: (A,B,C,D)		D	D
Correlation (R^2):			
Regression Parameter (B0):	Size and Time	0	0
Regression Parameter (B1):	Size and Time	1	1
Projected A&M (P):	P=B0+B1*E	58	
Estimated Total Size (T):	T=P+B-D-M+R	166	
Estimated Total New Reusable (NR):	sum of * items	0	
Estimated Total Development Time:	Time=B0+B1*E		240
Prediction Range:	Range	0	0
Upper Prediction Interval:	UPI=P+Range	0	0
Lower Prediction Interval:	LPI=P-Range	0	0
Prediction Interval Percent:		70%	70%



INSTITUTO TECNOLÓGICO SUPERIOR DE ALAMO TEMAPACHE

PROBE Historical Data - Methods B and C					Plan		Actual		
ID	Name	End Date	Time	A&M	E	Time	A&M	Outlier	
PROBE Method Parameters - Methods B and C									
	R ² Size	Beta0 Size	Beta1 Size	R ² Time	Beta0 Time	Beta1 Time			
Method C	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>			
Method B	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>			

PROBE Historical Data - Method A					Plan		Actual		
ID	Name	End Date	Time	A&M	E	Time	A&M	Outlier	
PROBE Method Parameters - Method A									
	R ² Size	Beta0 Size	Beta1 Size	R ² Time	Beta0 Time	Beta1 Time			
Method A	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>			

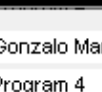
Registro: 1 de 1 Sin filtro Buscar

RESUMEN DEL PLAN DEL PROYECTO



Registro: 1 de 1 Filtrado Buscar





Carnegie Mellon

Software Engineering Institute

Personal Software Process

PSP Test Report

Student

Gonzalo Martinez Silverio

Start Date

03-nov.-23

Program

Program 4

End Date

Instructor

TANIA TURRUBIATES LOPEZ

Language

Java

Test Number

01

Test Name

PROGRAMA 4

Objective

El objetivo de esta prueba es verificar la funcionalidad del programa Java que calcula el tamaño relativo de diferentes conjuntos de datos.

Description

El programa utiliza dos HashMaps, datosLocMetodo y datosPagsCapitulo, para almacenar los datos. El método calcularTamañoRelativo se utiliza para calcular el tamaño relativo de los datos. Primero calcula el logaritmo natural de cada punto de datos, luego calcula la media y la desviación estándar de estos valores. Luego utiliza estas estadísticas para clasificar los puntos de datos en cinco categorías: "VS", "S", "M", "L" y "VL". El método devuelve un nuevo HashMap con estas categorías como claves y los tamaños relativos como valores.

Conditions

La prueba se realizó en un sistema que ejecuta Windows 11 con Java instalado. El código se compiló y se ejecutó utilizando el IDE NetBeans y JDK.

Expected Results

El resultado esperado es que el programa calculará correctamente los tamaños relativos de los puntos de datos en los HashMaps datosLocMetodo y datosPagsCapitulo, e imprimirá estos resultados similares a la tabla 3.

Actual Results

El código funciona como se esperaba, los resultados fueron similares a la tabla 3.



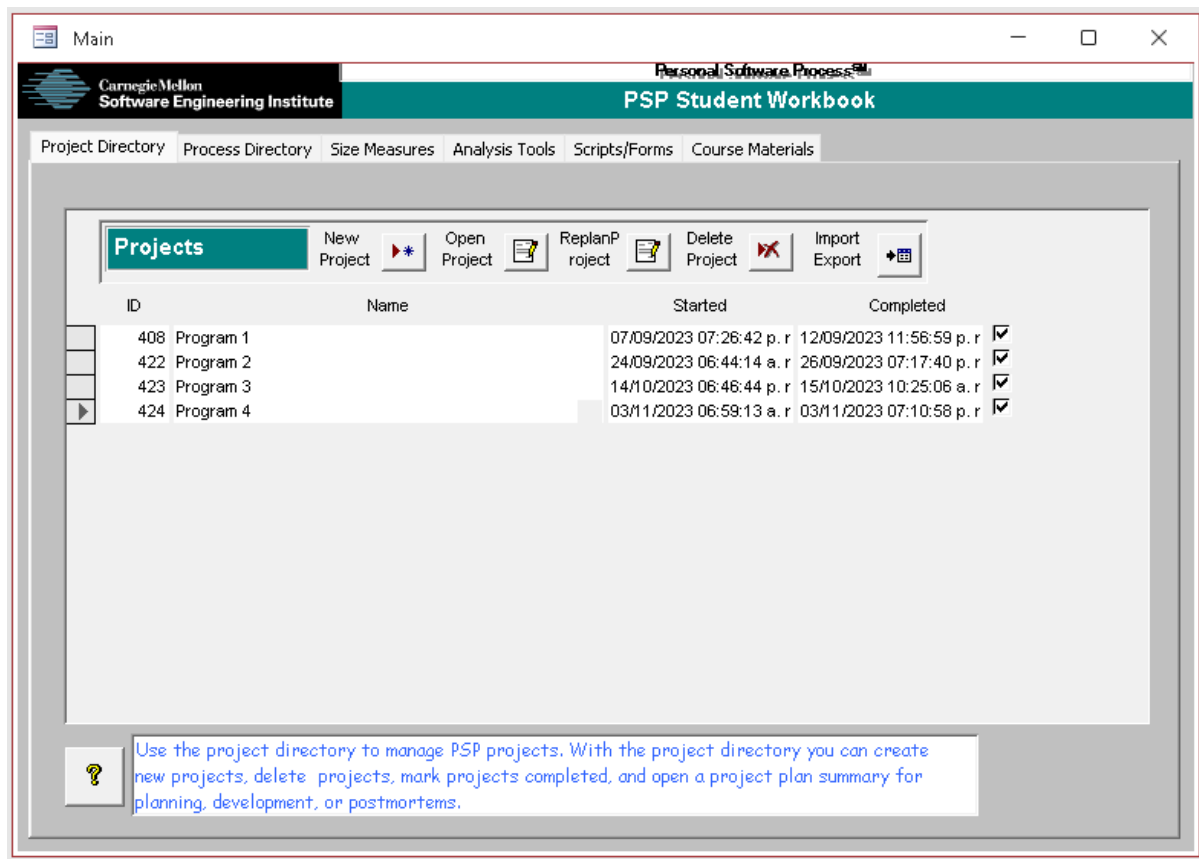
PLANES DE TAREAS Y HORARIOS

Registro: 1 de 1 Sin filtro Buscar



INSTITUTO TECNOLÓGICO SUPERIOR DE ALAMO TEMAPACHE

MARCAMOS LA FINALIZACIÓN DEL PROYECTO.



CONCLUSION:

Las etapas de planeación, diseño, codificación, compilación y pruebas se llevaron a cabo de manera eficiente y exitosa en el proyecto. A pesar de enfrentar un retraso en el tiempo estimado para la elaboración, se lograron las metas principales del proyecto. La calidad y la funcionalidad del software se mantuvieron, lo que demuestra una planificación efectiva y una ejecución exitosa. Además, se identificaron áreas de mejora para futuros proyectos, como la gestión del tiempo y las estimaciones de desarrollo.

En este contexto de PSP1, el código ha demostrado ser una herramienta efectiva para realizar cálculos de regresión lineal. Además, el proceso PSP1 ha permitido al programador llevar un seguimiento disciplinado de su trabajo, mejorar su productividad y aplicar buenas prácticas en el desarrollo de software. El éxito de esta prueba valida la utilidad del proceso PSP1 como un enfoque efectivo para la mejora personal del programador y la entrega de código de alta calidad. El código se encuentra en conformidad con estos estándares y está listo para ser utilizado en proyectos futuros. Con lo anterior puedo decir que es importante continuar aplicando los principios del PSP para mantener y mejorar la calidad de trabajo a lo largo de la carrera profesional.



INSTITUTO TECNOLÓGICO SUPERIOR DE ALAMO TEMAPACHE

OBSERVACIONES PROPIAS PARA MEJORAR:

1. Mejorar la concentración.
2. Mejorar los tiempos en cada etapa para ser más eficiente.
3. Mejorar al momento de codificar

BIBLIOGRAFIA:

Watts S. Humphrey (2005). PSP A Self-Improvement Process for Software Engineers. Addison-Wesley Professional.

PSP Academic Material (2016). Acceso el 20 de Agosto de 2016 desde Team Software Process. Software Engineering Institute, Carnegie Mellon University. Sitio Web: <http://www.sei.cmu.edu/tsp/tools/academic/index.cfm>