Universidad de las Ciencias Informáticas

Facultad 5

Título: *"Paquete de métricas incorporado a una herramienta para los proyectos del Polo de Hardware y Automática".*

Trabajo de Diploma para optar por el título de
Ingeniero en ciencias Informáticas

Autor(es): Adanlay Rivero Montero
Yoderky Vivas Gándara
Tutor: Ludisley La Torre Hernández

Ciudad de La Habana
Junio, 2008
"Año del 50 aniversario del Triunfo de la Revolución"

Datos de Contacto

Ing. Ludisley La Torre Hernández.
Graduada de Ingeniera en Ciencias Informáticas en el año 2007 y profesor instructor con 2 años de experiencia docente en la Universidad de las Ciencias Informáticas (UCI).

Agradecimientos

Compartidos:

A la Revolución, por brindarnos la oportunidad de ser mejores y por confiar su futuro en nosotros.

A Fidel, nuestro Comandante en Jefe, máximo guía y forjador de esta gran casa de estudios que es la (UCI), en la que hemos hecho realidad nuestros sueños como estudiantes, y a la que hemos de llevar siempre en nuestros corazones, por aportarnos durante estos 5 años un cúmulo imperecedero e invaluable de conocimientos y experiencias.

A nuestra tutora Ludisley por su empeño y dedicación.

Muchas Gracias.

Agradecimientos

Adanlay:

A mí querida madre por ser la mejor madre del mundo, por estar ahí en todo momento, por apoyarme y confiar en mí, por sus consejos paciencia y amor.

A mi padre por apoyarme y confiar en mí, por ser más que un padre un amigo por sus consejos y por preocuparse tanto por mí.

A mi tío Marcos, por ayudarme en todo lo que necesité por sus palabras de aliento y por confiar en mí, estoy eternamente agradecido.

A toda mi gran familia, por apoyarme por quererme, disculpen si no los menciono a todos, a mi hermana mis primos mis abuelos sobre todo, a mis tías, sepan todos que los quiero y les agradezco todo el amor y la educación que me dieron.

A Yoderky, por ser un excelente compañero de tesis y un gran amigo, por soportarme en todo momento, por no molestarse al despertarlo todas las mañanas para trabajar en la tesis, por estar siempre cuando lo necesité.

A mi novia, por apoyarme, por entender lo importante que es este trabajo y el tiempo que requiere, por quererme y ayudarme en todo momento.

A mis amigos (Juan Luis, Roberto, Enerys, Ana Isis, Ernesto), por apoyarnos en todo momento por compartir los conocimientos en aras de poder hacer un trabajo con calidad, al resto del grupo tanto el viejo como el nuevo, por hacer de mi una mejor persona.

A todos

Muchas Gracias.

Agradecimientos

Yoderky:

Quiero agradecerle a mi mamá, mi tío panchito, mi papá Sergio, mi abuela Eloisa gracias por estar ahí siempre cuando los necesité, por sus consejos, por su guía, su comprensión, Ustedes, son los pilares de los que me sostengo... Son los dioses de mi religión, los cuales venero. A toda mi familia, Yoxy, Yaxy, Mary, Nosley, Omarcito, Nory, Zahelys, Janier, Danna, Arley, en fin a toda que es mucha por mencionar pero que siempre los tengo presente, a mis "hermanitos" Pedro, Gesley y Manuel a todos gracias por si cariño. A todos mis amigos los que han logrado que la UCI sea para mí una experiencia inolvidable, especialmente a Robe, Ene, venty, Bety, Ana Ernest, gracias por compartir este espacio-tiempo conmigo. A Adán mi compañero de tesis, creo que si tuviera volviera a seleccionar un compañero de tesis no encontraría en 1 millón nadie que te remplace, ni como compañero ni como amigo, Gracias a todos…

Dedicatoria

Adanlay:

A mi mamá y mi papá por ser el gran motivo de mi inspiración y esfuerzo,

Por apoyarme en todo momento.

A mi hermana y a toda mi familia, este triunfo también es de ustedes,

Mil gracias por confiar en mí.

Yoderky:

A mi mamá, mi tío, mi abuela y Sergio…

Les debo lo que soy hoy y lo que seré mañana. Gracias

Resumen

La aplicación de métricas para el proyecto es actualmente un proceso fundamental para todas las empresas productoras de software tanto a nivel mundial como en nuestro país. El objetivo fundamental de la aplicación de estas métricas es llevar el control y seguimiento de los resultados de las mediciones realizadas durante el tiempo el desarrollo del proyecto.

En la Universidad de las Ciencias Informáticas (UCI), existen numerosos proyectos productivos, pero no en todos emplean o utilizan métricas, además en no todos se guardan datos históricos de los resultados obtenidos con las mediciones realizadas al proyecto de desarrollo de software en caso de que se realicen estas mediciones. Esto no impide que se produzca con calidad pero si puede provocar retardo en el término del producto en si, pues al no tener datos históricos en que basarse, entonces se debe hacer todo desde cero y no se podrán hacer correctas estimaciones así como tampoco tendrían alguna experiencia anterior en que apoyarse en el momento de tomar una decisión.

El tema del trabajo surge como respuesta a los planteamientos anteriores para tratar de dar solución a dichos problemas, es por ello que el objetivo del mismo es proponer un paquete de métricas incorporado a una extensión para la herramienta de gestión de proyecto Trac que permitan el control y seguimiento de los proyectos de Software en el Polo de Hardware y Automática con el fin de facilitar el registro de los datos históricos de las mediciones que se realizan en los proyectos y validar la propuesta del paquete de métricas.

PALABRAS CLAVE: Métricas para el proyecto, data histórica, control, seguimiento, software, extensión, Trac.

Tabla de contenido

Capítulo 1 Fundamentación Teórica *14*
Introducción *14*
1.1 Medición *14*
1.2 Medida *15*
1.3 Importancia de realizar una medición *15*
1.4. Proceso de medición *16*
1.4.1 Actividades y tareas que se deben tener en cuenta en el momento de realizar una medición *17*
1.5 Conjunto de mediciones básicas para proyectos de software *18*
1. 6 Métrica *18*
1.7 Métricas de Software *19*
1.8 Características de las métricas del software *20*
1.9 Establecimiento de una línea base *21*
1.10 Métricas en el Proyecto *22*
1.11 Características de las Métricas para el proyecto *22*
1.12 Indicador *23*
1.13 Aplicación de métricas para proyectos de software en el mundo *23*
1.13.1 Métricas en Cuba *25*
1.13.2 Métricas en la Universidad de las Ciencias Informáticas *26*
1.14 Tendencias y Tecnologías actuales *28*
1.14.1 Sistema Operativo GNU/Linux *28*
1.14.2 Distribución de Linux: Debian GNU/Linux *29*
1.14.3 Herramientas para la gestión de proyectos *30*
1.14.4 Herramienta Trac *31*
1.14.5 Extensiones para el Trac *32*
1.14.6 Lenguaje de Programación *33*
1.14.7 Lenguaje de Modelado NET BEANS *34*
1.14.8 Metodología de Desarrollo de Soft: Open-Up *35*
1.14.9 Eclipse *35*
1.14.10 Matplotlib *36*
1.14.11 Sqlite *36*
Capítulo 2: Diseño de la Solución *38*
Introducción *38*
2.1 Situación actual de los proyectos del Polo de Hardware y Automática *38*
2.2 Propuesta del paquete de Métricas *38*
2.4 Arquitectura *44*
2.5 - Diseño del Extensión *45*
2.5.1 - Funcionalidades definidas *45*
2.5.2 Requisitos no Funcionales *45*
2.6 Casos de Uso Definidos *46*

2.6.1 Identificar el actor 46
2.6.2 Diagrama de CU 46
2.6.3 Descripción de los CU 47
2.6.4 Diagramas de Secuencia 53
2.7 Modelo de Diseño e Implementación 59
2.7.1 Patrones de Diseño 59
2.7.2 Diagrama de clases 60
2.7.3 Diagrama de Despliegue 69
Capítulo #3 Pruebas y seguimiento de la solución planteada 71
Introducción 71
3.1 Diseño de clases de pruebas 71
3.2 Manual de Usuario basado en los prototipos funcionales. 83
3.3 Resultados presentes y visión futura 85
3.4 Pruebas 85
Conclusiones 87
Recomendaciones 88
Referencias Bibliográficas 89
Bibliografía Consultada 93
Anexos 96
Anexo # 1 96
Anexo # 2 98
Anexo # 3 103
Glosario 105

Introducción

La industria del software se desarrolla a un ritmo acelerado, aunque la producción no es lo suficientemente alta, esto se debe en la mayoría de los casos, a la no aplicación de técnicas de Ingeniería y Gestión del Software. Estas técnicas son de vital importancia pues les indica seguridad a los usuarios y clientes de estos productos, proporcionándoles a los mismos sus pedidos a tiempo y con buena calidad. Nuestro país no se encuentra exento del incremento del uso de las Tecnologías de la Informática y las Comunicaciones, hecho que lo demuestra es la creciente formación de profesionales en la esfera de la informática. La Universidad de las Ciencias Informáticas, creada al calor de la Batalla de Ideas, es una universidad innovadora de excelencia científica, académica y productiva que forma profesionales integrales altamente comprometidos con la Revolución, cuya misión es producir software y servicios informáticos a partir de la vinculación estudio-trabajo. Para ello integra los procesos de formación, investigación y producción en torno a una temática para convertirla en una rama productiva.
La UCI promueve el desarrollo de productos y servicios informáticos en aquellas ramas donde Cuba tiene un reconocido prestigio en el mundo, a través del concurso de los mejores especialistas del país para lograr una solución de calidad e impacto tanto en la sociedad cubana actual como a nivel internacional.

En el curso 2006-2007 se propuso implantar en la universidad el modelo de Polo Productivo, que tuvo su antecedente en la Infraestructura Productiva (IP) que ya se había creado. La idea de los polos es crear un espacio natural para ejecutar proyectos temáticos. Dentro de estos polos se encuentra el “Polo de Hardware y Automática” a desarrollar por la Facultad 5.

En la universidad, de manera general, no se realiza un registro de los datos históricos con los resultados de las mediciones que se realizan a los proyectos productivos. En la facultad 5 y especialmente en el Polo de Hardware y Automática se han realizado esfuerzos para registrar las mediciones que se realizan en cada uno de los proyectos, pero aún queda mucho por hacer en ese sentido. Esto podría traer como consecuencia varios problemas y efectos negativos, como por ejemplo, no se podría hacer una correcta y acertada estimación de tiempo, esfuerzo, costo, así como tampoco se podría tener como referencia dichas mediciones para futuras

tomas de decisiones, también podría retrasar el desarrollo de los proyectos y la calidad de los mismos se vería comprometida.

Para dar respuesta a la situación problemática expuesta anteriormente se considera el siguiente problema científico ¿Cómo gestionar los datos históricos de las mediciones realizadas en los proyectos del polo de Hardware y Automática?

Según el problema científico expuesto anteriormente se plantea como **objeto de estudio**: El proyecto de desarrollo de software y dentro de este el **campo de acción** Las mediciones que se realizan a los proyectos de software en el Polo de Hardware y Automática.

Luego de este análisis se formula como **objetivo general**: proponer un paquete de métricas incorporado a una extensión que permitan el control y seguimiento de los proyectos de Software en el Polo de Hardware y Automática.

Del objetivo general de la investigación se derivaron las siguientes tareas investigativas.

Tareas Investigativas:

- Realización de un estudio de las métricas más utilizadas en los proyectos de software para adquirir conocimiento sobre el tema.
- Definición de las variables a evaluar para seleccionar las métricas adecuadas.
- Elaboración de un paquete de métricas para incorporarlas al Trac.
- Implementación de una extensión para comenzar a registrar los datos históricos de las mediciones que se realicen.

De acuerdo con el problema científico **la idea a defender** es la siguiente: La propuesta de un paquete de Métricas incorporado a una extensión para la herramienta de gestión de proyecto Trac, facilitará tener un mayor control y seguimiento de los proyectos de software en el Polo de Hardware y Automática.

Para el cumplimiento de estos objetivos se llevan a cabo varios métodos y técnicas en la búsqueda y procesamiento de la información como son:

A nivel teórico:

Métodos de análisis-síntesis: Para el estudio de las concepciones y los conceptos empleados dentro del uso de las métricas en el proyecto, analizando todos los documentos para la extracción de los elementos más importantes sobre el tema en cuestión.

Análisis histórico–lógico: Para conocer, con mayor profundidad los antecedentes y las tendencias actuales referidas al uso de las métricas en el proyecto de software sabiendo así la trayectoria histórica que tiene a través de su origen.

A nivel empírico:

Encuesta: Para conocer la realidad en cuanto a la aplicación de estas métricas en la universidad, y sobre el registro de los datos históricos generados a partir de estas mediciones.

Con el paquete de métricas que se propondrá en la presente investigación y la implementación de un extensión para incorporar dichas métricas al Trac de cada proyecto, posibilitará guardar los datos históricos de las mediciones que se realicen, aportando información relevante para la toma de decisiones. Esto será de gran beneficio pues estos datos históricos permitirán realizar estimaciones de tiempo, esfuerzo, contribuyendo a lograr una correcta gestión en los proyectos del polo.

El presente trabajo consta de una Introducción, el Desarrollo dividido en tres capítulos, las conclusiones generales, recomendaciones, referencias bibliográficas, bibliografía consultada, los Anexos y el glosario de términos.

Capítulo 1 Fundamentación Teórica En este capítulo se mostrará los conceptos de medida, medición, métrica, métricas de software, métricas para el proyecto, indicadores, sistema de

métricas y el estado del arte de las empresas de software en el mundo, en el país y en la Universidad. Se definen las tecnologías utilizadas para implementar la solución.

Capítulo 2 Diseño de la Solución En este capítulo se expondrán las métricas que fueron seleccionadas para comenzar a registrar sus resultados en los proyectos del Polo, se enuncia una breve descripción de las mismas. Además se listan las funcionalidades definidas para la extensión y a partir de ellas se realizará el diseño e implementación del mismo.

Capítulo 3 Pruebas y Seguimiento de la Solución Planteada En este capítulo diseñarán y realizarán pruebas a la extensión obtenida con el objetivo de comprobar que cumple con todos los objetivos planteados, además se realizará un manual de usuario con el objetivo de explicar el funcionamiento de la extensión, facilitar su utilización así como contribuir a realizar un correcto uso de la misma.

Capítulo 1 Fundamentación Teórica

Introducción

En la actualidad el término métricas de software es ampliamente difundido y utilizado. En el polo de Hardware y Automática de la facultad 5 se están haciendo esfuerzos para lograr registrar los datos históricos de las mediciones que se realizan en cada uno de los proyectos, para esto es necesario utilizar un paquete de métricas que posibiliten lograr un mayor control y seguimiento de los proyectos de software. Durante la Gestión del Proyecto se realiza una planificación de los recursos que se necesitan para el desarrollo del mismo, por lo que se deben obtener estimaciones del costo, tiempo, presupuesto y el esfuerzo humano requeridos. Una de las formas de lograr una correcta Gestión de Proyecto es sin dudas por medio de las mediciones de software y de las métricas que se utilicen en el proyecto para recolectar datos históricos que permitan realizar estimaciones más exactas.

Durante más de cinco décadas, el área de los conocimientos que explora la medición del software ha generado definiciones que han traído confusión en la utilización de los términos métricas de software y mediciones de software, sin embargo, en muchas literaturas los términos métrica, medida o medición son usados como vocablos análogos. (Zuse, 1998).

1.1 Medición

Medición es la terminología relacionada con el acto de medir software. Una medición es un conjunto de operaciones cuyo objetivo es determinar el valor de un atributo dado de una entidad, utilizando una forma de medir. Los resultados de la medición son las medidas. (PFLEEGER, 2001).

La medición del software se refiere a derivar a un valor numérico para algún atributo de un producto de software o un proceso de software. Comparando estos valores entre ellos y con los estándares aplicados en la organización, es posible sacar conclusiones de la calidad del software o de los procesos del software. (Sommerville, 2002).

Algunos aspectos importantes sobre medición.

- Debe llevar a la acción, y no ser empleada simplemente para acumular datos.
- Deben tener un propósito claramente definido. La medición del software apoya la gestión y la mejora de los procesos y productos de software.
- Es una herramienta primaria para las actividades del ciclo de vida de la gestión de sistemas y software, la evaluación de la viabilidad de los planes de proyecto, y la supervisión de la correspondencia de las actividades del proyecto con estos planes.
- Es una disciplina importante en la evaluación de la calidad de los productos de software y de la capacidad de los procesos del software organizacionales.

1.2 Medida

Un Medida proporciona una indicación cuantitativa de extensión, cantidad, dimensiones, capacidad y tamaño de algunos atributos de un proceso dado. Es el resultado de una medición.

Según la Norma ISO Medida es valor numérico para un atributo cuya magnitud se desea valorar en función de una escala concreta. (PFLEEGER, 2001).

Cuando puedas medir lo que estás diciendo y expresarlo en números, sabrás algo acerca de eso; pero cuando no puedes medirlo, cuando no puedes expresarlo en números, tus conocimientos serán escasos y no satisfactorios.

Lord Kelvin

1.3 Importancia de realizar una medición

Una de las razones principales del incremento masivo en el interés por la medición de software ha sido la percepción de que las métricas son necesarias para la mejora de la calidad del proceso.

Hay cuatro razones para medir los procesos del software, los productos y los recursos, (PRESSMAN, 1998):

- Caracterizar: Para comprender mejor los procesos, los productos, los recursos y los entornos y para establecer las líneas base para las comparaciones con evaluaciones futuras.
- Evaluar: Para determinar el estado con respecto al diseño. Las medidas permiten conocer cuándo los proyectos y procesos están perdiendo la pista, de modo que puedan ponerse bajo control. Además para valorar si se cumplen o no los objetivos de calidad trazados y para evaluar el impacto de la tecnología y las mejoras en los productos y procesos.
- Predecir: Para poder planificar. Los valores que se observan para algunos atributos pueden ser utilizados para predecir otros, lo que contribuye a establecer objetivos alcanzables para el coste, planificación, y calidad, de manera que se puedan aplicar los recursos apropiados, además permite analizar los riesgos y realizar intercambios diseño coste.
- Mejorar: Se mide para mejorar cuando se recoge la información cuantitativa que ayuda a identificar obstáculos, problemas de raíz, ineficiencias y otras oportunidades para mejorar la calidad del producto y el rendimiento del proceso.

1.4. Proceso de medición

El proceso de medición se describe a través de un modelo que define las actividades y tareas que se necesitan para especificar adecuadamente qué información de la medición se requiere, cómo los resultados de las mediciones y del análisis serán aplicados, y cómo determinar si los resultados del análisis son válidos. El proceso de medición de software debe ser flexible, ajustable y adaptable a las necesidades de diferentes usuarios.

El **Proceso de Medición** especificado en [ISO/IEC 15939] comprende las siguientes actividades:

- Establecer y mantener el compromiso con respecto a la medición.
- Planificar el proceso de medición.
- Realizar el proceso de medición.
- Evaluar la medición.

1.4.1 Actividades y tareas que se deben tener en cuenta en el momento de realizar una medición

1.4.2.1 Establecer y mantener el compromiso con respecto a la medición

- Aceptar los requisitos para la medición
- Asignar los recursos

1.4.2.2 Planificar el proceso de medición

- Caracterizar la unidad organizacional.
- Identificar las necesidades de información.
- Seleccionar medidas.
- Definir los procedimientos para la recopilación, el análisis, y el reporte de los datos.
- Definir los criterios para evaluar los productos de información y el proceso de medición.
- Revisar, aprobar, y suministrar los recursos a las tareas de medición.
- Adquirir y desplegar las tecnologías de apoyo.

1.4.2.3 Realizar el proceso de medición

- Integrar los procedimientos.
- Recopilar los datos.
- Analizar los datos y desarrollar los productos de información.
- Comunicar los resultados.

1.4.2.4 Evaluar la medición

- Evaluar los productos de información y el proceso de medición.
- Identificar las mejoras potenciales.

1.5 Conjunto de mediciones básicas para proyectos de software

Mediciones de Desempeño: Si no hay métricas de desempeño no habrá seguimiento efectivo del estado del proyecto, no se conocería el estado actual de tiempos y esfuerzos y no se pudiera comparar planificado con real.

Mediciones de Desarrollo: Si no hay métricas de desarrollo no habrá evaluación del proceso de desarrollo ni de sus resultados, no se puede cuantificar cuan bien esta resultando un nuevo proceso y no se puede comparar el desempeño actual con una línea base.

Mediciones del Producto: Si no hay métricas del producto entonces no habrán productos estables, no existirán criterios consistentes para decidir cuando un producto es estable y no se conocerían las tasas de defectos y de soluciones.

1. 6 Métrica

Las métricas son un medio para entender, monitorizar, controlar, predecir y probar el desarrollo software (Briand, 1996).

Una métrica es una medida cuantitativa del grado en que un sistema, componente o proceso posee un atributo dado. (IEEE "Estandard Glosary of Software Engering Terms ")

Es un criterio para determinar la distancia entre dos entidades. (Zuse, 1998)

Toda métrica es una medida, pero una medida no tiene por qué ser una métrica. El término adecuado es "métrica" a pesar de que ambos se usan indistintamente. (Zuse, 1998)

Una métrica es un algoritmo o cálculo realizado para combinar dos o más medidas base. La escala y la unidad de las medidas derivadas dependerán de las escalas y unidades de las medidas base que la componen, así como la forma en que se combinan por la función.

1.7 Métricas de Software

Una métrica de software es una correspondencia entre uno o más atributos del entorno de desarrollo del software, y cualquier otro atributo. (Fenton, 1997).

Las métricas de software proporcionan fórmulas matemáticas para medir diferentes atributos del software. Entre los más frecuentes se encuentran el tamaño, la complejidad. (Briand, 1997). Se definen las métricas de software como "La aplicación continua de mediciones basadas en técnicas para el proceso de desarrollo del software y sus productos, para suministrar información relevante a tiempo, así el administrador junto con el empleo de estás técnicas mejorará el proceso y sus productos" (GONZÁLEZ, 2001).(Ver Figura 1)

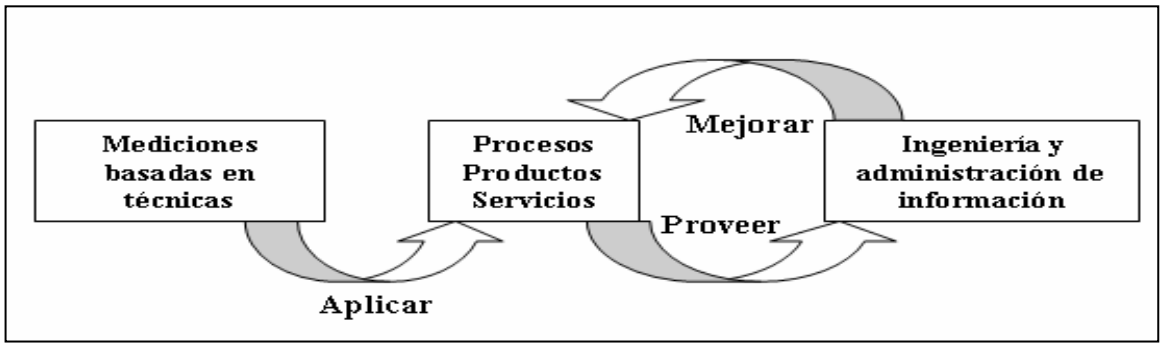


Figura 1. Aplicación de Métricas.

Se han propuesto cientos de métricas para el software, pero no todas proporcionan suficiente soporte práctico para su desarrollo. Algunas demandan mediciones que son demasiado complejas, otras son tan esotéricas que pocos profesionales tienen la esperanza de

entenderlas, y otras violan las nociones básicas intuitivas de lo que realmente es el software de alta calidad. Es por eso que se han definido una serie de características o propiedades que deben acompañar a las métricas efectivas de software, ver acápite 1.8.

1.8 Características de las métricas del software

Simples y fáciles de calcular: Deberían ser relativamente fácil aprender a obtener la métrica y su cálculo no debería demandar un esfuerzo o cantidad de tiempo inusuales.

Empírica e intuitivamente persuasivas: Satisfacer las nociones intuitivas del ingeniero sobre el atributo del producto en cuestión.

Consistentes y objetivas: Deberían siempre producir resultados sin ambigüedad. Un tercer equipo debería ser capaz de obtener el mismo valor de métrica usando la misma información del software.

Consistentes en el empleo de unidades y tamaños: El cálculo matemático de la métrica debería emplear medidas que eviten extrañas combinaciones de unidades.

Independientes del lenguaje de programación: Deben basarse en el modelo de análisis, diseño o en la estructura del programa.

Eficaces en el mecanismo para la realimentación de la calidad: Proporcionar al desarrollador de software información que le lleve a un producto final de mayor calidad.

Ventajas de las métricas de software

Se considera que las métricas de software ayudan a los desarrolladores a valorar el trabajo desarrollado, proveen la información necesaria para la toma de decisiones técnicas, proporcionan datos objetivos que pueden ser usados en la planificación de futuros proyectos,

ayudan a la evaluación de los modelos de análisis y de diseño, y a la formulación de casos de prueba, valoran la productividad de los desarrolladores, ayudan a evaluar la calidad de los productos o sistemas que se construyen así como también a entender que ocurre durante el desarrollo y el mantenimiento.

A continuación se muestra el proceso de recopilación de métricas del software.

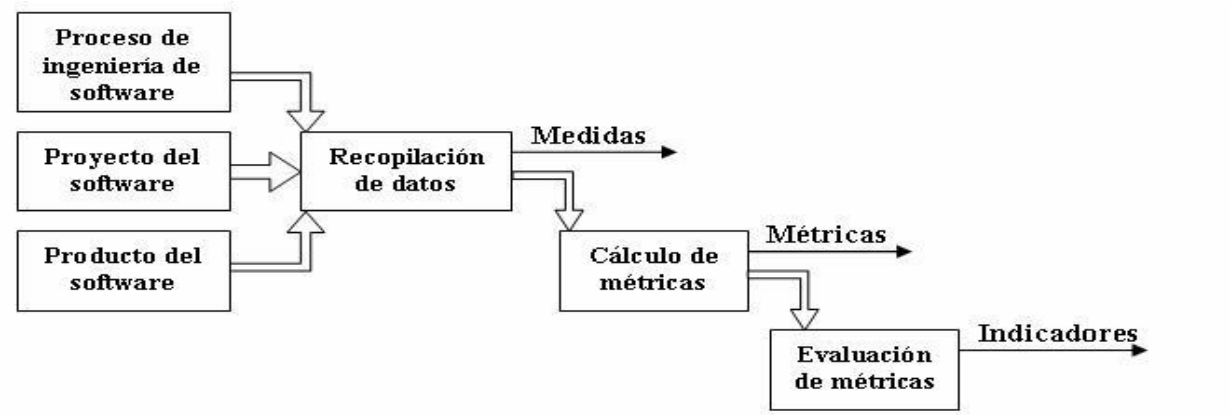


Figura 2 Proceso de recopilación de métricas del Software.

1.9 Establecimiento de una línea base

Un punto de partida para realizar estimaciones es establecer una línea base de métricas que permita a una organización sintonizar su proceso de ingeniería del software para eliminar las causas de los defectos que tienen el mayor impacto en el desarrollo del software, es fundamental que una línea base contenga datos recopilados de proyectos desarrollados anteriormente lo que requiere una investigación histórica de los mismos, la línea base no es más que la recopilación de medidas, métricas e indicadores que guíen el proyecto de desarrollo del software. Las métricas permiten entender tanto el proceso técnico que se utiliza para desarrollar un producto, como el propio producto. El proceso para intentar mejorarlo, midiendo el producto para intentar perfeccionar su calidad. Existen disciplinas dentro del tema Métricas entre las que se encuentran las cuales se aplican durante todo el proceso de desarrollo de un proyecto de software. El uso fundamental de estas métricas es proveer de información al

equipo de desarrollo. El valor de la información está en analizar los datos de tendencias, día a día. Dentro de las disciplinas antes mencionadas se encuentran Métricas para el Proceso, Métricas para el Producto y Métricas par el Proyecto. La investigación se centra en el estudio de las Métricas para el Proyecto debido a la importancia que estas implican para obtener un producto con la calidad esperada.

1.10 Métricas en el Proyecto

Las Métricas en el proyecto permiten entender, monitorizar, controlar, predecir y probar el desarrollo de proyectos de software. Estas métricas evalúan el estado del proyecto actual, realizan un seguimiento de los riesgos potenciales, detectan aéreas de problemas antes de que se conviertan en críticas, ajusta el flujo y las tareas de trabajo. De forma general las métricas son utilizadas para que profesionales e investigadores tomen mejores decisiones que actúen directamente en la mejora continua del proyecto.

Una Métrica de un proyecto es la medida de alguna propiedad de un entregable del proyecto o del proceso de administración de proyecto, efectuada para conocer el avance o los desvíos al Plan original. Las métricas son utilizadas para medir el estado, efectividad o progreso de las actividades de un proyecto y así contribuir a la toma de decisiones estratégicas ante los desvíos, incidentes o diferentes problemas que surgen en la ejecución.

1.11 Características de las Métricas para el proyecto

Las Métricas para el Proyecto permiten al Jefe de Proyecto de software y a la alta dirección evaluar el estado de un proyecto terminado o en ejecución así como estimar el desempeño de un nuevo proyecto. Seguir la pista de los riesgos potenciales basada en experiencias anteriores. Detectar los procesos de realización de software con problemas antes de que se conviertan en críticos, ajustar el Proceso de Realización de Software. Evaluar la habilidad del equipo de proyecto y de las Áreas productoras divisionarias.

1.12 Indicador

Un indicador es una métrica o una combinación de métricas que proporcionan una visión profunda del proceso del software, del proyecto de software o del producto en sí.

Los indicadores son una medida que proporciona la evaluación o estimación de determinados atributos derivados de un modelo con respecto a las necesidades de información definidas, no son más que la base para el análisis y la toma de decisiones.

1.13 Aplicación de métricas para proyectos de software en el mundo

Durante muchos años el proceso de desarrollo de software ha sido considerado como un arte dejado a la improvisación del jefe del proyecto. Los proyectos se dirigían guiados por las consideraciones técnicas, y no por una correcta Gestión. Las actividades de estimación y de planificación quedaban relegadas a un mero acto protocolario al comienzo del proyecto. Posteriormente, el seguimiento y control se realizaban sin un mínimo de rigor, dada la baja calidad de la estimación y la planificación realizada. Las desviaciones en costos y tiempo han sido consideradas como algo natural en un proyecto software. No se recogen datos sobre el proceso de desarrollo del software, sin datos históricos como guía, la estimación de los costos no es buena y los resultados previstos muy pobres, sin una indicación sólida de la productividad, no es posible evaluar con precisión la eficacia de las nuevas herramientas, técnicas o estándares.

La Industria del Software actual afronta una crisis debido a una serie de factores entre los que se encuentra:

- Insuficiente calidad del producto final.
- Estimaciones de duración de proyectos y asignación de recursos inexactas.
- Retrasos en la entrega de productos terminados, están fuera de control los costos de desarrollo y mantenimiento de productos.
- Escasez de personal calificado en un mercado laboral de alta demanda.
- Tendencia al crecimiento del volumen y de la complejidad de los productos.

De manera general las empresas productoras de software están reconociendo la importancia que tienen las mediciones y con ellas las métricas para lograr producir software con calidad. En empresas que se dedican exclusivamente a la informática, se tiene noción de la necesidad de formalizar los mecanismos de estimación, comprendiendo que los registros históricos de antiguos proyectos realizados podrían ser de mucha ayuda a la hora de estimar con exactitud el esfuerzo, tiempo de desarrollo, costo, posibles errores, recursos y tamaño para los nuevos proyectos. Es válido aclarar que en ocasiones los resultados de los procesos de medición no son interpretados de la mejor manera, pues aún existen compañías que no tienen una cultura adecuada sobre la medición, desconociendo el alcance de madurez y calidad que pudiera alcanzar el producto final.

Existe una extensión del Trac que se llama TracMetricsPlugin, en esta extensión están incluidas varias métricas, dentro de las funcionalidades de la extensión se encuentra el llevar un control de las tareas o sistema de tickets que se maneja dentro del trac, mostrando gráficos de tendencias respecto a los tickets y también a cerca de las versiones y las actualizaciones que se realizan al Subversion. Es muy útil pero no permite introducir valores o nuevas métricas las cuales sean de interés de proyectos de software de manera particular solo e basa en la arquitectura y la información que tiene el Trac por defecto.

Varios estándares y modelos de madurez han incluido métricas de software, a continuación se muestran algunos de ellos (PROCESS 2001)

ISO 15504

Incluye dentro de la categoría de los procesos organizacionales (en la segunda parte de la norma) al proceso de medición, incluyendo la definición de métricas, la gestión de los datos (incluidos los históricos), y el uso de las métricas en la organización.

Familia ISO 9000-2000

Se establece la necesidad de implementar el proceso de medición con el objetivo de controlar la calidad del producto, la capacidad del proceso y la satisfacción del cliente, la gestión utiliza métricas como entrada fundamental para la planificación, control y gestión del proyecto.

CMMI

Incorpora un área de proceso denominada "Medición y Análisis", cuyo objetivo es desarrollar y establecer una capacidad de medición que se pueda usar para dar soporte a las necesidades de información de la organización y que proporcionen resultados objetivos que sean útiles para la toma de decisiones y acciones correctivas.

PSM

PSM constituye el documento base a partir del que se ha elaborado el nuevo estándar ISO/IEC 15939 sobre la medición del software, para la que proporciona detalles adicionales respecto de las actividades y tareas.

1.13.1 Métricas en Cuba

La producción de software en Cuba se encuentra en un proceso de evolución pues inicialmente se producían pocos software y no existían empresas que se dedicaran específicamente a esto, y actualmente ya existe una cultura de producción de software existiendo numerosas empresas dedicadas a esta rama de la economía. Incidir en el desarrollo de la industria informática y establecer en ella parámetros de excelencia es imprescindible. Para esto es necesario medir el producto los procesos y los proyectos. Las mediciones, a pesar de ser una de las áreas en la ingeniería de software donde se ha investigado hace muchos años, no han sido bien comprendidas, ni aplicadas. La implementación de las mediciones requiere un cambio cultural importante. Como parte de las mediciones se encuentran las métricas, como antes se dijo se ha investigado mucho, pero aun no se han aplicado los conocimientos adquiridos en ese campo. A pesar de que se reconoce su importancia en el momento de realizar un trabajo con calidad, y hacer correctas estimaciones de esfuerzo, costo y tiempo.

1.13.2 Métricas en la Universidad de las Ciencias Informáticas

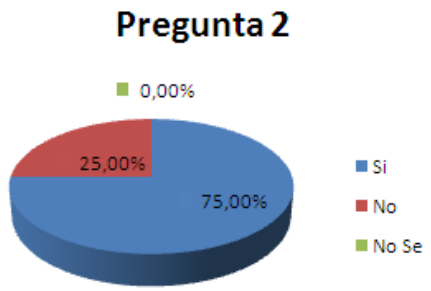
En la Universidad de las Ciencias Informáticas también se ha investigado en estas disciplinas de la Ingeniería de Software, pero al igual que en el resto del país aun queda mucho por hacer en este sentido. El proceso de estimaciones y las mediciones se tiene actualmente concebido como un proceso poco productivo. Pero esto no es acertado pues, estas mediciones se utilizan para corregir errores a tiempo que pudieran producirse durante el proceso de desarrollo del software. Existen insuficiencias en la calidad del producto final, así como estimaciones de duración de proyectos y asignación de recursos inexactas. También hay retrasos en la entrega de productos. Muchos de estas ineficiencias o errores podrían ser eliminados realizando mediciones y con la aplicación de métricas.

Como parte de la investigación, se realizó un estudio de los principales proyectos dentro de la Universidad y se aplicó una encuesta con el objetivo de saber la realidad en cuanto al empleo de métricas, y la realización de mediciones y estimaciones. Partiendo del análisis de la encuesta realizada se puede concluir que se obtuvieron resultados ilustrativos, por ejemplo varios directivos o miembros de los proyectos han realizado estudios sobre métricas el 76.92% del total realizó los estudios antes mencionados sin embargo aun así el conocimiento sobre el tema es bastante bajo, se puede apreciar que el 53.85% de las personas encuestadas, consideran que su el conocimiento sobre el tema es bajo. En el 75.00% de los proyectos no se guardan registros históricos esto puede traer como consecuencia errores de estimaciones en caso de que estas se realicen pues por ejemplo en la universidad el 76.92% de los proyectos si realizan estimaciones pero un 15,38% de ellos no las realizan, y en el 73.33% de los proyectos de la UCI no aplican métricas.

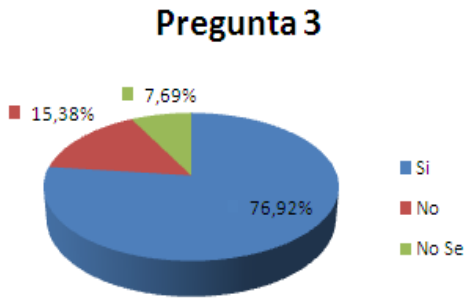
En el polo de Hardware y Automática se realizó un estudio similar, investigando sobre la aplicación de métricas y la salva de los datos de las mediciones en caso que se realicen. Finalmente se concluyó que dentro del polo existen los mismos problemas que arrojó la encuesta. Pues no se hacen correctas estimaciones debido a la no existencia de registros históricos.

A continuación se muestran algunos de los gráficos ilustrando los resultados de dicha encuesta en el formato de pregunta y seguidamente el gráfico, ver anexo # 2.

Pregunta # 2 ¿Su proyecto tiene almacenados registros históricos?



Pregunta # 3 ¿Realizan estimaciones?



Con el objetivo de aportar una solución y contribuir a mejorar este proceso en la universidad y en el polo, se propone la implementación de una extensión para incorporárselo al Trac, la extensión contendrá un paquete de métricas que permitan guardar los datos históricos de las mediciones que se realizan dentro de los proyectos de software. A continuación se describirán las tecnologías a utilizar para la realización de la extensión.

1.14 Tendencias y Tecnologías actuales

1.14.1 Sistema Operativo GNU/Linux

GNU/Linux es un término que se utiliza al referirse a sistemas operativos que son similares a Unix, pero que son basados en el núcleo de Linux, el cual es el encargado de administrar la memoria adecuadamente, repartir el tiempo de procesamiento para todos los programas y comunicarse con los dispositivos de almacenamiento y aplicaciones instaladas para guardar los archivos necesarios.

Gran parte de las herramientas que contiene el sistema, vienen dada por el proyecto GNU, de ahí que su nombre sea GNU/Linux, presentando gran estabilidad y garantizando que los usuarios trabajen de manera confiada y segura, a través de las actualizaciones realizadas por el sistema. Es un sistema operativo multitarea, multiusuario y multiplataforma, permite ejecutar hasta siete terminales virtuales que pueden ser utilizadas por diferentes usuarios y posee un poderoso entorno gráfico, que se encuentra basado en el modelo cliente – servidor, soportando conexiones de red local, utilizando el protocolo TCP/IP.

Es totalmente gratuito y aunque posea versiones que se pagan, es aún más barato que comprar Windows, las distribuciones tienen muchos programas útiles que son fácilmente instalables y desinstalables, y totalmente seguros, sin versiones de prueba o peligrosos freeware con virus, no necesita reiniciarse a menos que haya una modificación directa en el núcleo y es completamente configurable y optimizable en todos sus aspectos, facilitando que corra sobre cualquier sistema sin importar el hardware.

Su gran estabilidad ha hecho de este sistema operativo una opción muy a tener en cuenta por aquellos usuarios que se dediquen a trabajar a través de redes, naveguen por Internet o se dediquen a la programación.

Esta investigación será desarrollada a partir de las herramientas que brinda el software libre. Utilizando su independencia tecnológica al posibilitar la libertad de uso y distribución de los programas.

1.14.2 Distribución de Linux: Debian GNU/Linux

Una distribución GNU/Linux es un agrupamiento del núcleo del sistema operativo (Linux) y otra serie de aplicaciones de uso general. Su objetivo es facilitar la instalación, la configuración y el mantenimiento de un sistema GNU/Linux. En la actualidad existen una gran variedad de distribuciones cada una de ellas creada para satisfacer las necesidades de los usuarios, brindándoles facilidad y seguridad a la hora de usarlo, entre las más importantes se encuentran: Ubuntu, Knoppix, RedHat, SuSE y Debian, siendo esta última la más utilizada en el Polo de Hardware y Automática en la Universidad de las Ciencias Informáticas.

Debian o Proyecto Debian (en inglés Debian Project) es la única distribución de GNU/Linux conformada por usuarios voluntarios que pretenden crear y mantener un sistema operativo GNU basado en software libre. Es la única de las grandes distribuciones que no tiene intereses comerciales ni empresariales, son sus propios usuarios, quienes mantienen la distribución de modo comunitario, actualizando la distribución diariamente, auxiliando a través de las listas de correo a los usuarios que pertenezcan a la comunidad e implementando nuevos paquetes de software. Colectivamente, existe un compromiso de disposición y calidad, permitiendo para ello que no se distribuyan software con errores. El objetivo de esta gran comunidad, es recopilar, difundir y promover el uso del software libre, proporcionando un sistema operativo maduro, estable y configurable.

Lo que más distingue a Debian de otras distribuciones GNU/Linux es su sistema de gestión de paquetes. Estas herramientas dan al administrador de un sistema Debian, control completo sobre los paquetes instalados en su sistema, incluyendo la capacidad de instalar un sólo paquete o actualizar el sistema operativo por completo.

Se decidió usar Debian porque presenta una amplia colección de software disponible, así como un grupo de herramientas que facilitan el proceso de instalación y actualización. A diferencia de otras distribuciones, no tiene marcado ningún entorno gráfico y posee un magnífico soporte de estabilidad en las aplicaciones. Los módulos del LDAP (Lighweight Directory Access Protocol) se pueden ejecutar sin problemas permitiendo que los usuarios usen sus sesiones en cualquier máquina dentro del área de trabajo, ahorrando recursos de hardware. Además de ser la distribución utilizada por el proyecto SCADA Nacional, al cual va dirigida la aplicación de la propuesta de la investigación es decir la extensión.

1.14.3 Herramientas para la gestión de proyectos

En la actualidad, el uso de las nuevas tecnologías de información ha traído consigo la utilización de herramientas que posibilitan la recopilación, la seguridad y la administración sistemática de la información.

Las herramientas para la Gestión de Proyecto pueden definirse como aquellas herramientas o instrumentos que soportan la realización de aplicaciones, actividades o acciones como la generación, codificación o transferencia del conocimiento (Ruggles, 1997).

Dichas herramientas permitirán almacenar y gestionar la información obtenida o generada durante el desarrollo de los proyectos. Cada hito, tarea o subtarea puede implicar la obtención o generación de documentación (actas de reuniones, talleres, seminarios, o investigaciones, etc.); el sistema debe permitir un mayor rendimiento, al reducir los tiempos para el almacenamiento, localización, y búsqueda de la información.

1.14.4 Herramienta Trac

La herramienta Trac es un sistema Web libre para la gestión de proyectos y seguimiento de errores, es ligera, simple, extensible y permite enlazar información entre una base de datos de errores de software, un sistema de control de versiones y el contenido de una Wiki.

Cubre las necesidades técnicas para el desarrollo de proyectos de software. Integra una Wiki, que permite mantener activa y en uso la documentación, una vista de los cambios recientes (Timeline), un control de hitos (Roadmap) para conocer el estado del desarrollo del proyecto, una interface para la revisión del código fuente (Browse Source), una gestión de bugs (Tickets) con posibilidad de abrir, asignar y cerrar incidencias y un potente buscador (Search).

Está desarrollado en torno a la idea de un núcleo al que se le pueden añadir Extensiones que proporcionan distintas funcionalidades (casi todos los componentes estándar son módulos que pueden ser activados, desactivados o reemplazados o modificados por otros).

EL Trac posee varias características, dentro de ellas se puede destacar una visión abierta de la gestión de proyectos, no obliga a utilizar ninguna metodología determinada, proporcionando para ello tres elementos básicos: un gestor de hilos, un navegador de control de versiones (svn) y un sistema de control de tareas. Todo esto a través de una interfaz Web con edición de tipo wikipedia. Otras características que presenta esta herramienta es que es un proyecto que no requiere la compra de licencias para el uso de la misma, sino que permite la inserción de errores a través de órdenes de trabajo como son los tickets, para el control de versiones en el desarrollo, brindando además facilidades para la administración desde la web para los usuarios, componentes, etc., y permitiendo chequear desde este ambiente web todas las actualizaciones realizadas, así como mantener el control de acceso y la asignación de tareas usando los usuarios que ya se poseen para el control de versiones. Permite la definición de las listas para la mayor parte de los campos, tales como componentes, versiones, hitos, clasificaciones de errores por tipos, prioridades, etc.

A diferencia de otras plataformas de gestión de proyectos, no se lleva una contabilidad tan estricta del tiempo dedicado al proyecto por cada uno de los miembros, sino que persigue que el desarrollador tenga mayor naturalidad a la hora de trabajar. La función de esta herramienta es servir como aplicación de gestión de tareas, que permita cruzar los datos de las tareas con las introducciones en el sistema de control de versiones.

La herramienta Trac puede ser configurada de tres maneras diferentes, como módulo de python junto con apache, como un sitio por cgi-bin o como un servidor independiente. El Trac funciona desde un servidor web, que puede ser uno propio (que sería en este caso el trac) o puede ser uno estándar (dígase lighttpd, apache2) sirviendo de soporte para ejecutar código python usando scripts de CGI, FastCGI o mod_python.

1.14.5 Extensiones para el Trac

Las Extensiones son un módulo de software, que se le añaden a un sistema más grande para aumentar sus funcionalidades, en este caso el Trac, sin afectar las ya existentes que posee dicho sistema. En el Trac se encuentra las siguientes extensiones:

- Autenticación con formularios y usuarios en LDAP, BD o fichero.
- Uso de otros VCS como Bzr, GIT, Mercurial o Monotone.
- Servicios adicionales como blogs, foros, etc.

La implementación de las extensiones ofrece nuevas ventajas, algunas de ellas se puntualizan a continuación:

- Sencillez y limpieza en la instalación: debido a la arquitectura modular, se puede decir que una extensión se conecta con la aplicación (en este caso Trac) de la manera más limpia y sin la necesidad de configurarse. Por lo general, una extensión puede ser eliminado de un modo similar sin afectar siquiera al funcionamiento habitual de la aplicación. La extensión se comunica con la aplicación destino manejando así los

mecanismos que son proporcionados por la aplicación, siendo la integración total con la plataforma. El conjunto completo de la aplicación adicionado a los extensión hacen visible al usuario como un todo sin presentar división, ya que en todo momento el usuario tendrá la sensación de estar manejando las funcionalidades suministradas por la aplicación mientras que la realidad se trata únicamente de un conjunto de elementos independientes trabajando en unión.

- Multiplataforma: este es el caso en que la aplicación (Trac) se encuentra disponible para otras plataformas, el cambio de las extensiones seria insignificante ya que por lo general, se utilizaría la misma interfaz de comunicación.

1.14.6 Lenguaje de Programación

Python es un lenguaje de programación orientada a objetos creado por Guido Van Rossum, actualmente es desarrollado y mantenido por una comunidad de desarrolladores de código abierto bajo la control de la Python Software Foundation.
Es un lenguaje interpretado, logrando ser modificado instantáneamente sin necesidad de ser recompilado, lo cual ayuda a hacer mejoras de forma rápida y simple, es multiplataforma facilitando que cualquier sistema operativo sea compatible con el, siempre y cuando exista un intérprete programado para el. Posee una de las documentaciones más grandes y detalladas en el mundo de la informática, admitiendo varios estilos de programación: Programación orientada a objetos, programación estructurada y programación funcional.

Permite mantener de forma sencilla la interacción con el sistema operativo, y resulta muy adecuado para manipular archivos de texto. Esta característica hace que muchas distribuciones de GNU/Linux utilicen Python para sus herramientas de configuración. También es ampliamente utilizado en la Web. Actualmente tiene una licencia compatible con GPL, lo que significa que se puede modificar y distribuir libremente su código.

La sencilla sintaxis de Python y el poder ejecutar en tiempo real el código, hacen de el un lenguaje ideal para la implementación de Extensiones para la herramienta Trac. En Python no es necesario declarar el tipo de dato que va a contener una determinada variable, sino que su tipo se determinará en tiempo de ejecución según el tipo del valor al que se asigne, y el tipo de esta variable puede cambiar si se le asigna un valor de otro tipo.
Python es adecuado para tareas de manipulación de texto. Con la ayuda de sus bibliotecas permite que varias aplicaciones trabajen conjuntamente, siendo usado para desarrollar entornos gráficos de aplicaciones, disponiendo de un intérprete por línea de comandos en el que se pueden introducir sentencias. Cada sentencia se ejecuta y produce un resultado visible, que puede ayudar a entender mejor el lenguaje y probar los resultados de la ejecución del código rápidamente.

1.14.7 Lenguaje de Modelado NET BEANS

NetBeans es un proyecto exitoso de código abierto con una gran base de usuarios, una comunidad en constante crecimiento, fue fundado por Sun MicroSystems en junio 2000 y continúa siendo el patrocinador principal de los proyectos de código abierto. Actualmente se encuentran disponibles dos productos: el NetBeans IDE y NetBeans Platform. NetBeans IDE es un entorno de desarrollo - una herramienta para que los programadores puedan escribir, compilar, depurar y ejecutar programas. Está escrito en Java pero puede servir para cualquier otro lenguaje de programación. Existe además un número importante de módulos para extender el NetBeans IDE. NetBeans IDE es un producto libre y gratuito sin restricciones de uso. También está disponible NetBeans Plataform: una base modular y extensible usada como estructura de integración para crear grandes aplicaciones de escritorio. Empresas independientes asociadas, especializadas en desarrollo de software, proporcionan extensiones adicionales que se integran fácilmente en la plataforma y que pueden también utilizarse para desarrollar sus propias herramientas y soluciones. Ambos productos son de código abierto y gratuito para uso tanto comercial como no comercial. El código fuente está disponible para su reutilización de acuerdo con la Common Development and Distribution License (CDDL) v1.0 and the GNU General Public License (GPL) v2.Dispone de soporte para crear interfaces

gráficas de forma visual, desarrollo de aplicaciones web, control de versiones, colaboración entre varias personas, creación de aplicaciones compatibles con teléfonos móviles, resaltado de sintaxis y además sus funcionalidades son ampliables mediante la instalación de packs.

1.14.8 Metodología de Desarrollo de Soft: Open-Up

Open Up es un proceso de desarrollo de software completo ya que puede ser manifestado como todo el proceso para construir un sistema. Es extensible ya que en el proceso se puede agregar o adaptar según lo vayan requiriendo los sistemas, sin embargo solo esta incluido lo fundamental (menos de 20 artefactos). Open UP es ágil, ligero y proporciona una comprensión detallada del proyecto, beneficiando a clientes y desarrolladores sobre productos a entregar y su formalidad. Se centra en una arquitectura temprana para reducir al mínimo los riesgos y organizar el desarrollo. Es la metodología utilizada por desarrolladores de alto nivel en todo el mundo por sus altas cualidades administrativas. Su desarrollo es dirigido por casos de uso.

1.14.9 Eclipse

La plataforma Eclipse consiste en un Entorno de Desarrollo Integrado (IDE, Integrated Development Environment) abierto y extensible. Un IDE es un programa compuesto por un conjunto de herramientas útiles para un desarrollador de software. Como elementos básicos, un IDE cuenta con en un editor de código, un compilador/intérprete y un depurador. Eclipse sirve como IDE Java y cuenta con numerosas herramientas de desarrollo de software. También da soporte a otros lenguajes de programación, como son C/C++, Cobol, Fortran, PHP o Python. A la plataforma base de Eclipse se le pueden añadir extensiones (extensiones) para extender la funcionalidad. El término Eclipse además identifica a la comunidad de software libre para el desarrollo de la plataforma Eclipse. Este trabajo se divide en proyectos que tienen el objetivo de proporcionar una plataforma robusta, escalable y de calidad para el desarrollo de software con el IDE Eclipse. Este trabajo está coordinado por la Fundación Eclipse, que es una organización sin ánimo de lucro creado la promoción y evolución de la plataforma Eclipse dando soporte tanto a la comunidad como al ecosistema Eclipse.

1.14.10 Matplotlib

Matplotlib es una potente librería gráfica, muy parecida en su concepción a los comandos gráficos de Matlab, con los que cualquiera puede hacer unas gráficas científicas más o menos apañadas. Matplotlib es una biblioteca para la generación de gráficos a partir de datos contenidos en listas o arrays en el lenguaje de programación Python y su extensión matemática NumPy. Proporciona una API, pylab, diseñada para recordar a la de MATLAB. Software grafico SO: multiplataforma, Última versión**:** 0.98.5.2 (2008-12-11) Matplotlib es una librería de trazado de gráficos del python puro con la meta de los diagramas de la calidad del makingpublication usando a un familiar del sintaxis a los usuarios del matlab. Las aplicaciones de la biblioteca numéricas para dirigir largedata fijan y apoyan una variedad de salida se centralizan Descripción: Matplotlib es una librería de funciones para integrar a Python la capacidad de generar gráficas en formatos apropiados para Agg, Cairo, GTK, GTKAgg, GTKCairo, PS (postscript), TkAgg, WX, WXAgg, Paint, GD, EMF. El objetivo es que los alumnos aprendan a generar gráficas con esta herramienta.

1.14.11 Sqlite

SQLlite es una pequeña librería programada en lenguaje C que implementa motor de base de datos multiplataforma que no precisa configuraciones, es rápido y la ventaja fundamental es que permite utilizar un amplio subconjunto del lenguaje estándar SQL. Combina el motor y la interfaz de la base de datos en una única biblioteca, y almacena los datos en un único archivo de texto plano. Esto hace que cada usuario pueda crear tantas bases de datos como desee sin la necesidad de la intervención de un administrador de bases de datos que gestione los espacios de trabajo, usuarios y permisos de acceso. Su potencia se basa fundamentalmente en la simplicidad, lo que hace que no sea una buena solución en entornos de tráfico muy elevado. SQLite encapsula toda la base de datos en un único fichero.

De manera general podemos decir que en el desarrollo del capítulo actual se ha realizado un estudio del estado del arte de las empresas con respecto a la utilización de métricas. También

se estudiaron los conceptos fundamentales como son métricas de software, métricas para el proyecto entre otras, también se hizo un profundo estudio de las tendencias y tecnologías actuales para poder seleccionar las más apropiadas para darle solución a los problemas planteados entre las principales se encuentra el lenguaje para el desarrollo del extensión así como el IDE de programación a utilizar y el IDE para el diseño del extensión como tal.

Capítulo 2: Diseño de la Solución

Introducción

En el presente capítulo se propondrá un paquete de métricas que contribuyan a mejorar o corregir algunos de los principales problemas existentes en el Polo de Hardware y Automática de la facultad 5 en cuanto a guardar los datos históricos de las mediciones que se realicen dentro de los proyectos. Se analizará la arquitectura que tendrá la extensión, así como los casos de uso que describen el proceso de recolectar los datos históricos.

2.1 Situación actual de los proyectos del Polo de Hardware y Automática

En los proyectos del Polo de Hardware y Automática de la facultad 5 se han creado nuevos proyectos que en el momento de realizar la planificación inicial estimaron sin contar con valores históricos que sustente dichas estimaciones, por lo que la planificación corre el riesgo de no ser la más acertada. Para tratar de dar solución a algunos de estos problemas surge la propuesta de esta investigación, la implementación de una extensión que contenga un paquete de métricas para adicionarlo al Trac y así poder guardar los datos históricos de todas las mediciones que se realicen dentro de cualquier proyecto del polo.

2.2 Propuesta del paquete de Métricas

Según el estudio realizado tanto en la universidad como las necesidades existentes en el polo se decidió escoger las siguientes métricas:

Métrica LOC:

Cantidad de líneas de código fuente. (La cantidad está en dependencia del lenguaje en que se este programando.)

Métrica HPD:

Horas del programador diarias. A mayor número de horas mejor resultado más rápido acabará proyecto, recordar que siempre va a ser menor que 24h (día completo).

Métrica CED:

Cantidad de errores al culminar la jornada diaria. Entre menor sea el número de errores mejor será mientras más se acerca a 0.

Métrica Personas:

Cantidad de personas en el proyecto. (Debe estar en dependencia del proyecto, del tiempo que se necesita para terminar si son muchos, algunos no tendrán trabajo, si son pocos la carga será mucha y no se logrará terminar a tiempo).

Métrica HPT:

HPT es la sumatoria de todas las horas diarias del programador (HPD), esta métrica se expresa en unidades de tiempo. Es el total de horas que el programador dedicó una tarea o proyecto en si.

Métrica CHF:

CHF = **LOC/HPD**

Donde:

LOC: Líneas de códigos fuente

HPD: Horas del programador diarias.

Métrica que se encarga de medir la productividad del programador, mientras mayor sea el resultado mayor será la productividad del programador.

Métrica Esfuerzo:

Esfuerzo = **Personas** / **Meses**

Métrica que se encarga de medir el esfuerzo del equipo de desarrollo, mientras mayor sea el resultado, mayor esfuerzo se habrá realizado.

Métrica Calidad:

Calidad = (**D** / **KLDC**)

D: Número de defectos encontrados

LOC: Líneas de código fuente

D: Defectos

Mientras menor sea el número de errores mayor será la calidad, mientras sea más cercano a 0 mejor será el resultado.

Estimación de Errores:

EED = **E**/ (**E** + **D**)

Donde:

E: Errores encontrados antes de entregar el software al usuario.

D: Defectos encontrados después de la entrega

Valor ideal EED = 1 No se encontraron defectos— mientras más cercano a 1 sea el valor menor será la cantidad de errores por lo que será mejor resultados)

Métrica Por ciento de tareas completadas:

Es de vital importancia llevar un registro y mediciones de las tareas que se desarrollan durante el proyecto, pues de esta manera pueden realizarse mejores asignaciones de recursos y tiempo así como tener una medida del progreso del trabajo realizado respecto al planificado teniendo en cuenta el cumplimiento de las tareas.

El porciento de tareas completadas se calcula mediante la fórmula:

TaC% = (TaC/TaP) *100

Donde:

TaC: Número de tareas completadas

TaP: Número de tareas planificadas

TaC %: Porciento de tareas Completadas

Análisis del Procedimiento:

A medida que el porciento de tareas completadas se acerca a 100% será mejor pues el número de tareas completadas estará próximo al número de tareas planificadas. Esta métrica proporciona una medida de cuántas tareas se han completado con respecto a las planificadas al acercarse un determinado hito o fase en el desarrollo del software, siendo una indicación del progreso alcanzado.

Métrica Productividad:

El término productividad tiene diferentes interpretaciones en las distintas bibliografías, en algunos casos se refiere a la producción de código por hora y en otros al esfuerzo necesario para desarrollar un software, en sentido general puede definirse como la relación que existe entre el tamaño del software por unidad de tiempo o esfuerzo realizado. Las medidas de productividad más utilizadas están basadas en líneas de código o puntos de función. A medida que se avanza en los ciclos de desarrollo del proyecto, la productividad se incrementa.

La productividad está definida por las siguientes ecuaciones:

Productividad = LOC/Horas

Productividad = KLDC/Esfuerzo

Productividad = PF / Esfuerzo

Donde:

LOC: Líneas de código

KLDC: Miles de líneas de código

PF: Puntos de Función.

Métrica Por ciento de error de estimación de Tamaño:

La estimación es una de las actividades prioritarias en la gestión de proyectos, por lo que es importante estimar el tamaño basado en datos históricos. La estimación del tamaño es un punto de partida para realizar cálculos y estimaciones de tiempo, costo y esfuerzo, por lo general el tamaño puede medirse en líneas de código, puntos de función, puntos de características, entre otros, aunque en el (Personal Software Process) PSP por sus siglas en ingles y en español significa Proceso Personal de Software) se define esta métrica basada en líneas de código. La hipótesis a probar es: "a través de los diferentes ciclos de desarrollo del proyecto la estimación de tamaño se ajusta gradualmente al tamaño real del producto a realizar".

El por ciento de error de estimación del Tamaño está dado por:

EET % = ((**TR** - **TE**) / **TE**)*100

EET = **TR** – **TE**

TE -------- 100%

EET ------ **EET** %

Donde:

EET %: Por ciento de Error de Estimación del Tamaño.

EET: Error de Estimación Tamaño.

TR: Tamaño Real.

TE: Tamaño Estimado.

En esta métrica mientras menor sea el por ciento de error de estimación de tamaño será mejor pues indicará que el error está oscilando en valores cercanos a cero, lo cual significa que el tamaño real está próximo al estimado por lo tanto no se ha sobre estimado o subestimado

durante la planificación. Si el valor de por ciento es negativo significa el tamaño real estuvo por debajo del estimado y en caso contrario el tamaño real estuvo por encima del estimado. Teniendo registros de cuanto puede desviarse el tamaño real del software respecto al planificado pueden realizarse mejores estimaciones para futuros trabajos con características similares, de manera que pueda minimizarse lo más posible el por ciento de error en la estimación del tamaño.

Métrica Porciento de error de estimación de Tiempo:

De manera análoga a la métrica anterior también puede analizarse el error de estimación del tiempo. Estos registros ayudarán a hacer mejores estimaciones de tiempo para trabajos futuros. El porciento de error de estimación de Tiempo está dado por la siguiente ecuación:

EETi % = ((**TiR** - **TiE**) / **TiE**)* 100
EETi = **TiR** – **TiE**
TiE -------100%
EETi ----- **EETi** %

Donde:
EETi: Error Estimación Tiempo
TiR: Tiempo Real
TiE: Tiempo Estimado

Métrica por ciento de cumplimiento de tareas en el período:

Esta métrica ayudará a tener una idea de cuanto se ha avanzado en la realización o el cumplimiento de las tareas planificadas. Cuando este valor sea mayor y tienda a 100% entonces el resultado será mejor.

TCf/TPf * 100

Donde:

TCf= tareas completadas a la fecha

TPf= tareas planificadas a la fecha

2.4 Arquitectura

La extensión que se desea implementar utilizará la arquitectura del Trac, a continuación se muestran las características fundamentales de la misma.

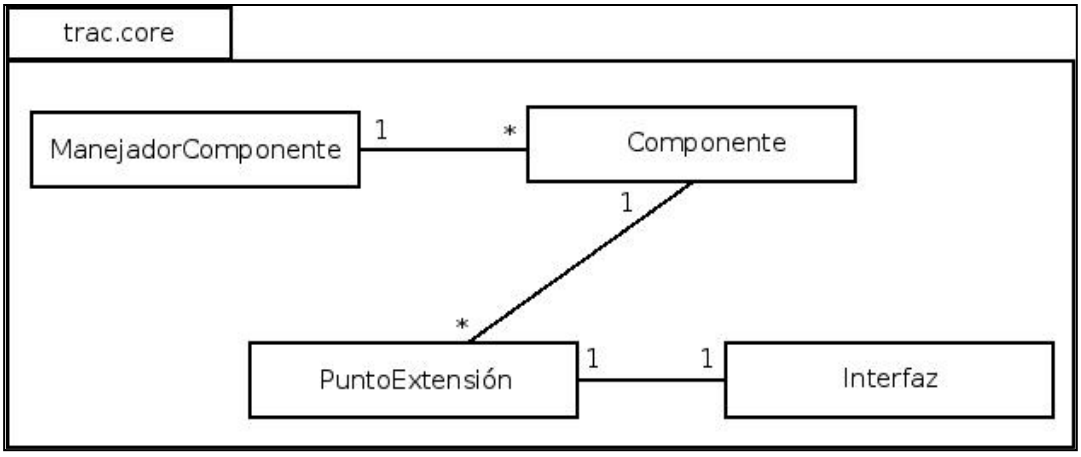


Figura 3: Arquitectura del Trac.

Dentro del sistema arquitectónico que se muestra, la herramienta para el trabajo con métricas a desarrollar en la investigación quedaría enmarcada dentro del subsistema Componente. En la figura se muestra que el trac esta compuesto por un núcleo. Dentro del núcleo del trac se tienen varios subsistemas como son ManejadorComponente, Componente, PuntoExtensión e Interface. El subsistema ManejadorComponente esta formado por uno o varios Componentes este subsistema es el encargado de que en todo momento exista solo una instancia de cada componente en el sistema. Aplicando así el patrón Singleton. El subsistema Componente no es más que los servicios o funcionalidades que brinda el dicho componente. Debido a que la aplicación a realizar solo agregará funcionalidades al trac entonces encaja perfectamente en el

subsistema Componente. También existen los PuntoExtensión que son como su nombre lo indica puntos de extensión que se le hace a los componentes permitiendo crear interfaces del nuevo componente que pueda ser implementado por otros componentes.

2.5 - Diseño del Extensión

Según las necesidades observadas en el polo y con el objetivo de tener un registro de los datos históricos se definieron las siguientes funcionalidades.

2.5.1 - Funcionalidades definidas

RF1- Permitir Insertar modificar o eliminar una métrica.

RF2- Registrar los valores tanto de las mediciones como de las métricas.

RF3- Realizar el cálculo de una métrica determinada.

RF4- Visualizar el comportamiento de una métrica en el tiempo.

RF5- Mostrar un reporte general de las tendencias de cada una de las métricas.

2.5.2 Requisitos no Funcionales

Usabilidad

- La Extensión luego de instalado deberá visualizarse con calidad en los principales navegadores existentes (Internet Explorer, Opera, Firefox y otros).

Fiabilidad

- El sistema debe ser capaz de mantener la calidad del dato de manera que garantice su integridad durante su aplicación, procesamiento y almacenamiento en el BD.
- Cada vez que se detecta un cambio en la BD se debe replicar de manera automática los cambios.

Funcionamiento

- Los tiempos de respuestas del sistema serán de corta duración, así como la evaluación que se efectuará.

Soportabilidad

- El sistema debe ser:

- De fácil instalación, configuración y puesta en marcha.
- De arquitectura abierta y distribuida, modular, de capacidad escalable.
- Programado orientado a objeto.

- El sistema debe correr sobre un Servidor Apache 2.0 y utilizar SQLite como sistema gestor de Base de Datos.
- El sistema presenta dependencias con las siguientes librerías, lib mod_python, libapache2-mod-pam, python-genshi, libapache2-mod-python, python-matplotlib.

2.6 Casos de Uso Definidos

Para dar cumplimiento a las funcionalidades se identificó al actor y se definieron los siguientes casos de uso.

CU1- Gestionar Métricas.

CU2- Registrar Valores.

CU3- Realizar Cálculo.

CU4- Mostrar Reportes.

2.6.1 Identificar el actor

El actor del sistema será el usuario (líder del proyecto o encargado de realizar el proceso de medición en el proyecto) el cual podrá insertar modificar o eliminar alguna métrica, así como Registrar valores para las métricas en el sistema (Proceso de medición) y tener un reporte de la tendencia tanto de una métrica como de todas las métricas en el sistema.

2.6.2 Diagrama de CU

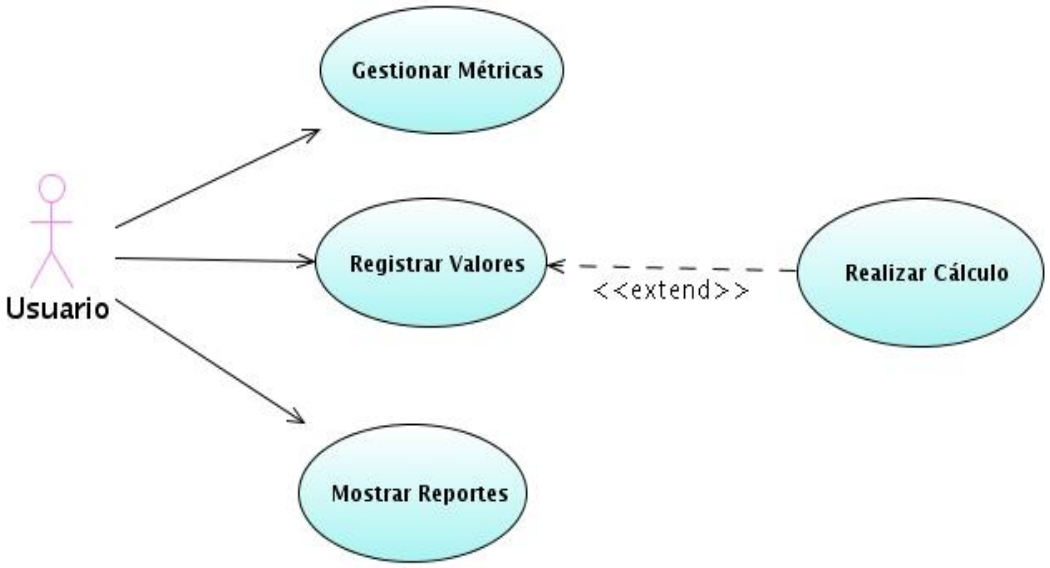


Figura 4: Diagrama de casos de uso.

2.6.3 Descripción de los CU

2.6.3.1 Caso de Uso: Gestionar Métricas.

Breve Descripción

Este caso de uso representa la funcionalidad de que un usuario pueda insertar, modificar o eliminar una métrica del sistema. Está dedicado completamente a la gestión de métricas dentro de la extensión.

Breve Descripción de los Actores

Actor: Usuario

El sistema cuenta con un único actor, el usuario, tiene privilegios de insertar, modificar o eliminar una métrica del sistema.

Precondiciones

El usuario debe estar autenticado.

El usuario debe asegurarse de la correcta introducción de los campos requeridos.

Flujo Básico de Eventos

1. El caso de uso comienza cuando el usuario, selecciona la opción gestionar métricas.
2. El sistema muestra una interfaz con tres opciones: insertar, modificar o eliminar una métrica.
3. El usuario desea insertar una nueva métrica.
4. El sistema muestra una interfaz con los campos requeridos para insertar una nueva métrica el sistema espera para cada una de las métricas los siguientes campos: Nombre, Expresión (Ecuación que la define), la unidad de medida así como una breve descripción de cada métrica.
5. El usuario inserta los valores requeridos para dicha métrica.
6. El sistema verifica la completitud de los datos introducidos por el usuario.
7. El sistema guarda la métrica en cuestión y permite seguir insertando nuevas métricas, además muestra una tabla con el listado actual de las métricas existentes en el sistema en ese momento.

Flujos Alternos

Modificar Métrica

Si en el paso 3 del flujo básico el usuario desea modificar una métrica existente en el sistema.

1. El sistema muestra una interfaz donde se muestra un listado de todas las métricas existentes en el sistema.
2. El usuario escoge la métrica que desea modificar.
3. El sistema carga en la interfaz todos los campos de la métrica seleccionada.
4. El usuario edita nuevamente la métrica, actualizando los campos deseados y

guarda los cambios.

5. El sistema verifica la completitud de los datos y actualiza los cambios realizados.

Eliminar Métrica.

Si en el paso 3 del flujo básico el usuario desea eliminar una métrica existente en el sistema.

1. El sistema muestra una interfaz con un listado de todas las métricas.
2. El usuario selecciona la o las métricas que desea eliminar y ejecuta la acción.
3. El sistema elimina y actualiza la interfaz con el listado actual de las métricas existentes en el sistema.

Postcondiciones

- El sistema inserta en la Base de Datos (BD) la métrica con sus valores correspondientes introducidos por el usuario. Los cuales son Nombre, Expresión, Unida de medida y una breve descripción y se actualiza la interfaz con los nuevos datos.
- El sistema actualiza en la BD la métrica con sus valores correspondientes introducidos por el usuario. Los cuales son Nombre, Expresión, Unida de medida y una breve descripción y se actualiza la interfaz con los nuevos datos.
- El sistema elimina de la BD la métrica con sus valores correspondientes seleccionada por el usuario los cuales son Nombre, Expresión, Unida de medida y una breve descripción y se actualiza la interfaz con los nuevos datos.

2.6.3.2 Caso de Uso: Registrar Valores

Breve Descripción

El caso de uso representa la funcionalidad de registrar tanto valores mediciones como de métricas el sistema.

Breve Descripción de los Actores

El sistema cuenta con un único actor el usuario, este actor tiene privilegios de registrar valores para cualquier métrica existente en el sistema.

Precondiciones

Deben existir métricas en el sistema para poder registrar los valores correspondientes.

El usuario debe asegurarse de la correcta introducción de los campos requeridos.

Flujo Básico de Eventos

1. El usuario selecciona la opción registrar valor.
2. El sistema muestra un listado de todas las métricas en el sistema y junto a cada una un vínculo a la interfaz registrar valor.
3. El usuario selecciona insertar valor a una medición.
4. El sistema muestra una interfaz con la posibilidad de insertar el valor de dicha medición.
5. El usuario inserta el valor de la medición.
6. El sistema valido la completitud de los datos requeridos.
7. El sistema guarda los valores en la base de datos y muestra una tabla con los valores históricos de la medición así como un gráfico con la tendencia de dicha medición en el tiempo.

Flujos Alternos

Si en el paso 4 el usuario escoge insertar valor a una métrica.

1. El sistema muestra una interfaz con los campos de la métrica seleccionada.
2. El usuario registra los valores requeridos.
3. El sistema valida la completitud de los datos.
4. El sistema realiza el cálculo
5. El sistema guarda los valores de dicha métrica y muestra un listado con los valores históricos.
6. El sistema muestra en un gráfico la tendencia de la métrica en el tiempo.
7.

Postcondiciones

El sistema registra en la BD los valores de la medición.

El sistema registra en la BD los valores de la métrica.

2.6.3.3 Caso de Uso: Realizar Cálculo

Breve Descripción

El caso de uso comienza cuando el usuario desea guardar el valor de una métrica para ello es necesario realizar el cálculo correspondiente.

Breve Descripción de los Actores

El sistema cuenta con un único actor el usuario, este actor tiene privilegios de registrar valores para cualquier métrica en el sistema.

Precondiciones

Deben existir métricas en el sistema.

Se debe haber seleccionado insertar valores para una métrica.

El usuario debe garantizar la completitud de los datos suministrado para realizar el cálculo según la operación definida.

Flujo Básico de Eventos

1. El usuario selecciona una métrica para guardar su valor.
2. El usuario introduce los datos de cada uno de los parámetros de la métrica seleccionada y especifica la operación matemática que se debe realizar.
3. El sistema realiza el cálculo según la operación matemática definida en la métrica.
4. Se guardan los valores del cálculo en la base de datos.

Postcondiciones

El sistema guarda en la BD el resultado del cálculo de la métrica.

Requerimientos especiales

Este caso de uso es extendido del caso de uso Registrar Valor

2.6.3.4 Caso de Uso: Mostrar Reporte

Breve Descripción

El caso de uso comienza cuando el usuario desea visualizar el comportamiento de una métrica o tener un reporte general, o un listado con todas las métricas existentes.

Breve Descripción de los Actores

El sistema cuenta con un único actor el usuario, este actor tiene privilegios ver tanto un reporte de tendencia de una métrica, como un reporte general de todas las métricas en el sistema.

Precondiciones

El usuario debe estar autenticado.

Flujo Básico de Eventos

1. El usuario selecciona la opción mostrar reporte.
2. El sistema muestra interfaz con dos opciones ver tendencia de una métrica o reporte general.
3. El usuario selecciona la opción ver tendencia de una métrica.
4. El sistema muestra una interfaz donde el usuario escoge la métrica de la cual desea ver la tendencia en el tiempo.
5. El sistema muestra un gráfico con la representación de la métrica en el tiempo.

Flujos alternos

Si en el paso 3 el usuario desea ver un reporte general.

1. El sistema muestra las tendencias de todas las métricas.

2.6.4 Diagramas de Secuencia

A continuación se muestran los diagramas de secuencia para los casos de usos definidos anteriormente.

Caso de Uso Gestionar Métricas

Escenario Insertar Métricas

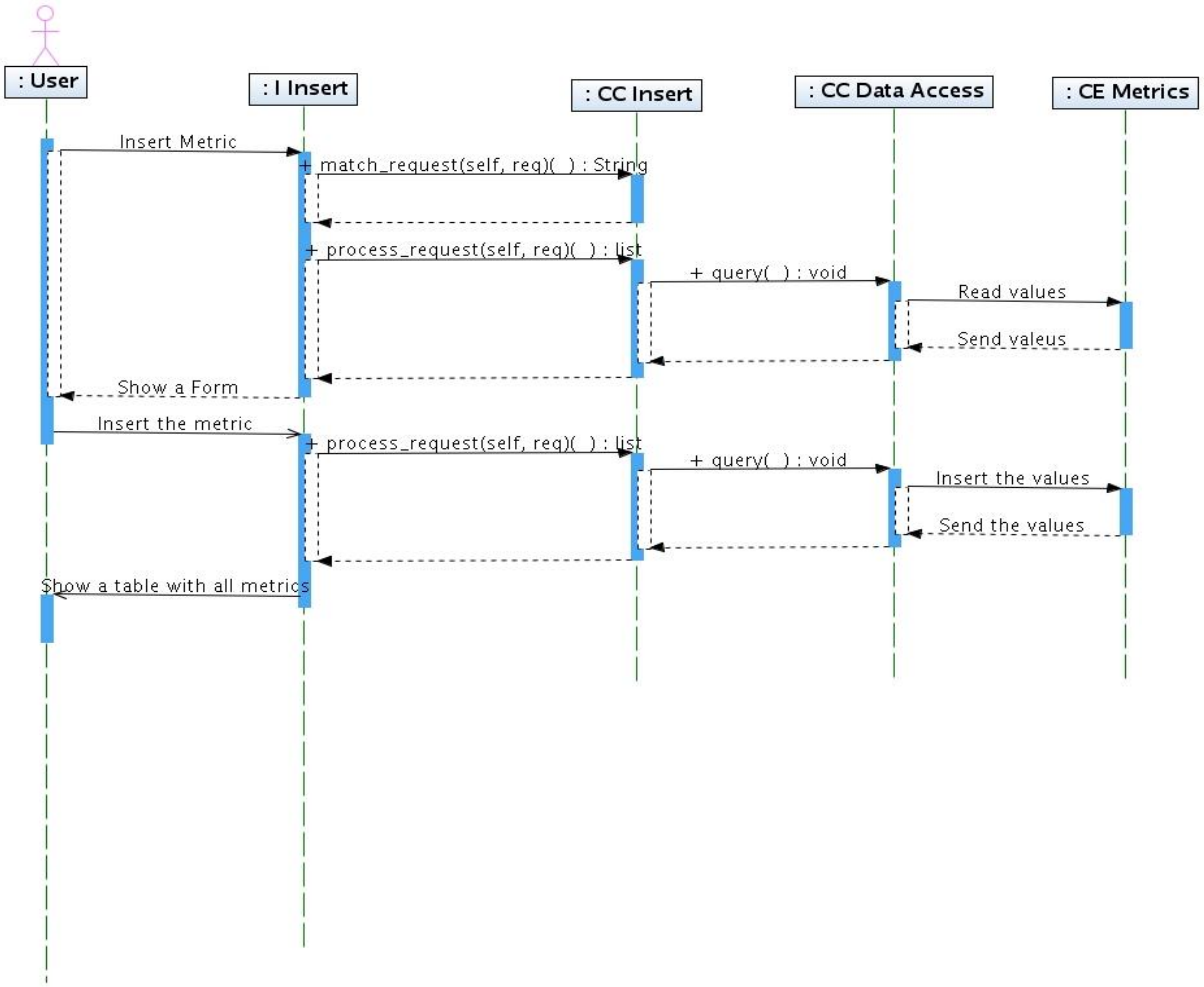


Figura 5 Diagrama de secuencia Insertar Métricas.

Escenario Modificar Métrica

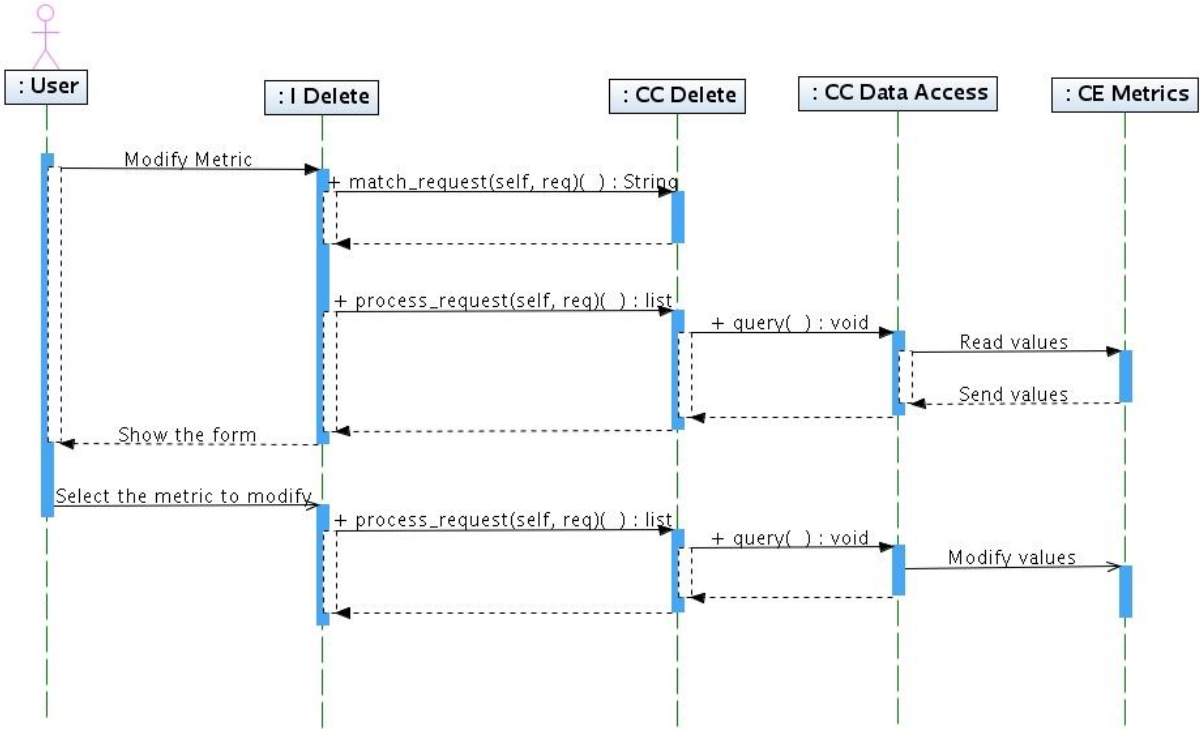


Figura 6 Diagrama de secuencia Modificar Métricas

Escenario Eliminar Métrica

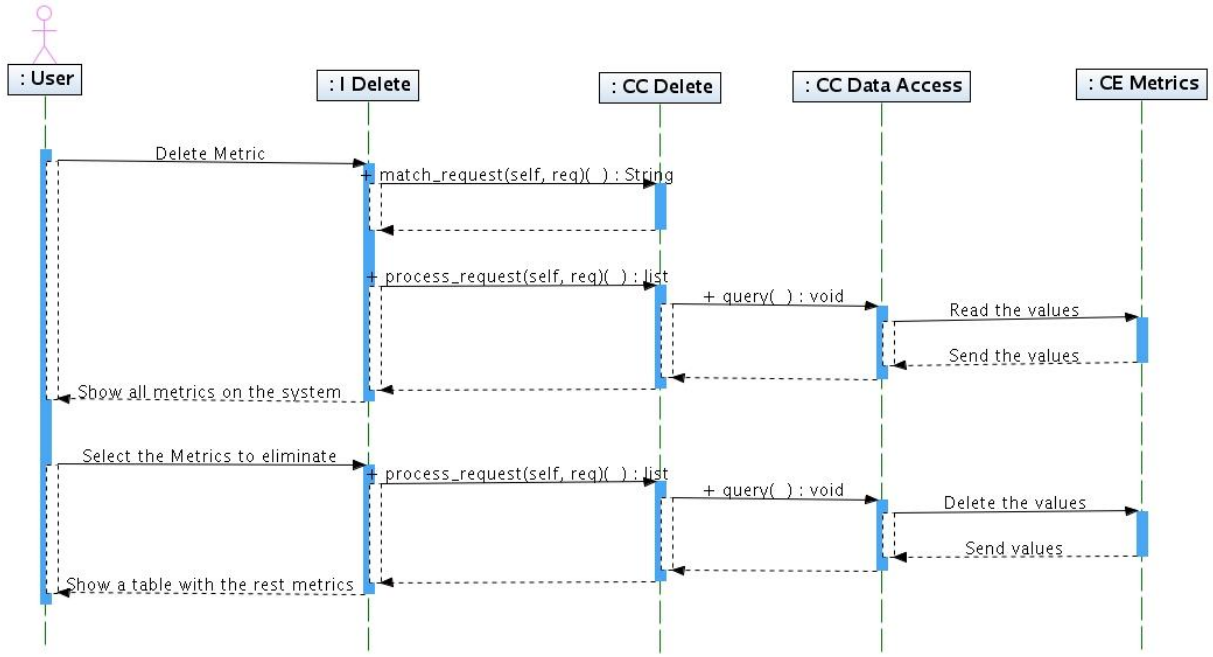


Figura 7 Diagrama de secuencia Eliminar Métricas

Caso de Uso Registrar Valores

Escenario Guardar Medición

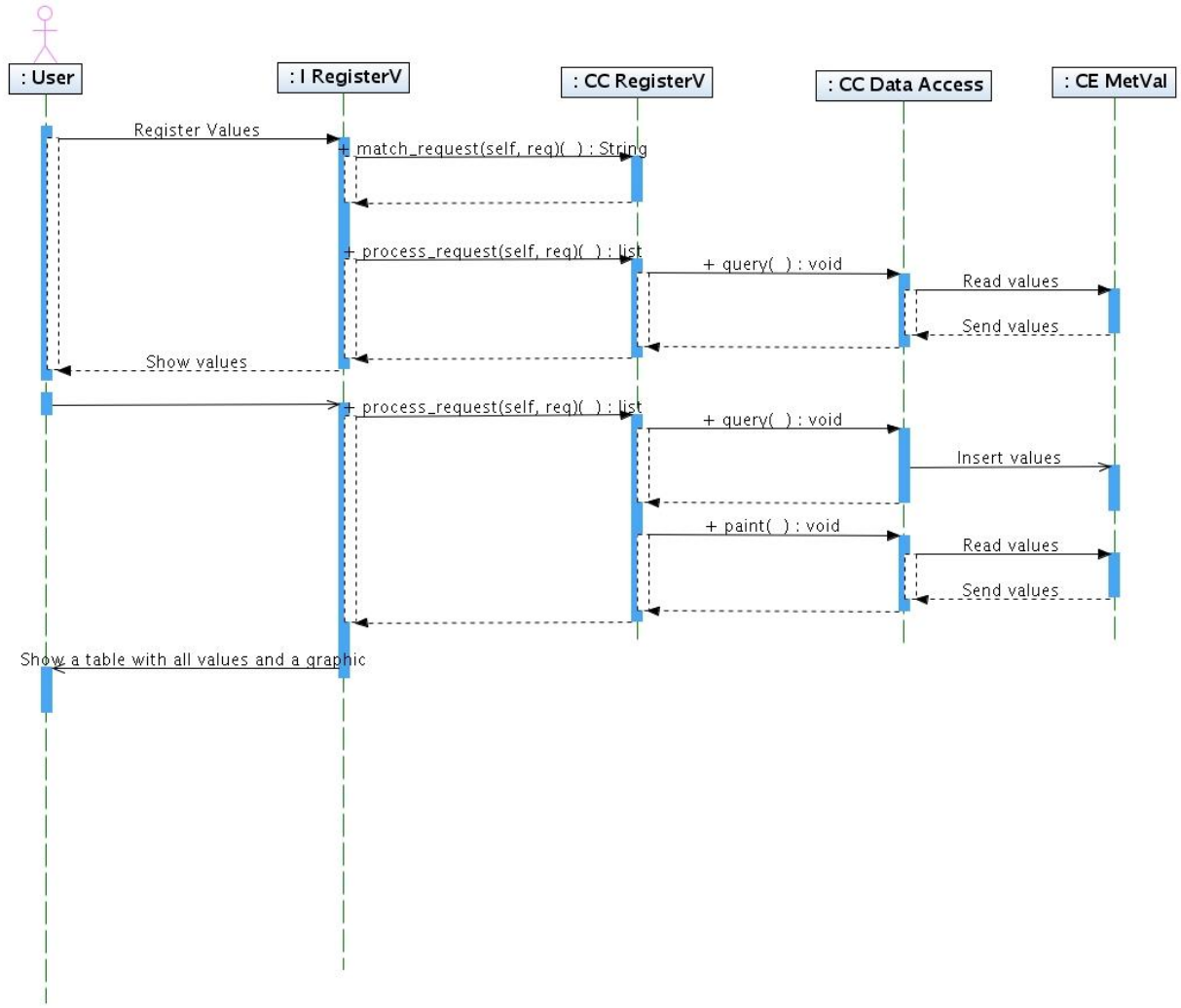


Figura 8 Diagrama de secuencia Guardar Medición

Escenario Guardar Métrica

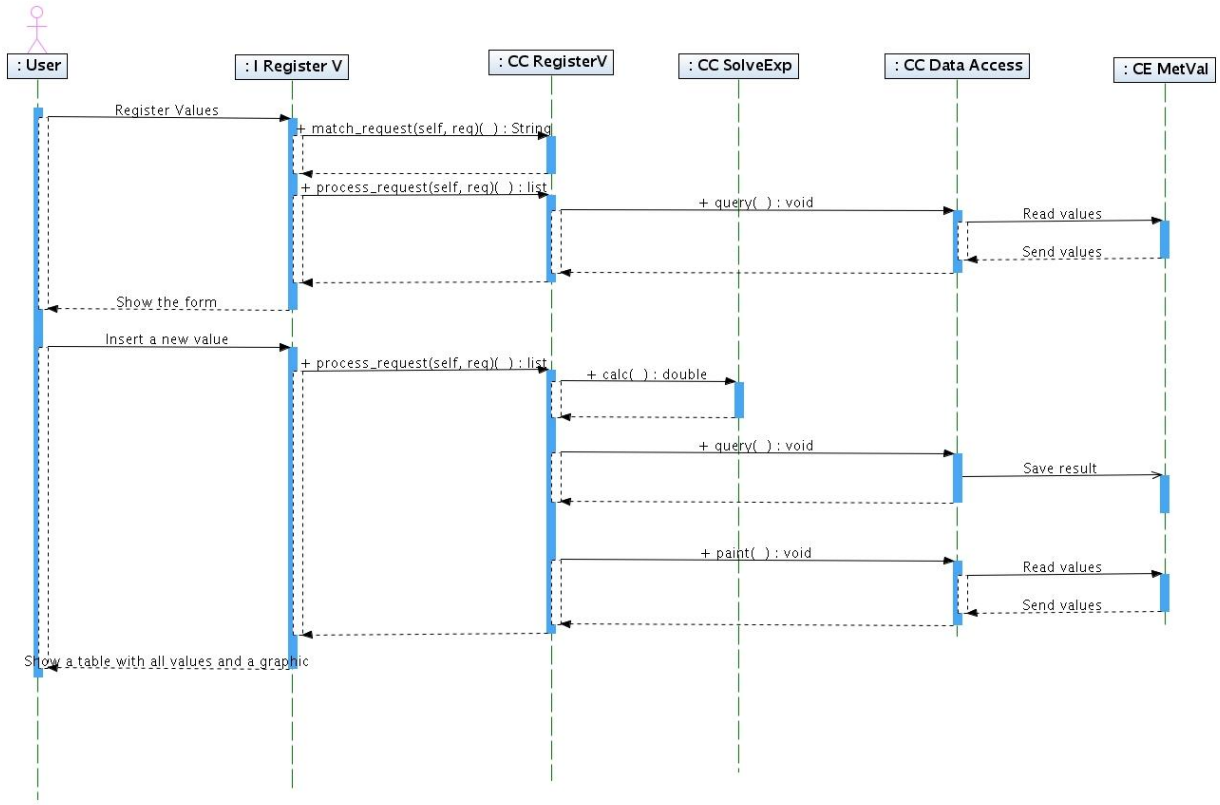


Figura 9 Diagrama de secuencia Guardar Métricas

Caso de Uso Mostrar Reporte

Escenario Mostrar Tendencia

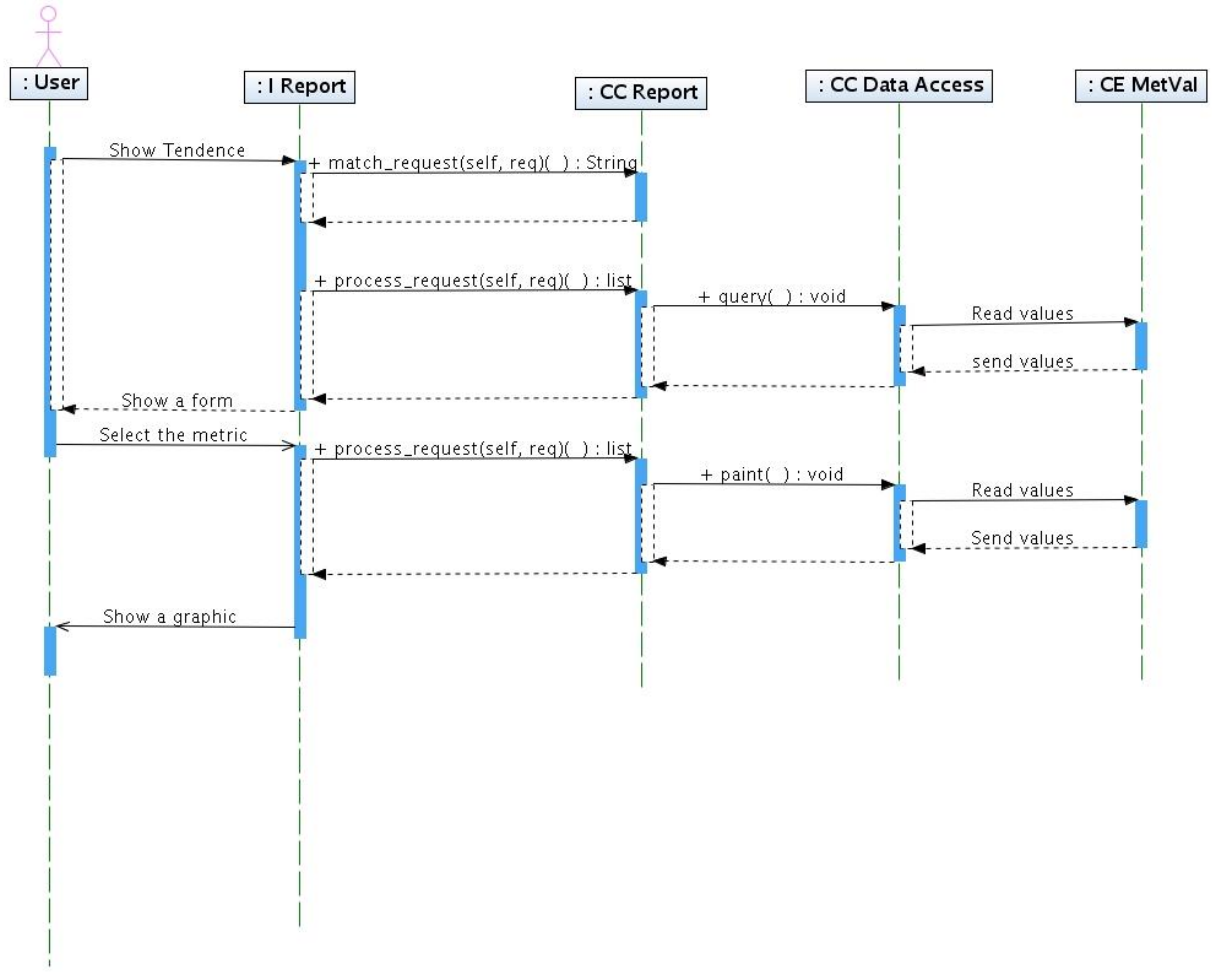


Figura 10 Diagrama de secuencia Mostrar Tendencia

Escenario Mostrar Reporte General

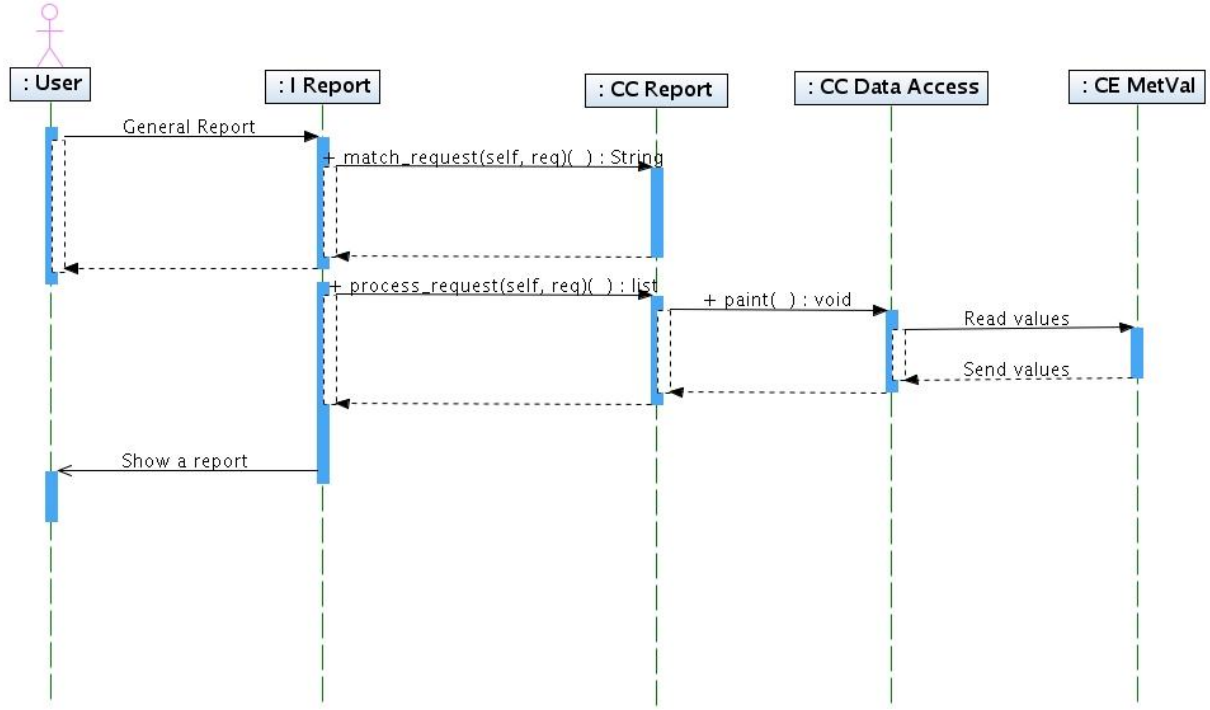


Figura 11 Diagrama de secuencia Mostrar Reporte General

2.7 Modelo de Diseño e Implementación

2.7.1 Patrones de Diseño

Patrón Singleton

Dentro de los patrones de diseño utilizados se encuentra el Patrón Singleton por su capacidad de proveer un mecanismo que limita el número de instancias de una clase y un único punto de acceso a la misma.

Es uno de los patrones más usados y sencillos que existen, garantizando que solo halla una instancia de una determinada clase y que se pueda tener total acceso ella, teniendo control estricto acerca de cómo y cuándo los clientes la acceden.

Patrón Polimorfismo

Se decidió la utilización de este patrón pues se tienen un grupo de clases que tienen los mismos métodos, y atributos, las cuales se podrían agrupar en una clase padre, y las hijas heredarían todos los métodos y atributos, solo que reimplementarían las funcionalidades que ellas en si necesiten. El uso del patrón traerá consigo numerosas ventajas. Esto se puede apreciar en la relación que existe entre la clase ManagePage esta clase cuenta con los siguientes que es la padre y sus hijas que son InsertPage, ModifyPage, DeletePage, ReportPage, y MetricsPage.

2.7.2 Diagrama de clases

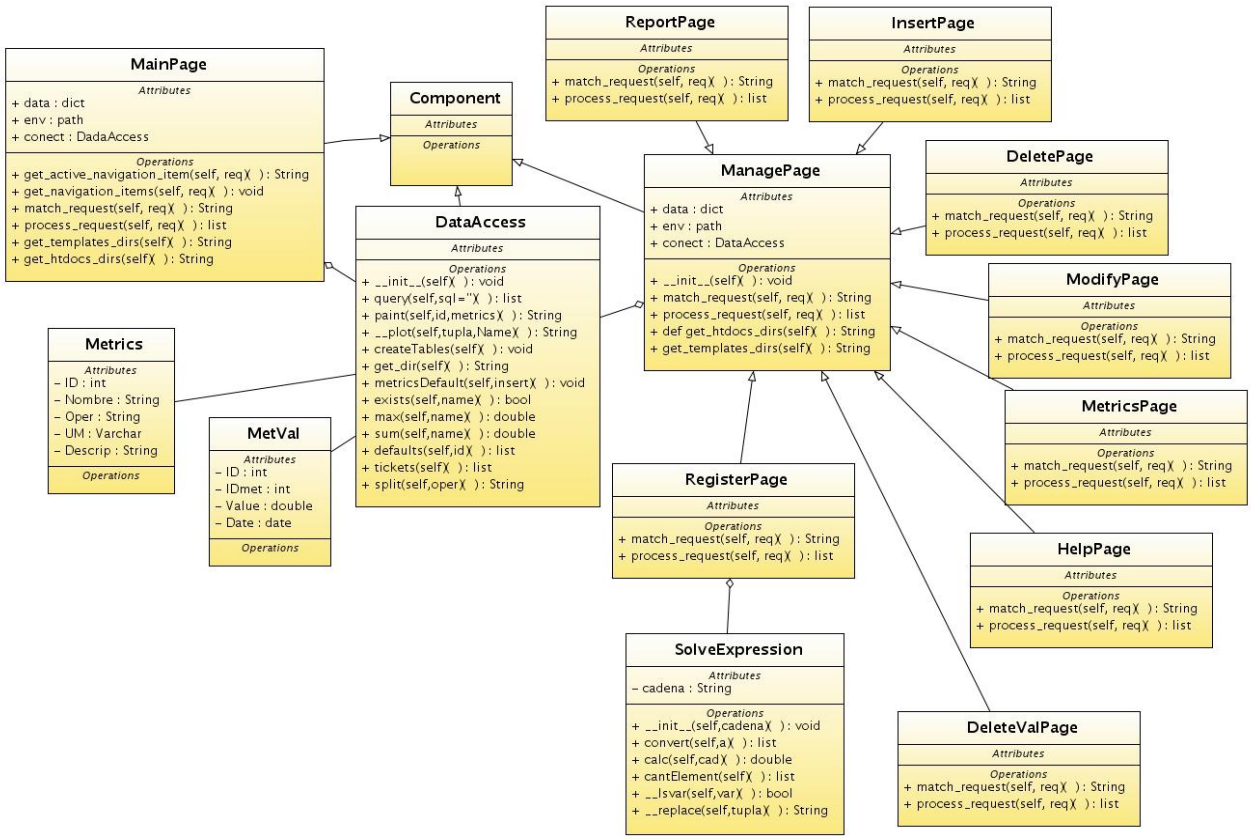


Figura 12 Diagrama de Clases del Diseño

A continuación se realizará una descripción de las clases fundamentales presentes en el diagrama.

Tabla # 1 Descripción de la clase MainPage.

Nombre: MainPage	
Descripción: Es la clase que se encarga asegurar el funcionamiento del extensión así como la creación de las tablas donde se hará persistente la información.	
Tipo de clase: Interfaz	
Atributo	**Tipo**

data	dict
env	path
conect	DataAccess
Para cada responsabilidad:	
Nombre:	match_request(self, req): String
Descripción:	Método encargado de capturar la petición del usuario y retornar la dirección donde se encuentra dicha petición
Nombre:	process_request(self, req)( ) : list
Descripción:	Método encargado de procesar la petición y retornar template_name, datos(diccionario), content_type
Nombre:	get_templates_dirs(self)( ) : String
Descripción:	Devuelve la dirección donde se encuentran los templates
Nombre:	get_navigation_items(self, req):String
Descripción:	Método que se encarga de crear el botón de la navegación del extensión
Nombre:	get_active_navigation_item(self, req):String
Descripción:	Método que devuelve la dirección de la página actual
Nombre:	get_htdocs_dirs(self):String
Descripción:	Método encargado de devolver la dirección de los CSS y de las imágenes.

Tabla # 2 Descripción de la clase ManagePage.

Nombre: ManagePage	
Descripción: Es la clase encargada administración de métricas además de implementar los métodos más importantes y que serán heredados por las clases hijas.	
Tipo de clase: interfaz	
Atributo	**Tipo**
data	dict
env	path
conect	DataAccess

Para cada responsabilidad:	
Nombre:	match_request(self, req): String
Descripción:	Método encargado de capturar la petición de modificar información y retornar la dirección donde se encuentra el template modificar información
Nombre:	process_request(self, req)( ) : String
Descripción:	Método encargado de procesar la petición de modificación de los campos de la información.
Nombre:	__init__(self):void
Descripción:	Constructor de la clase, no hace nada.
Nombre:	get_htdocs_dirs(self):String
Descripción:	Método encargado de devolver la dirección de los CSS y de las imágenes.
Nombre:	get_templates_dirs(self)( ) : String
Descripción:	Devuelve la dirección donde se encuentran los templates.

Tabla # 3 Descripción de la clase InsertPage.

Nombre: InsertPage	
Descripción: Es la clase encargada de realizar las acciones que permitan insertar una nueva métrica en el sistema.	
Tipo de clase: Interfaz	
Atributo	**Tipo**
Para cada responsabilidad:	
Nombre:	match_request(self, req): String
Descripción:	Método encargado de capturar la petición de modificar información y retornar la dirección donde se encuentra el template modificar información
Nombre:	process_request(self, req)( ) : String
Descripción:	Método encargado de procesar la petición de modificación de los campos de la información.

Tabla # 4 Descripción de la clase ModifyPage.

Nombre: ModifyPage	
Descripción: Es la clase encargada de realizar las acciones que permitan modificar los campos tanto a una medición como a una métrica en el sistema.	
Tipo de clase: Interfaz	
Atributo	**Tipo**
Para cada responsabilidad:	
Nombre:	match_request(self, req): String
Descripción:	Método encargado de capturar la petición de modificar información y retornar la dirección donde se encuentra el template modificar información
Nombre:	process_request(self, req)( ) : String
Descripción:	Método encargado de procesar la petición de modificación de los campos de la información.

Tabla # 5 Descripción de la clase DeletePage.

Nombre: DeletePage	
Descripción: Es la clase encargada de realizar las acciones que permitan eliminar una o varias métricas del sistema.	
Tipo de clase: Interfaz	
Atributo	**Tipo**
Para cada responsabilidad:	
Nombre:	match_request(self, req): String

Descripción:	Método encargado de capturar la petición de modificar información y retornar la dirección donde se encuentra el template modificar información
Nombre:	process_request(self, req)( ) : String
Descripción:	Método encargado de procesar la petición de modificación de los campos de la información

Tabla # 6 Descripción de la clase MetricsPage.

Nombre: MetricsPage	
Descripción: Es la clase encargada de realizar las acciones que permitan registrar valores tanto a una medición como a una métrica en el sistema.	
Tipo de clase: Interfaz	
Atributo	**Tipo**
Para cada responsabilidad:	
Nombre:	match_request(self, req): String
Descripción:	Método encargado de capturar la petición de modificar información y retornar la dirección donde se encuentra el template modificar información
Nombre:	process_request(self, req)( ) : String
Descripción:	Método encargado de procesar la petición de modificación de los campos de la información

Tabla # 7 Descripción de la clase DeletePage.

Nombre: ReportPage	
Descripción: Es la clase encargada de realizar las acciones que permitan visualizar las tendencias tanto de una métrica o de todas las métricas en el sistema.	
Tipo de clase: Interfaz	
Atributo	**Tipo**

Para cada responsabilidad:	
Nombre:	match_request(self, req): String
Descripción:	Método encargado de capturar la petición de modificar información y retornar la dirección donde se encuentra el template modificar información
Nombre:	process_request(self, req)( ) : String
Descripción:	Método encargado de procesar la petición de modificación de los campos de la información

Tabla # 8 Descripción de la clase DataAccess.

Nombre: DataAccess	
Descripción: Es la clase encargada de la hacer la conexiones y consultas a la base de datos.	
Tipo de clase: control	
Atributo	**Tipo**
env	path
Para cada responsabilidad:	
Nombre:	__init__(self, cadena): void
Descripción:	Constructor de la clase
Nombre:	Query(self, req)( ) :list
Descripción	Método encargado de realizar todas las consultas a la Base de Datos SQLite
Nombre:	CreateTables(self)( ) : void
Descripción:	Método encargado de crear las tablas en la Base de Datos
Nombre:	paint(self, id, metrics):String
Descripción:	Método que se encarga de escoger según el id las métricas que se van a graficar a través del método __plot
Nombre:	__plot(self, tupla, Name): String
Descripción:	Método que se encarga de pintar el gráfico según valores seleccionados y guarda la imagen de la figura con el nombre pasado por parámetros por defecto es “Tendence.png”.

Nombre:	get_dir(self): String
Descripción:	Método que se encarga de devolver la dirección donde se guardaran las imágenes
Nombre:	MetricsDefault(*self*, insert=False): void
Descripción:	Método que se encarga de insertar las métricas por defecto, el paquete de métricas seleccionados
Nombre:	exists(*self*, name): bool
Descripción:	Método devuelve verdadero en caso de existir una métrica con el nombre pasado por parámetro
Nombre:	max(*self*, name):float
Descripción:	Método encargado de devolver el máximo valor entre lo valores registrados de la métrica pasada por parámetros
Nombre:	sum(*self*, name):float
Descripción:	Método encargado de devolver la suma de todos los valores registrados de la métrica pasada por parámetros
Nombre:	defaults(*self*, id= 7): void
Descripción:	Método para calcular las métricas por defecto
Nombre:	tickets(self) : list
Descripción:	Método encargado de calcular el por ciento de tareas sin terminar a través de los tickets
Nombre:	OnlyInsert(*self*, valor, id)
Descripción:	Método para insertar las métricas que requieren de un solo valor, no hay más de un valor de medición del mismo.
Nombre:	split(*self*, oper)
Descripción:	Método encargado de sustituir en la expresión las sumas y los máximos por el valor de los mismo una vez realizada la consulta.

Tabla # 9 Descripción de la clase SolveExpression.

Nombre: SolveExpression

Descripción: Es la clase encargada resolver la ecuación dada una métrica.	
Tipo de clase: control	
Atributo	**Tipo**
cadena	String
Para cada responsabilidad:	
Nombre:	__init__(self, cadena):void
Descripción:	Constructor de la clase.
Nombre:	convert(self, a):list
Descripción:	Método que devuelve una lista con los elementos de la cadena.
Nombre:	calc(self, cadena):int
Descripción:	Método encargado de realizar el cálculo de la métrica específica
Nombre:	cantElement(self):int
Descripción:	Método que devuelve la cantidad de elementos de la cadena.
Nombre:	__Isvar(self, var):bool
Descripción:	Método que devuelve si un carácter es variable o no.
Nombre:	__replace(self, tupla):list
Descripción:	Método que remplaza las variables por números para realizar el cálculo correspondiente.

Tabla # 10 Descripción de la clase Metric.

Nombre: Metric	
Descripción: Es la clase encargada de la persistencia de las métricas	
Tipo de clase: Entidad	
Atributo	**Tipo**
ID	int
Name	String
Oper	String
UM	varchar
Descrip	String

Para cada responsabilidad:	
Descripción:	
Nombre:	

Tabla # 11 Descripción de la clase MetVal.

Nombre: MetVal	
Descripción: Es la clase encargada de la persistencia de las métricas	
Tipo de clase: Entidad	
Atributo	**Tipo**
ID	int
Idmet	int
Value	double
Date	date
Para cada responsabilidad:	
Descripción:	
Nombre:	

2.7.3 Diagrama de Despliegue

Un diagrama de despliegue muestra las relaciones físicas entre los componentes de hardware y de software que existen en un determinado sistema.

La herramienta de Gestión de Proyecto Trac, estará instalada en el Servidor Apache utilizando como sistema Gestor de Base de Datos SQlite. Todas las PC clientes podrán acceder a dicho servidor mediante el protocolo de comunicación <HTTP>, garantizando que en cada estación de trabajo los usuarios tengan acceso a la herramienta.

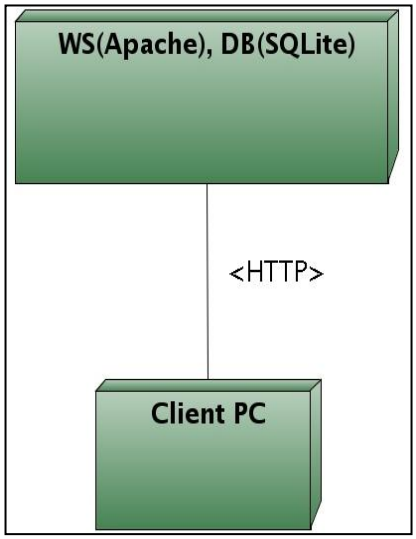


Figura 13 Diagrama de Despliegue

De manera general en este capítulo se realizó el diseño y la implementación de la aplicación. Para el correcto diseño e implementación de aplicación se hizo un estudio detallado de la arquitectura base del trac con el fin de entender su funcionamiento y lograr acoplar la solución dentro del de manera correcta, además se hizo uso de patrones de diseño. Durante el desarrollo de este capítulo se desarrollaron los artefactos correspondientes a la metodología de desarrollo empleada. Se recomienda mejorar el diseño e implementación de la aplicación pues esta es solo la primera versión. Se espera que sea de ayuda para lograr un mayor control y seguimiento de los proyectos de software pues tanto la propuesta del paquete de métricas como la utilización de la aplicación brindará numerosas ventajas, podría contribuir a mejorar algunos de los aspectos que más afectan la producción de software dentro del polo de Hardware y Automática y de la universidad en general.

Capítulo #3 Pruebas y seguimiento de la solución planteada

Introducción

En el presente capítulo se realizarán pruebas con el objetivo de comprobar el correcto funcionamiento de la aplicación, para lo cual la aplicación deberá estar instalada en un servidor, local, luego si se concluye que la aplicación está lista para ser liberada, se procederá a instalarse en varios de los proyectos del polo con el fin de comenzar a registrar los datos históricos de las mediciones que se realicen en los mismos.

3.1 Diseño de clases de pruebas

A continuación se muestra el diseño de casos de pruebas elaborados para la aplicación.

Caso de Uso Gestionar Métricas:

Descripción general

Este caso de uso permite al usuario insertar una nueva métrica en el sistema, modificar una existente o eliminar alguna o todas las que desee.

Las pruebas a realizar a este caso de uso son:

1 Insertar Métrica.

2 Modificar Métrica.

3 Eliminar Métrica.

3.2. CP 1 Insertar Métrica

3.2.1 Descripción

Este caso de prueba le permite al usuario insertar una nueva métrica.

3.2.2 Flujo central.

1. El usuario abre la aplicación.
2. El sistema carga la página principal del Trac.
3. El usuario selecciona el botón “Metrics”
4. El sistema carga la página principal de la extensión.
5. El usuario selecciona la opción Gestionar Métricas.
6. El sistema muestra una interfaz con tres vínculos.
7. El usuario selecciona la opción Insertar Métricas.
8. El sistema muestra una interfaz con los campos necesarios para insertar una métrica, los cuales son Nombre de la métrica, Expresión, Unidad de medida y Descripción.
9. El usuario debe introducir los datos necesarios para insertar correctamente una métrica.
10. El sistema valida la completitud de los datos introducidos.
11. El sistema guarda los valores introducidos y permite seguir introduciendo nuevas métricas.

3.2.3 Condiciones de ejecución.

- El usuario debe estar autenticado en el sistema.

3.2.4 Iteraciones

Tabla # 12 Especificación del caso de prueba Insertar Métrica.

Clases Válidas	Clases Inválidas	Resultado Esperado	Resultado de la Prueba	Observaciones
Introduce todos los campos correctamente, Ejemplos Nombre: LOC, Expresión: LOC/HPD, Unidad de medida #, Descripción: métricas adad.		El sistema introduce la nueva métrica, permite seguir insertando nuevas, y se muestra una tabla con un listado actual de las métricas en el sistema.		
	El usuario deja campos vacios Ejemplo: ,Expresión: A, Unidad de medida: %, Descripción: Métrica 1	El sistema muestra un mensaje de error alertando que existen campos vacíos.		
	El usuario introduce valores	El sistema muestra un mensaje alertando de que la unidad de		

	incorrectos. Ejemplos: unidad de medida: asd	medida es incorrecta.		

3.3 CP 2 Modificar Métrica

3.3.1 Descripción

Este caso de prueba le permite al usuario modificar una métrica existente en el sistema.

3.3.2 Flujo central.

1. El usuario abre la aplicación.
2. El sistema carga la página principal del Trac.
3. El usuario selecciona el botón “Metrics”
4. El sistema carga la página principal de la extensión.
5. El usuario selecciona la opción “Manage Metrics”.
6. El sistema muestra una interfaz con tres vínculos.
7. El usuario selecciona la opción “Modify Metrics”.
8. El sistema muestra una interfaz con un listado de todas las métricas a modificar.
9. El usuario selecciona la métrica que desea modificar.
10. El sistema muestra una interfaz donde se cargan los datos de dicha métrica.
11. El usuario modifica actualiza los campos deseados.
12. El sistema valida la completitud de los datos insertados por el usuario.
13. El sistema actualiza la métrica con los nuevos valores.

3. 3.3 Condiciones de ejecución.

- El usuario debe estar autenticado en el sistema.
- Deben existir métricas en el sistema.

3.3.4 Iteraciones

Tabla # 13 Especificación del caso de prueba Modificar Métrica.

Clases Válidas	Clases Inválidas	Resultado Esperado	Resultado de la Prueba	Observaciones
El usuario introduce los campos que desea modificar, el resto permanece como estaba. Ejemplos: Nombre: ABC, Expresión: A+A, Unidad de medida: #, Descripción: esta métrica es para asdasd.		El sistema actualiza la métrica con los nuevos valores.		
	El usuario modifica los campos deseados, pero deja campos vacios Ejemplo Nombre: AA, Expresión , Unidad de Medida: $,	El sistema muestra un mensaje de error, alertando de que no pueden existir campos vacíos.		

	Descripción: adasdasdas			
	El usuario modifica los campos deseados, pero comete errores, Ejemplos Nombre: 1AS, Expresión AS, Unidad de Medida: #, Descripción: adasdasdas	El sistema muestra un mensaje de error alertando de que existe un error. Y el campo nombre es inválido.		

3.4. CP 3 Eliminar Métrica

3.4.1 Descripción

Este caso de prueba le permite al usuario eliminar una o varias métricas del sistema.

3.4.2 Flujo central.

1. El usuario accede a la aplicación.
2. El sistema carga la página principal del Trac.
3. El usuario selecciona el botón “Metrics”
4. El sistema carga la página principal de la extensión.
5. El usuario selecciona la opción “Manage Metric”.
6. El sistema muestra una interfaz con tres vínculos.
7. El usuario selecciona la opción “Delete Metric”

8. El sistema muestra una interfaz con un listado de todas las métricas existentes en el sistema y un "checkbox" correspondiente a cada una de las métricas.
9. El usuario marca las o las métricas que desea eliminar.
10. El sistema elimina las métricas del sistema.

3.4.3 Condiciones de ejecución

El usuario debe estar autenticado en el sistema.

Deben existir métricas en el sistema.

3.4.4 Iteraciones

Tabla # 14 Especificación del caso de prueba Eliminar Métrica.

Clases Válidas	Clases Inválidas	Resultado Esperado	Resultado de la Prueba	Observaciones
El usuario marca una o varias métricas		El sistema elimina la métrica del sistema.		
	El usuario no marca ninguna métrica.	El sistema no realiza ninguna acción.		

Caso de Uso Gestionar Métricas

Descripción general

Este caso de uso permite al usuario registrar valores tanto a una métrica o una medición existente en el sistema.

Las pruebas a realizar a este caso de uso son:

1. Registrar valor a una medición.
2. Registrar valor a una métrica.

3.5. CP 4 Registrar valores

3.5.1 Descripción

Este caso de prueba le permite al usuario insertar valores tanto a una métrica como a una medición existente en el sistema.

3.5.2 Flujo central.

El usuario accede a la aplicación.

1. El sistema carga la página principal del Trac.
2. El usuario selecciona el botón “Metrics”
3. El sistema carga la página principal de la extensión.
4. El usuario escoge la opción “Register Values”
5. El sistema muestra una interfaz con un listado de todas las métricas.
6. El usuario escoge una métrica para insertarle valores.
7. El sistema muestra un interfaz con los campos necesarios para introducir los valores.
8. El sistema verifica la completitud de los datos.
9. El sistema realiza el cálculo correspondiente.
10. El sistema guarda el resultado del cálculo.

3.5.3 Condiciones de ejecución

- El usuario debe estar autenticado en el sistema.
- Deben existir métricas en el sistema.

3.5.4 Iteraciones

Tabla # 15 Especificación del caso de prueba Registrar Valores.

Clases Válidas	Clases Inválidas	Resultado Esperado	Resultado de la Prueba	Observaciones
El usuario inserta todos los valores requeridos por la métrica Ejemplo: 5		El sistema introduce el valor muestra un listado con las mediciones anteriores para esa métrica, además de un gráfico donde se puede observar la tendencia de dicha métrica en el tiempo		
	El usuario introduce letras en vez de números	El sistema muestra un mensaje de error alertando que el campo solo lleva números.		
	El usuario deja campos vacíos	El sistema muestra un mensaje de error alertando que debe llenar los campos para poder realizar la acción.		

Caso de Uso Mostrar Reporte:

Descripción general

Este caso de uso permite al usuario ver la tendencia de una métrica o un reporte general de todas las métricas.

Las pruebas a realizar a este caso de uso son:

1. Tendencia.
2. Reporte General.

3.6. CP 5 Tendencia

3.6.1 Descripción

Este caso de prueba le permite al usuario ver los gráficos de tendencia una métrica o medición existente en el sistema.

3.6.2 Flujo central.

1. El usuario accede a la aplicación.
2. El sistema carga la página principal del Trac.
3. El usuario selecciona el botón “Metrics”.
4. El sistema carga la página principal de la extensión.
5. El usuario selecciona la opción “Reports”
6. El sistema muestra una interfaz con un “combobox”, en el cual aparece predefinido la opción “Show all”, seguido de las métricas existentes en el sistema.
7. El usuario selecciona la métrica a la cual desea tener su reporte, el cual consiste en un gráfico con la tendencia de ella en el tiempo.

8. El sistema muestra el gráfico o el valor en caso de que la métrica tenga un solo valor.

3.6.3 Condiciones de ejecución

- El usuario debe estar autenticado en el sistema.
- Deben existir métricas en el sistema.

3.6.4 Iteraciones

Tabla # 16 Especificación del caso de prueba Tendencia.

Clases Válidas	Clases Inválidas	Resultado Esperado	Resultado de la Prueba	Observaciones
El usuario selecciona una de las métricas existentes en el sistema		El sistema grafica la tendencia de la métrica en el tiempo.		

3.7. CP 6 Reporte General

3.7.1 Descripción

Este caso de prueba le permite al usuario ver un reporte general de todas las métricas y mediciones existentes en el sistema.

3.7.2 Flujo central.

1. El usuario accede a la aplicación.
2. El sistema carga la página principal del Trac.
3. El usuario selecciona el botón “Metrics”.

8. El sistema carga la página principal de la extensión.
9. El usuario selecciona la opción “Reports”
10. El sistema muestra una interfaz con un “combox”, en el cual aparece predefinido la opción” Show all”, seguido de las métricas existentes en el sistema.
11. El usuario selecciona la opción predefinida “Show All”.
12. El sistema muestra los gráficos o los valores en caso de que alguna métrica tenga un solo valor.

3.7.3 Condiciones de ejecución

- El usuario debe estar autenticado en el sistema.
- Deben existir métricas en el sistema.

3.7.4 Iteraciones

Tabla # 17 Especificación del caso de prueba Reporte General.

Clases Válidas	Clases Inválidas	Resultado Esperado	Resultado de la Prueba	Observaciones
El usuario selecciona la opción “Show All”		El sistema muestra los gráficos o los valores de las métricas existentes en el sistema.		

3.2 Manual de Usuario basado en los prototipos funcionales.

Para poder tener en la aplicación es necesario tener instalado un Trac en un servidor, este servidor puede estar en cualquier locación física, solamente se necesita permiso de administración para instalar el extensión, posteriormente para su uso y mantenimiento no se necesitan estos permisos. Cuando es instalado el extensión se activa un nuevo botón en la barra de navegación del trac llamado Metrics, para el uso de la aplicación es necesario dar clic en el botón luego automáticamente se carga en la interfaz principal de la aplicación. Esta interfaz cuenta con cuatro vínculos fundamentales los cuales proporcionan las funcionalidades principales. Los vínculos antes mencionados son Manage Metrics, Register Values, Reports, Help in Spanish.

1. *Manage Metrics*

Cuando se accede a este vínculo muestra una interfaz con tres nuevos vínculos los cuales son *Insert Metrics*, *Delete Metrics* y *Modify Metrics*.

1.1 *Insert Metrics*: Esta funcionalidad de la aplicación permite al usuario insertar una nueva métrica, cuando se da clic en ese vínculo se accede a una interfaz con todos los campos necesarios para insertar una nueva métrica. Los campos son *Name*, *Expression*, *Unit of Measure* y *Description* de la métrica a insertar. En el campo *Name* está referido al nombre por el cual responde la métrica, este campo no puede contener números, *Expression* se refiere a la ecuación matemática la cual define la métrica, para este campo se definieron palabras reservadas *max(),* y *sum(),* las cuales tiene la funcionalidad de representar el máximo de una valor y la sumatoria de un valor respectivamente. Además contiene un campo *Unit of Measure* en el cual el usuario debe escoger en que se expresa la métrica en el caso de que no exista se debe seleccionar U, referido a unidades. Por último con tiene un campo *Description* en el cual se debe hacer referencia a una breve descripción de la métrica, para que se utiliza, o el significa desde el punto de vista del usuario.

1.2 *Delete Metrics*: Esta funcionalidad permite al usuario eliminar una métrica existente en el sistema, el sistema muestra una tabla con el listado de las métricas existentes en el sistema y un *checkbox* correspondiente para cada métrica, esto permite seleccionar una o varias métricas cuando se desee eliminar del sistema.

1.3 *Modify Metrics*: esta funcionalidad le permite al usuario modificar cualquier métrica existente en el sistema, para ello el sistema muestra un listado con cada una de las métricas y un vínculo, cuando se accede a este vínculo el sistema carga todos los campos de la métrica seleccionada, posibilitando al usuario modificar los campos que desee.

2. *Register Values*

Esta funcionalidad le permite al usuario realizar mediciones, guardando los valores correspondientes a cada una de las métricas, ya sea una medición o una métrica en si. Al acceder al vínculo *Register Values* el sistema muestra un listado con cada una de las métricas y un vínculo para insertar valor a cada una de las métricas. En caso de que se seleccione una medición el sistema cargará una interfaz permitiendo al usuario insertar el valor de la medición junto a la fecha actual. Si la selección es una métrica el sistema muestra una interfaz con tantos valores como requiera la métrica, además de realizar el cálculo internamente y se almacenará el resultado del cálculo.

3. *Report*

Esta funcionalidad brinda la posibilidad de visualizar los valores de las mediciones que se realizan dentro del proyecto. Al acceder al vínculo *Report*, el sistema muestra una interfaz brindando la posibilidad de escoger una métrica en específico, o la opción *show all,* mostrando con esta ultima, un reporte general de todas las métricas existentes en el sistema.

4. Help in Spanish

Esta es una sección donde el usuario puede apoyarse para el trabajo con la aplicación. Es recomendable su utilización para poder hacer un uso correcto de la aplicación.

3.3 Resultados presentes y visión futura

La extensión realizada como resultado final de la investigación que tenia como objetivo facilitar el proceso de salva de los datos históricos de las mediciones que se realizan a los proyectos del Polo de Hardware y Automática brinda varias ventajas como son, el fácil manejo y mantenimiento de la información además de una fácil visualización de la información, además muestra gráficamente los valores para cada una de las métricas en el tiempo, esto permitirá al líder del proyecto en el cual este instalada la extensión hacer lo mismo estimaciones de tiempo costo tamaño entre otras, así como tenerlo en cuenta para futuras toma de decisiones. La aplicación fue desplegada en varios proyectos del polo entre los que se encuentran despGalba, Retriver, entre otros. En todos los casos los líderes de proyecto muestran un positivo grado de aceptación con respecto a la aplicación debido a su fácil uso y las funcionalidades que esta brinda, además de automatizar el proceso de medición y salva de los datos históricos de las mediciones que se realicen en cada uno de los proyectos.

3.4 Pruebas

La extensión implementada se sometió a pruebas de caja negra, estas pruebas fueron realizadas por el equipo de calidad de la facultad. Las pruebas arrojaron un total de 17 no conformidades (ver anexo 2), para corregir estas no conformidades se realizaron un conjunto de acciones correctivas (ver anexo 3).

En el presente capítulo se diseñaron los casos de prueba para cada uno de los escenarios básicos de la aplicación. Además se realizó un manual de usuario con el objetivo de explicar de forma muy sencilla el manejo básico de la aplicación y su claro entendimiento. También se realizó una entrevista a los líderes de proyecto en los cuales fue desplegada la aplicación con el objetivo de saber el grado de satisfacción así como la utilidad de la aplicación.

Conclusiones

Al llegar al término del trabajo de diploma se puede concluir que:

- Del estudio realizado sobre las métricas mas utilizadas en los proyectos de software se obtuvieron varias variables a evaluar para seleccionar las métricas adecuadas según las necesidades del Polo de Hardware y Automática
- Se elaboró un paquete de métricas para comenzar a realizar mediciones dentro de los proyectos del Polo.
- Se implementó una extensión en el cual están contenidas dichas métricas para comenzar a registrar los datos de las mediciones que se realicen.
- Se realizaron pruebas a la extensión con el objetivo de verificar que cumple con todos los requisitos planteados.

Recomendaciones

Se recomienda:

1. Instalar la extensión en cada uno de los proyectos del Polo de Hardware y Automática para comenzar a registrar los datos históricos de cada proyecto y en todos los proyectos de la universidad en los cuales se realice la gestión de proyecto con el Trac.
2. Adicionar nuevas funcionalidades a la extensión de manera que se logre una versión más completa para la gestión de las métricas en los proyectos de producción de software en la Universidad.
3. Exponer la presente investigación en eventos científicos o relacionados con el tema.
4. Instalar la extensión TracMetricsPlugins, debido a las ventajas y funcionalidades que este brinda.

Referencias Bibliográficas

[INFORMACIÓN, C. T. Y. D. N. N. C. D. T. D. L]. 2005. *Norma Cubana, ISO/IEC 9126-1:2001, Ingeniería de Software—Calidad del Producto—, Oficina Nacional de Normalización.* 2005.

A, NAVARRO. 2005. Ingeniería de Software. [Online] 2005. http://www.fdi.ucm.es/profesor/anavarro/8._Planificacion_temporal_y_plan_del_proyecto_del_software.pdf.

A, Navarro. 2005. Planificación de proyectos de software. [Online] 2005. http://www.fdi.ucm.es/profesor/anavarro/5._Planificacion_de_proyectos_de_software.pdf.

ALVAREZ. 2003. Aseguramiento de Calidad de Software. [Online] 2003. http://weblogs.udp.cl/nicolas.boettcher/archivos/(9448)Aseguramiento_de_calidad_del_software_la.ppt..

Briand. 1996. *Property-based software engineering measurement, IEEE Transactions on software engineering.* 1996. Vol. 22.

—. 1997. *Response to: Comments on ?Property-Based Software Engineering Measurement: Refining the Additivity Properties?* 1997. Vol. 23.

C, GONZALEZ. 2005. Conceptos generales de calidad total. [Online] 2005. http://www.monografias.com/trabajos11/conge/conge.shtml.

CARBALLO, R. 2006. Métricas para la estimación de los defectos del software. [Online] 2006. http://www.calidaddelsoftware.com/eventos/Solo%20Pruebas2006/MetricasDefectosSoloPruebas2006.pdf.

Chirico, Mike. 2007. SQLite Tutorial. [Online] 2007. http://souptonuts.sourceforge.net/readme_sqlite_tutorial.html.

2002. CMMI. [Online] 2002. http://www.proginternet.com.ar/cmmi.php.

2006. *CMMI, P. T. CMMI® for Development, Version 1.2.* 2006. p. 573.

Diez, E. 2003. Generación Asistida del Mapa de Actividades de Proyectos de Desarrollo de Software. [Online] 2003. http://www.itba.edu.ar/capis/rtis/rtis-5-1/generacionmapaactividades.pdf.

ESTRADA. 2000. *Case Corporativo para el proceso de control de cambios.* La Habana : s.n.,

2000.

Fenton. 1997. *Software Metrics: A Rigorous Approach.* 2. London : Chapman & Hall , 1997.

Fernández. 2001. Calidad del Software. [Online] 2001. http://www.ati.es/gt/calidad-software/presentacion.htm.

FORTIÉN, J. D. 2007. Mejora de proceso en Entornos Virtuales. [Online] 2007. http://www.informaticahabana.com/evento_virtual/files/jandrich.doc.

GARCÍA, M. N. 2004. Medición de la calidad del software en el ámbito de la especificación de requisitos. [Online] 2004. http://www.sc.ehu.es/jiwdocoj/remis/docs/mmg-2000.ppt.

GELVIS SEQUERA, KENNY VIVAS. 1998. CMMI. Planificación del Proyecto. [Online] 1998. http://carolina.terna.net/ingsw3/datos/CMMI_PlanificacionProyecto.ppt#258,3,CMMI_y_la_Planificación_del_Proyecto.

GIRALDO, O. P. 2004. Métricas, Estimación y Planificación en Proyectos de Software. [Online] 2004. http://www.willydev.net/descargas/willydev_planeasoftware.pdf.

GONZÁLEZ. 2001. *Las Métricas de Software y su Uso en la Región.* México : s.n., 2001.

GRAMAJO, E, et al. 2002. Combinación de alternativas para la estimación de proyectos software. [Online] 2002. http://www.itba.edu.ar/capis/webcapis/RGMITBA/articulosrgm/R-ITBA-22-estimacion.pdf.

HIGHSMITH, J A. 2006. Adaptive Software Development: A Collaborative Approach to Managing Complex Systems. Modelos de Estimación. [Online] 2006. http://www.wikilearning.com/articulo/estimacion_de_proyectos-modelos_de_estimacion/9637-3.

HUMPHREY, W S. 2000. *Introducción al Proceso de Software del Equipo.* 2000. p. 5.

J, GRACIA. 2005. CMM - CMMI Nivel 2. [Online] 2005. http://www.ingenierosoftware.com/calidad/cmm-cmmi-nivel-2.php.

Juristo, Natalia, Moreno, Ana M. y Vegas, Sira. 2006. *Técnicas de Evaluación de Software.* 2006.

Kuhlman, Dave. 2009. Beginning Python, Advanced Python, and Python Exercises. [Online] 2009. http://www.rexx.com/~dkuhlman/python_book_01.pdf.

L, Chalini. 2005. Métricas y estándares de comparación in Tesis Maestría. Ciencias con Especialidad en Ingeniería en Sistemas Computacionales. [Online] 2005.

http://www.ewh.ieee.org/reg/9/etrans/Marzo2005/paper119.pdf.

2001. La calidad del software. [Online] 2001.
http://www.idg.es/computerworld/cibernos/cibernos_sept2001.pdf.

LÓPEZ, C. 2007. Mejoramiento continúo principio de Gestión de la Calidad. [Online] 2007.
http://www.gestiopolis.com/canales5/ger/gksa/136.htm.

Lovelle, J.M.C. 1999. Calidad del Software. [Online] 1999.
http://gidis.ing.unlpam.edu.ar/downloads/pdfs/Calidad_software.PDF.

MEHTA, D. 2006. La Industria de la Tecnología de Información en la India. [Online] 2006.
http://www.gobernabilidad.cl/modules.php?name=News&file=print&sid=1066.

Mendoza, G. M. 2006. 2006. [Online] 2006.
http://mena.com.mx/gonzalo/maestria/calidad/presenta/iso_9126-3/.

MGCS. 2008. Modelos de Gestión de la Calidad del Software. [Online] 2008.
http://modelosdegestiondelacalidad.blogspot.com/2008/01/modelo-cmmi_12.html.

PERALTA, M. 2004. Estimación del esfuerzo basada en Casos de Usos. [Online] 2004.
http://www.itba.edu.ar/capis/rtis/rtis-6-1/estimacion-del-esfuerzo-basada-en-casos-de-usos.pdf.

PFLEEGER. 2001. *"Applying Software Metrics".* 2001.

POW-SANG. 2005. Estudio de las técnicas basadas en puntos de función para la estimación del esfuerzo en proyectos software. [Online] 2005.
http://sisbib.unmsm.edu.pe/BibVirtualData/publicaciones/risi/N1_2004/a10.pdf.

PRESSMAN. 1997. *Ingeniería de Software. Un Enfoque.* 5. 1997. pp. 51-61.

—. 1998. *Ingeniería del Software. Un Enfoque Práctico.* 1998.

Ruggles. 1997. *herramientas de gestion de proyecto.* Universidad Politécnica de Valencia : s.n., 1997.

SANCHEZ, M.A. 2004. Metodologías De Desarrollo De Software. [Online] 2004.
http://www.informatizate.net/articulos/metodologias_de_desarrollo_de_software_07062004.html

Smarciani. 2004. Financiamiento Institucional. [Online] 2004.
http://www.inta.gov.ar/bariloche/desarrollo/gesrural/trabajos/planificacion/Archivos/.

Sommerville. 2002. *Ingeniería de Software, Addison Wesley.* 6ta. 2002.

TERUEL, A. 2001. *La Planificación de un Proyecto de Desarrollo de Software.* 2001.

2003. *Un modelo de Referencia para la Gestión de Configuración en la PYME de Software.* La Habana : s.n., 2003. p. 16.

Zuse, H. 1998. *A Framework for Software Measurement.* Berlin : s.n., 1998.

Bibliografía Consultada

[INFORMACIÓN, C. T. Y. D. N. N. C. D. T. D. L]. 2005. *Norma Cubana, ISO/IEC 9126-1:2001, Ingeniería de Software—Calidad del Producto—, Oficina Nacional de Normalización.* 2005.

Chirico, Mike. 2007. SQLite Tutorial. [En línea] 2007. http://souptonuts.sourceforge.net/readme_sqlite_tutorial.html.

DURÁN, M. R. 2007. Mediciones prácticas de software y sistemas (PSM): una propuesta para la producción de software en la UCI. [En línea] 2007. http://www.informaticahabana.com/evento_virtual/files/Trabajo_Informatica_2007_OK.doc.

Edgewall Software, Genshi. 2008. Python toolkit for generation of output for the web. [En línea] 2008. http://genshi.edgewall.org/.

2006. Encuestas. CAELUM. [En línea] 2006. http://www.calidaddelsoftware.com/modules.php?name=Surveys&op=results&pollID=3.

2007. Estimación de proyectos de software. [En línea] 2007. http://www.ie.inf.uc3m.es/grupo/docencia/reglada/psi/unidad8-DOC.pdf.

ESTRADA. 2000. *Case Corporativo para el proceso de control de cambios.* La Habana : s.n., 2000.

GIRALDO, O. P. 2004. Métricas, Estimación y Planificación en Proyectos de Software. [En línea] 2004. http://www.willydev.net/descargas/willydev_planeasoftware.pdf.

Hunter, John, Dale, Darren Dale y Droettboom, Michael. 2009. Matplotlib: python plotting. [En línea] 2009. http://matplotlib.sourceforge.net/index.html.

2005. Ingeniería del software. [En línea] 2005. http://www.monografias.com/trabajos15/ingenieria-software/ingenieria-software.shtml.

2001. *Introducción al Proceso Software Personal.* La Habana : Félix Varela, 2001. pág. 2.

L, Chalini. 2005. Métricas y estándares de comparación in Tesis Maestría. Ciencias con Especialidad en Ingeniería en Sistemas Computacionales. [En línea] 2005. http://www.ewh.ieee.org/reg/9/etrans/Marzo2005/paper119.pdf.

2001. La calidad del software. [En línea] 2001.

http://www.idg.es/computerworld/cibernos/cibernos_sept2001.pdf.

La Torre Hernández, Ludisley. Cepero Núñez, Mariela. 2007. Propuesta de Métricas para perfeccionar la Gestión de la Calidad en los Procesos de Desarrollo de Software. [En línea] 2007. http://bibliodoc.uci.cu/TD/TD_0499_07.pdf.

Lovelle, J.M.C. 1999. Calidad del Software. [En línea] 1999. http://gidis.ing.unlpam.edu.ar/downloads/pdfs/Calidad_software.PDF.

2005. *Manual de Java Script.* 2005.

MEHTA, D. 2006. La Industria de la Tecnología de Información en la India. [En línea] 2006. http://www.gobernabilidad.cl/modules.php?name=News&file=print&sid=1066.

NetBeans UML®. 2009. Project. NetBeans UML® Project. [En línea] 2009. http://uml.netbeans.org/.

Nova. 2005. Planificación de Proyectos de Software. [En línea] 2005. [Citado el:] http://www.getec.etsit.upm.es/articulos/gproyectos/art4.htm.

O'Reilly & Associates. 1998. *¿Qué es el Lenguaje para Modelamiento Unificado (UML)?* 1998.

OpenUP. 2008. OpenUP. [En línea] 2008. http://epf.eclipse.org/wikis/openup/.

PERALTA, M. 2004. Estimación del esfuerzo basada en Casos de Usos. [En línea] 2004. http://www.itba.edu.ar/capis/rtis/rtis-6-1/estimacion-del-esfuerzo-basada-en-casos-de-usos.pdf. http://sisbib.unmsm.edu.pe/BibVirtualData/publicaciones/risi/N1_2004/a10.pdf.

PRESSMAN. 1997. *Ingeniería de Software. Un Enfoque.* 5. 1997. págs. 51-61.

—. 1998. *Ingeniería del Software. Un Enfoque Práctico.* 1998.

Python Software Foundation. 2009. SQLite3 — DB-API 2.0 interface for SQLite databases. [En línea] 2009. http://docs.python.org/lib/module-sqlite3.html.

2006. Requisitos del software: organización y calidad. [En línea] 2006. http://members.fortunecity.com/patriciob/estimacion.html.

Sethanandha, Bhuricha Deen. 2009. Plugin that provides project progress statistics and quality metrics. [En línea] 2009. http://trac-hacks.org/wiki/TracMetrixPlugin.

2009. Sitio Oficial de Python. [En línea] 2009. http://www.python.org/.

2009. Sitio Oficial de SQlite. [En línea] 2009. http://www.sqlite.org/.

2009. Sitio oficial del TRAC. [En línea] 2009. http://trac.edgwall.org.

Team. 2007. Gestión de proyectos con dotProject. [En línea] 2007. http://www.abartiateam.com/dotproject.html.

2003. *Un modelo de Referencia para la Gestión de Configuración en la PYME de Software.* La Habana : s.n., 2003. pág. 16.

Anexos

Anexo # 1

Encuesta sobre el uso de métricas en la UCI.

Datos del Encuestado

Nombre y Apellidos:_____

Correo Electrónico: _____ Rol que desempeña:_____

Facultad:_____Polo:_____Proyecto:_____

Estado de terminación _____

1- ¿Ha realizado estudios relacionados con las métricas del software?

___Si ___No

Comentario: _____

2- Su conocimiento en cuanto a este tema considera que es:

___Alto___ Medio___Bajo

3- ¿Su proyecto tiene almacenados registros históricos?

____Si ____ No ____ No se

Comentario:_____

4- ¿Realizan estimaciones?

____Si ____ No ____ No se

En que se basan:

_____

5- ¿Utilizan métricas en su proyecto?

____Si ____ No ____ No se

En caso afirmativo indique si son Métricas para:

____Proceso de desarrollo de Software.

____Producto de Software.

____Proyecto de Software.

6- ¿Tiene experiencia en la aplicación de estas métricas?

____Esfuerzo (ESF) ____ Productividad (PR) ____ Tiempo de Desarrollo Planificado (TDESP)

____ Líneas de Código Fuente (KLCF) ____otras

Comentarios _____

7- ¿Utiliza otras métricas?

____Si ____ No ____ No se

Comentarios:_____

Anexo # 2

Lista de no conformidades de las pruebas realizadas

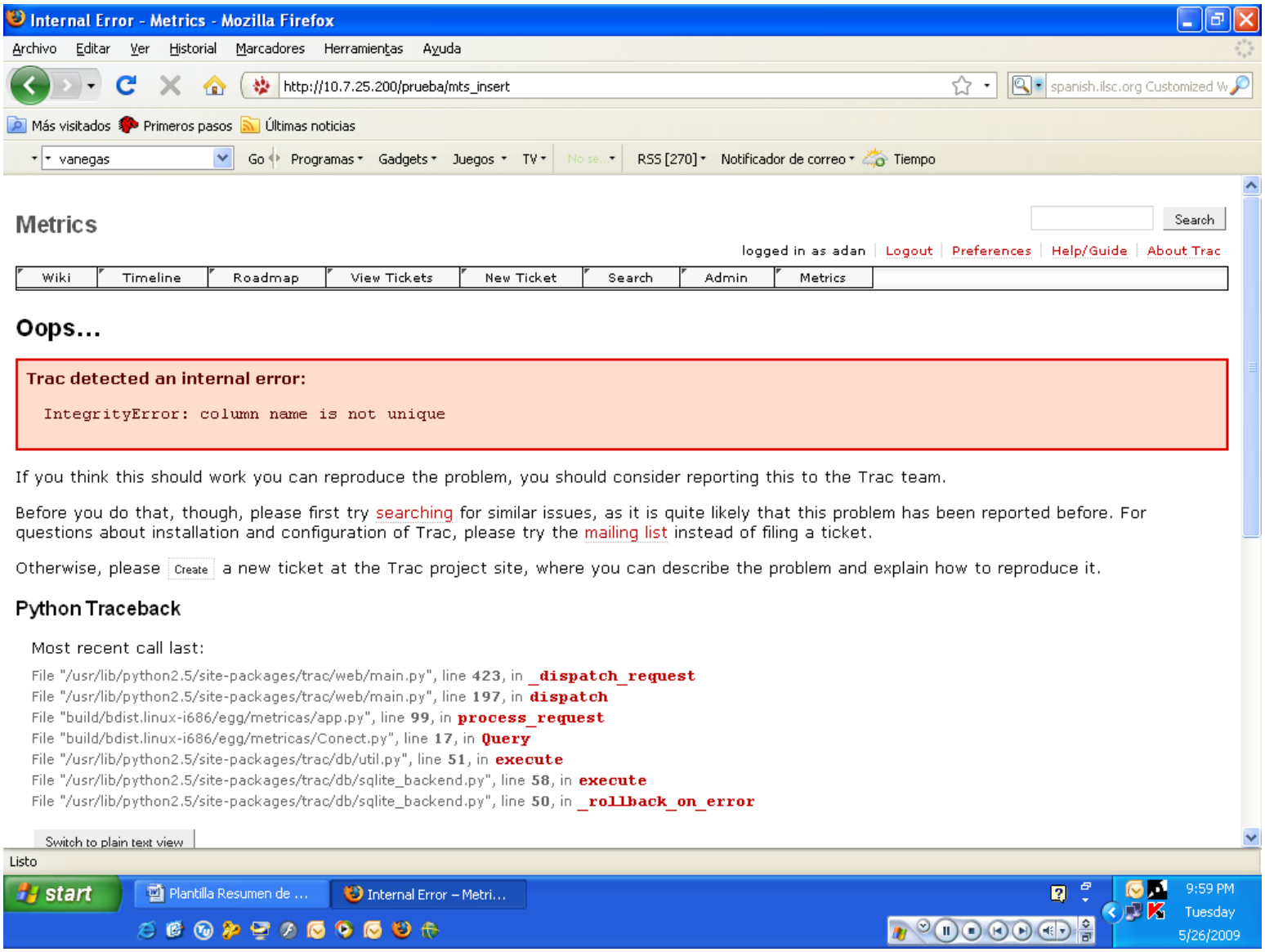
Elemento	No	No conformidad	Aspecto correspondiente	Etapa de detección	Importancia	Recomendación
Aplicación	1	La palabra "wrong" en el mensaje lanzado por el sistema debe ser en mayúscula.	-Usuario: Adan - Manage Metrics -Insert Metric - Poner Nombre - No poner expresión			Las letras que se encuentran luego de un punto, van con mayúscula.
Aplicación	2	El sistema explota, luego de tratar de insertar valores que ya están almacenados en la BD. Anexo 3	-Usuario: Adan - Manage Metrics -Insert Metric -Nombre: Vanegas - Expresión: max(0) Unit: U			Si se adiciona una expresión incorrecta el sistema debe mostrar un mensaje haciendo referencia al mismo. El mensaje debe ser más sencillo.
Aplicación	3	El sistema no informa cuando se adiciona correctamente una métrica.	-Usuario: Adan -Manage Metrics -Insert Metric - Poner Nombre - No poner expresión			Cuando se logra adicionar una métrica, se debe mostrar un mensaje al usuario sobre la acción realizada. Las métricas nuevas aparecen al final de la tabla pero siempre es bueno mostrar un pequeño mensaje, para no tener que ir al final de la lista cada vez que adicione una.

Aplicación	4	Cuando se escoge la opción: Delete sin tener ninguna métrica seleccionada el sistema no muestra ningún mensaje.	-Usuario: Adan - Manage Metrics -Delete Metrics			Cuando no se puede realizar una opción el Sistema debe mostrar un mensaje al usuario indicándole el error cometido.
Aplicación	5	Cuando se modifica una métrica y se cambia el valor, no se muestra ningún mensaje confirmando el cumplimiento de la acción.	-Usuario: Adan - Manage Metrics -Modify Metrics			Cuando se logra modificar una métrica, se debe mostrar un mensaje al usuario sobre la acción realizada. Los cambios aparecen en la tabla pero siempre es bueno mostrar un pequeño mensaje, para no tener que ir al final de la lista cada vez que adicione una.
Aplicación	6	Cuando se modifica una métrica y luego se presiona: Cancel, se borra lo que había sido modificado, pero la ventana se queda abierta.				Cuando se realiza la acción de cancelar y no se ha cambiado nada y se presiona nuevamente cancelar, el sistema debería ocultar la ventana correspondiente.
Aplicación	7	Para ver los reportes hay que presionar el botón: Send	-Usuario: Adan - Reports			En este caso no se está enviando ninguna información, el

						botón debería ser Show, por ejemplo.
Aplicación	8	No se especifica que son las métricas en la: Help in Spanish.	-Usuario: Adan - Help in Spanish			Debería aparecer una breve descripción de que son las métricas,
Aplicación	9	En la Help in Spanish aparece: Errores más comunes**,** primero, el (más), lleva tilde, luego, no hay ninguna descripción.	-Usuario: Adan - Help in Spanish			Debe estar completa la información que aparece en la ayuda.
Aplicación	10	Existen 2 faltas de ortografía en la explicación de **Report of Metrics,** las palabras son: más de un valor…, y en la penúltima oración aparece un: al fecha, donde debería ir: la fecha.	-Usuario: Adan - Help in Spanish			El sistema no debe contener faltas de ortografía.
Aplicación	11	En la descripción de: **Insert Metrics,** primera oración aparecen las palabras: la operación y que pretende, ese qué lleva tilde, sino no tiene concordancia la	-Usuario: Adan - Help in Spanish			El sistema no debe contener faltas de ortografía.

		oración.				
Aplicación	12	En la descripción de: **Insert Metrics,** última oración aparecen las palabras: explicándole que hay un error… No se especifica a quién están explicándole.	-Usuario: Adan - Help in Spanish			Se debe mantener un correcto vocabulario técnico

Anexo # 3


Internal Error - Metrics - Mozilla Firefox
Archivo Editar Ver Historial Marcadores Herramientas Ayuda
http://10.7.25.200/prueba/mts_insert
Más visitados Primeros pasos Últimas noticias
vanegas Go Programas Gadgets Juegos TV RSS [270] Notificador de correo Tiempo
Metrics
Search
logged in as adan Logout Preferences Help/Guide About Trac
Wiki Timeline Roadmap View Tickets New Ticket Search Admin Metrics
Oops...
Trac detected an internal error:
IntegrityError: column name is not unique
If you think this should work you can reproduce the problem, you should consider reporting this to the Trac team.
Before you do that, though, please first try searching for similar issues, as it is quite likely that this problem has been reported before. For questions about installation and configuration of Trac, please try the mailing list instead of filing a ticket.
Otherwise, please Create a new ticket at the Trac project site, where you can describe the problem and explain how to reproduce it.
Python Traceback
Most recent call last:
File "/usr/lib/python2.5/site-packages/trac/web/main.py", line 423, in _dispatch_request
File "/usr/lib/python2.5/site-packages/trac/web/main.py", line 197, in dispatch
File "build/bdist.linux-i686/egg/metricas/app.py", line 99, in process_request
File "build/bdist.linux-i686/egg/metricas/Conect.py", line 17, in Query
File "/usr/lib/python2.5/site-packages/trac/db/util.py", line 51, in execute
File "/usr/lib/python2.5/site-packages/trac/db/sqlite_backend.py", line 58, in execute
File "/usr/lib/python2.5/site-packages/trac/db/sqlite_backend.py", line 50, in _rollback_on_error
Switch to plain text view
Listo
start
Plantilla Resumen de ...
Internal Error – Metri...
9:59 PM
Tuesday
5/26/2009

Anexo # 4

Número de la No Conformidad	Acción Correctiva
1	Se corrigió el error del mensaje de alerta.
2	Se muestra un mensaje de error alertando cuando se trata de insertar una métrica existente en el sistema.
3	Se muestra actualmente un mensaje tanto cuando ser realizó la acción correctamente como cuando no, en caso de cometer un error se informa cual ha sido este.
4	Se muestra un mensaje alertando del error cometido en caso de que exista.
5	Se muestra un mensaje alertando que se modificó correctamente la métrica.
6	Se oculta la ventana cuando se da cancelar dos veces
7	Se cambio el nombre del botón por Show.
8	Se muestra una pequeña descripción de que son las métricas.
9	Se completó la información de la ayuda
10	Se completó la información de la ayuda
11	Se corrigió la falta de ortografía
12	Se corrigió la falta de ortografía

13	Se teclearon los dos puntos luego de la información a explicar.
14	Se corrigieron todas las faltas de ortografía en el sistema.
15	Se tiene un correcto vocabulario técnico.
16	Se tiene un correcto vocabulario técnico.

de software.

mod_python:

Es un módulo del servidor HTTP Apache que integra el lenguaje de programación Python en el servidor Apache. Está destinada a sustituir a (CGI) como método de ejecución de scripts de Python en un servidor web.

Monotone:

Es una herramienta software de fuente abierta para el control distribuido de versiones.

Multiplataforma:

Referido a la capacidad de correr en más de un sistema operativo.

Medida:

Un Medida proporciona una indicación cuantitativa de extensión, cantidad, dimensiones, capacidad y tamaño de algunos atributos de un proceso dado. Es el resultado de una medición.

Medición:

Es la terminología relacionada con el acto de medir software. Una medición (que es una acción) es un conjunto de operaciones cuyo objetivo es determinar el valor de un atributo dado de una entidad, utilizando una forma de medir.

Métricas:

Las métricas son un medio para entender, monitorizar, controlar, predecir y probar el desarrollo software y los proyectos.

Métricas de Software:

La aplicación continua de mediciones basadas en técnicas para el proceso de desarrollo del software y sus productos para suministrar información relevante a tiempo, así el administrador junto con el empleo de estás técnicas mejorará el proceso y sus productos.

Métricas para el proyecto:

Es la medida de alguna propiedad de un entregable del proyecto o del proceso de

administración de proyecto, efectuada para conocer el avance o los desvíos al plan original.

NetBeans IDE y NetBeans Platform:

Es un proyecto exitoso de código abierto con una gran base de usuarios, una comunidad en constante crecimiento, fue fundado por Sun MicroSystems en junio 2000.

Paint:

Biblioteca gráfica.

Paquete de métricas:

Referido a un conjunto de métricas a proponer en la investigación para su aplicación a proyectos de software.

Polimorfismo:

En programación orientada a objetos se denomina polimorfismo a la capacidad que tienen los objetos de una clase de responder al mismo mensaje o evento en función de los parámetros utilizados durante su invocación. Un objeto polimórfico es una entidad que puede contener valores de diferentes tipos durante la ejecución del programa.

Plugin:

Podríamos traducirlo por añadido o conector. Se trata de un pequeño programa que proporciona alguna funcionalidad específica a otra aplicación mayor o más compleja.

PS (postscript):

Biblioteca gráfica.

pylab:

Es un API que semeja a Matlab.

Python:

Es un lenguaje interpretado, logrando ser modificado instantáneamente sin necesidad de ser recompilado, lo cual ayuda a hacer mejoras de forma rápida y simple, es multiplataforma facilitando que cualquier sistema operativo sea compatible con el, siempre y cuando exista un

intérprete programado para el.

RedHat:

Distribución del sistema operativo GNU/Linux.

SCADA:

Acrónimo de Supervisory Control and Data Acquisition (en español, registro de datos y control de supervisión).

Scripts de CGI:

Scripts de CGI es el medio de comunicación que emplea un servidor Web para enviar información útil en ambos sentidos, entre el visualizador (browser) y su propio programa de cómputo.

Software:

Conjunto de instrucciones y datos codificados para ser leídas e interpretadas por una computadora.

Singleton:

Es un patrón de diseño, está diseñado para restringir la creación de objetos pertenecientes a una clase o el valor de un tipo a un único objeto.

Suse:

Distribución del sistema operativo GNU/Linux.

SQL:

Lenguaje de programación declarativo de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones sobre las mismas. Una de sus características es el manejo del álgebra y el cálculo relacional que permite ejecutar consultas o actualización de información en una base de datos.

Sqlite:

Es una pequeña librería programada en lenguaje C que implementa motor de base de datos multiplataforma que no precisa configuraciones.

svn:

Está referido al término subversion sistema de revisiones de versiones de software.

TIC:

Referido a las tecnologías de la información y las comunicaciones.

Trac:

Es un sistema Web libre para la gestión de proyectos y seguimiento de errores, es ligera, simple, extensible y permite enlazar información entre una base de datos de errores de software, un sistema de control de versiones y el contenido de una Wiki.

Ubuntu:

Distribución del sistema operativo GNU/Linux.

VCS:

Version Control System, en español sería sistema de control de versiones

WXAgg:

Biblioteca gráfica.

WX:

Biblioteca gráfica.