INSTITUTO SUPERIORMINERO METALURGICO

"Dr. Antonio Núñez Jiménez".

Facultad de Geología _ Minas

Moa, Holguín

Trabajo de diploma en opción al título

De

Ingeniería en Informática.

SISTEMA DE GESTIÓN DE INFORMES DE ENSAYOS

Autores: Robin Cortina Perdomo

Tutores: Ing. Eloy Rafael Jiménez Iglesias.

Ing. Nerius Perez Toirac

Moa-Cuba 2012

Declaración de Autoría

Declaro que soy el único autor de este trabajo y autorizo al Instituto Superior Minero Metalúrgico de Moa para que haga uso de este como estime pertinente.

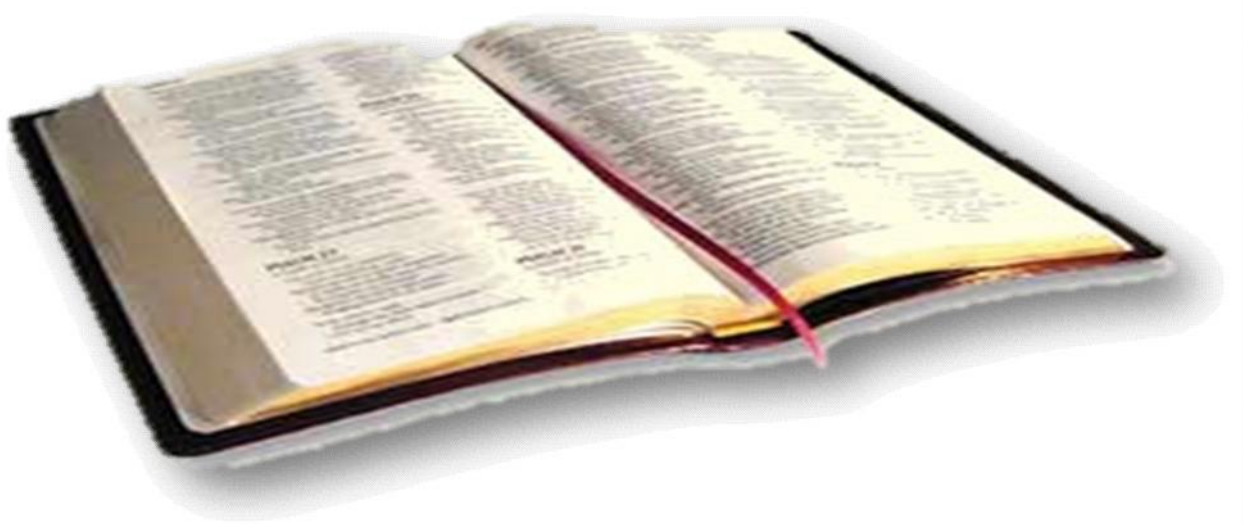
Para que así conste firmo la presente a los ____ días de mes de_____ del 2012.

Robin Cortina Perdomo
Nombre completo del Autor

Ing. Eloy Rafael Jiménez Iglesias
Nombre completo del Tutor

Ing. Nerius Perez Toirac
Nombre completo del Tutor

Pensamiento



Porque Jehová da la sabiduría y de su boca viene el conocimiento y la inteligencia

Proverbios 2:6

Agradecimientos

A Dios sobre todas las cosas por estar conmigo siempre y ayudarme.

A mi madre y mis hermanos.

A mi amada novia , que siempre me da apoyo y me anima con sus besos y dulzura.

A mis compañeros de grupo que han sido una agradable compañía durante estos cinco años.

A mis amigos Acel, Marcheco, Yoeldis , el Rubio, Juanci, a Yoannis y todos los demás por la buena compañía.

A Nerius por ser mi amigo y hermano, que incondicionalmente dedicó su tiempo para ayudarme

A todos los que de una forma u otra aportaron en la realización de este trabajo.

Gracias.

Dedicatoria

Este trabajo va dedicado primeramente a mi madre que ha dedicado su vida para educarme y darme lo que he necesitado para llegar hasta aquí.

A mis hermanos por ayudarme y compartir conmigo mi suerte.

A mi bella novia en especial por ser la más hermosa flor que Me ha regalado Dios.

Resumen

En los umbrales del tercer milenio la informática ha abarcado casi todas las esferas de la sociedad marcando así el factor fundamental para el perfeccionamiento de la gestión de la información en las distintas ramas donde se aplica. Debido a que tanto el desarrollo tecnológico, social como económico van en ascenso a una velocidad exagerada, se produce cada vez más un flujo mayor de informaciones, por lo que se hace inevitable la necesidad de aplicar la informática a estas áreas.

En los laboratorios de Mecánica de Suelos y Rocas del Instituto Superior Minero Metalúrgico de Moa (ISMMM) se realizan informes de diferentes tipos de ensayos. Todo este proceso genera un gran cúmulo de datos que debe ser gestionado. En la actualidad esta información que se encuentra en formatos diversos, ha provocado inconveniencias para su organización. Ante esta dificultad surge la necesidad de crear un sistema que gestione toda la información referente a los informes de ensayos. El sistema permitirá registrar todos los datos concerniente a los ensayos, o sea desde el mismo momento en que se realiza el muestreo, toda su trayectoria en el laboratorio, hasta que se realicen los cálculos y la graficación de los resultados. Además posibilitará la actualización de los informes, siempre que sea necesario.

Abstract

In the thresholds of the third millennium the science of the information has undertaken almost all the spheres of the society marking the fundamental factor for the improvement of the administration of the information in the different branches so where it work hard. Due to that so much the development of the technology, social like economical they go in ascent to an exaggerated speed, it are produced every time plus one old flow of information, for the one which is made inevitable the necessity of applying the science of the information to these areas.

In the laboratories of Mechanics of Floors and Rocks of the Institute Superior Mining metallurgist of Moa (ISMMM) is carried out reports of several types of rehearsals. All this process generates a great heap of datas that should be administered. At the present time this information that meets in diverse formats, has provoked inconvenience for their organization. In front of this difficulty the necessity surges of creating a system that administers all the information with respect to the informs of rehearsals. The system will allow to register all the concerning facts to the rehearsals, that is to say from the same moment in that it are carried out the pattern, all their trajectory in the laboratory, they until are carried out the calculations and the graph of the outputs. It will also facilitate the actualization of the informs, whenever it is necessary

Introducción

Con el pasar de los años la humanidad ha visto acopiar un caudal inmenso de conocimiento. Su permanente transmisión favoreció a que se aceleraran en forma exponencial, tanto el desarrollo científico-tecnológico como el desarrollo de la humanidad. La información se encuentra asociada perennemente a toda obra humana; de hecho, la mayoría de las actividades del ser humano está mediada por la información en consecuencia del propio proceso de interpretación de los distintos datos que nos rodean diariamente. Las organizaciones o instituciones reflejan en su quehacer cotidiano, la necesidad de establecer políticas encaminadas a realizar cambios que tributen a incrementar estructuras más competentes; han desaparecido viejas reglas y han surgido otras que exigen de nuevas concepciones gerenciales. Los usuarios cada vez más exigentes, en cuanto a rapidez, calidad, flexibilidad, requieren de las instituciones u organizaciones lo mejor de sí. Es evidente que para ello la información y el conocimiento deben estar presentes y su manejo es algo esencial en el proceso de toma de decisiones y toda actividad que se genere al respecto. Tradicionalmente, la transferencia de información-conocimiento entre generador y usuario, está muy ligada al uso de fuentes y canales formales e informales; son modalidades interdependientes y complementarias, vinculadas con la estructura y organización social de la Ciencia y la Tecnología (Pérez Giffoni y Sabelli, 2003).

Nuestro país no está exento al desarrollo de software y en los últimos años ha dado un salto cuantitativo y cualitativo en busca de la informatización de la sociedad. Las Universidades son las principales artífice en el desarrollo de software que cada día se adentra más en la sociedad, sometidas a procesos de cambios y transformaciones ante los retos que le plantea la actual sociedad del conocimiento, revelándose cada vez más como componente estratégico en la construcción de una sociedad por el rol social que juegan en la producción y transferencia de conocimientos. La importancia del vínculo entre las universidades, las entidades de producción y la sociedad en general, forman parte del avance de la sociedad de nuestros días.

El Instituto Superior Minero Metalúrgico de Moa (ISMMM) también forma parte de este proceso de desarrollo y evolución de la informática sobre todo en el área de Mecánica de Suelos y Rocas donde se realizan diferentes tipos de experimentos. Los Informes de Ensayos son la técnica utilizadas por los especialistas para almacenar toda la información referente a los diferentes tipos de experimentos que se realizan a la hora de construir cualquier tipo de edificación. En el Instituto Superior Minero Metalúrgico de Moa (ISMMM) también se realizan estos ensayos, de los más realizados en el instituto se encuentran Hinchamiento Controlado, Límites de Contracción, Límites de Plasticidad y Angulo de Reposo, en nuestra investigación se hará alusión a dos de estos Hinchamiento Controlado y Límites de Contracción.

-Hinchamiento Controlado: La finalidad de estos ensayos, es determinar la expansividad o aumento de volumen de una muestra de suelo cohesivo. En suelos que son expansivos, el hinchamiento que experimentan al humedecerse, depende enormemente de las condiciones de compactación. Cuanto más seco esté el suelo, mayor es la posibilidad de que se hinche o colapse, ocurrirá uno u otro según la presión externa que se aplique, sea esta inferior o superior a la presión de hinchamiento. El fenómeno de cambio de volumen de un suelo arcilloso es resultado directo de la disponibilidad y variación de la cantidad de agua que él posea.

-Límites de Contracción: El objetivo del ensayo es determinar el límite de contracción de un suelo. El límite de contracción se define como el contenido de humedad al cual un material al ser secado cesa de perder volumen. La determinación de los límites de consistencia, es muy importante porque nos muestra el comportamiento del suelo con las diferentes cantidades de agua presentes en él. Una de las características importantes es el comportamiento del suelo sin la presencia de agua, y es por eso que se realiza el ensayo de límite de contracción; ya que si el agua en el suelo está en proceso de evaporación, éste adquiere mayor solidez y va contrayéndose hasta que llega un momento en que cesa la contracción, no obstante continua el proceso de evaporación. A este límite en que cesa la contracción del suelo, aunque continué la evaporación del agua, es al que se llama límite de contracción.

Es de suma importancia realizar estos informes de ensayos por su aplicación diaria en la labor que se realiza en los laboratorios de mecánica de suelos y rocas, sin embargo en este momento no se cuenta con una herramienta informática que permita la realización del proceso de gestión de Informes de Ensayos lo que entorpece el desarrollo de esta actividad. Por tales motivos se decidió realizar un sistema informático que gestione el proceso de realización de los Informes de Ensayos.

Por tanto se tiene como **situación problemática** la siguiente: Actualmente en los Laboratorios de Mecánica de Suelos y Rocas del Instituto Superior Minero Metalúrgico de Moa (ISMMM) el proceso de realización de Informes de Ensayos se realiza de forma manual por lo que provoca serias dificultades durante la realización del informe, por esta razón se dio la tarea de realizar un sistema informático para gestionar la realización de Informes de Ensayos. Dada esta situación define como **problema científico** Necesidad de informatizar el cálculo de los ensayos de laboratorios de Mecánica de suelos y rocas en la disciplina de Geología Aplicada. Es por ello que el presente trabajo tiene como **objeto de estudio** El proceso de informatización de la gestión de informes de ensayos, el **campo de acción** La informatización de los procesos de ensayos en la carrera ingeniería geológica en el ISMM de Moa. Se tiene como **objetivo general** Elaborar un software para el cálculo de los ensayos de laboratorio de Mecánica de suelos y rocas en la carrera de ingeniería geológica. Para guiar nuestra investigación se plantea la siguiente **Idea a defender** La elaboración de un software para el cálculo de los ensayos de laboratorio de Mecánica de Suelos y Rocas en la carrera de ingeniería geológica que facilitará el trabajo de los especialistas permitiéndole mayor rapidez y eficiencia en el proceso.

Debido a esto planteamos los siguientes **objetivos específicos**:

-Realizar un análisis del estado actual del funcionamiento de los Informes de Ensayo.

-Realizar un estudio acerca de las tecnologías y herramientas actuales necesarias para la posterior implementación del sistema.

-Realizar el análisis, diseño e implementación del sistema.

-Garantizar la seguridad de los datos que se gestionan.

-Realizar la documentación del sistema propuesto.

Para el logro de los objetivos fue necesario plantearse las **siguientes tareas**:

-Detallar el proceso de gestión de los Informes de Ensayos.

-Fundamentación y análisis bibliográfico que permita la familiarización con las principales tecnologías y herramientas utilizadas actualmente para el desarrollo de aplicaciones informáticas.

-Seguir cada una de las etapas de la ingeniería del software hasta la implementación de un sistema que permita la gestión de los Informes de Ensayo.

-Crear una base de datos para almacenar las informaciones.

-Realizar el estudio de factibilidad que permita mostrar los costos y beneficios del sistema y su desarrollo.

-Prueba y Documentación del Sistema.

-Desarrollo del Manual de Usuarios.

Para cumplir estas tareas se han empleado **métodos teóricos** y **empíricos** de la investigación científica. Entre los métodos empíricos usados podemos citar la observación, entrevista, la modelación y el análisis de documentos para la recopilación de la información. La observación fue útil para entender el comportamiento del sistema y sus especificaciones. La entrevista nos ayudó a determinar los principales requerimientos del sistema y las nuevas funcionalidades que necesita plasmados en las historias de usuarios. Mediante el análisis de la documentación disponible conocimos el funcionamiento actual del proceso de realización de informes de ensayos.

Los métodos teóricos proporcionaron calidad a la investigación. En el desarrollo del proceso de investigación se usaron el análisis y síntesis para la recopilación y el procesamiento de la información obtenida en los métodos empíricos y arribar a las conclusiones de la investigación. Con la representación de la realidad se logró detectar problemas en la forma actual de manejar la

información y encontrar las funcionalidades que debe de tener el sistema que se propone, que lo harán más completo y le brindarán satisfacción al usuario con un producto de mayor calidad.

Aportes de la investigación:

Este trabajo tiene como **aporte práctico** la elaboración de software para el cálculo e interpretación de los ensayos de laboratorios de mecánica de suelos y rocas.

Capítulo 1 Fundamentación

1.1 Introducción

En este capítulo se define el flujo actual de los procesos, se hace un análisis crítico de la ejecución de los procesos, se hace una breve referencia sobre el estado del arte del sistema. También se hace una descripción sobre las tendencias y tecnologías actuales sobre las que se apoya el sistema, por otra parte se especifican las herramientas utilizadas para el desarrollo del sistema y el porqué de su uso.

1.2 Estado del arte

En el Instituto Superior Minero Metalúrgico de Moa no existe ningún software que realice la gestión de informe de ensayos. Se ha realizado la investigación acerca de algún software que realice estas funcionalidades o parte de ellas sin obtener resultados satisfactorios. Por tanto no hay un software que satisfaga la necesidad actual de realizar los informes de ensayos.

1.3 Herramientas y tecnologías para el desarrollo de Aplicaciones.

1.3.1 Lenguajes de Programación

Un lenguaje de programación es un lenguaje que puede ser utilizado para controlar el comportamiento de una máquina, particularmente una computadora. Consiste en un conjunto de reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos, respectivamente. A continuación se realiza un estudio de los principales lenguajes de programación existentes que se utilizan para la creación de aplicaciones de escritorio, como son Delphi, C++, Java, entre otros. (Subirós, 2009).

1.3.1.1 Delphi

Es un entorno de desarrollo de software diseñado para la programación de propósito general con énfasis en la programación visual. En Delphi se utiliza como lenguaje de programación una versión moderna de Pascal llamada Object Pascal. Es producido comercialmente por la empresa estadounidense CodeGear adquirida en Mayo de 2008 por Embarcadero Technologies, una empresa del grupo Thoma Cressey Bravo, en una suma que ronda los 30

millones de dólares. En sus diferentes variantes, permite producir archivos ejecutables para Windows, Linux y la plataforma .NET. (Delphi, 2004)

1.3.1.2 C++.

Es un lenguaje de programación, diseñado a mediados de los años 1980, por Bjarne Stroustrup, como extensión del lenguaje de programación C. El nombre C++ fue propuesto por Rick Masciatti en el año 1983, cuando el lenguaje fue utilizado por primera vez fuera de un laboratorio científico. Antes se había usado el nombre "C con clases". Se puede decir que abarca tres paradigmas de la programación:

• La programación estructurada.
• La programación genérica.
• La programación orientada a objetos. (C++, 2002)

1.3.1.3 Java.

Es un lenguaje de programación desarrollado por Sun Microsystems a principios de los años 1990. Las aplicaciones Java están típicamente compiladas en un bytecode, aunque la compilación en código máquina nativo también es posible. La implementación original y de referencia del compilador, la máquina virtual y las librerías de clases de Java fueron desarrolladas por Sun Microsystems en 1995. Desde entonces, Sun ha controlado las especificaciones, el desarrollo y evolución del lenguaje a través del Java Community Process, si bien otros han desarrollado también implementaciones alternativas de estas tecnologías de Sun, algunas incluso bajo licencias de software libre. (Eckel, 2000)

Las características principales que nos ofrece Java son:

-Simple

Ofrece toda la funcionalidad de un lenguaje potente. Debido a que C y C++ son los lenguajes más difundidos, Java se diseñó para ser parecido a C++ y así facilitar un rápido y fácil aprendizaje. Elimina muchas de las características de otros lenguajes para mantener reducidas las especificaciones del lenguaje y añadir características muy útiles como el garbage collector (reciclador de memoria dinámica).No es necesario preocuparse de liberar memoria, el reciclador se encarga de ello y como es un thread de baja prioridad, cuando

entra en acción, permite liberar bloques de memoria muy grandes, lo que reduce la fragmentación de la memoria.

-Orientado a objetos

Con el objetivo de mantener la simplicidad del lenguaje. Java trabaja con sus datos como objetos y con interfaces a esos objetos. Soporta las tres características propias del paradigma de la orientación a objetos: encapsulación, herencia y polimorfismo. Las plantillas de objetos son llamadas, clases y sus copias, instancias. Estas instancias, necesitan ser construidas y destruidas en espacios de memoria. Incorpora funcionalidades como por ejemplo, la resolución dinámica de métodos mediante una interfaz específica llamada RTTI (Run Time Type Identification) que define la interacción entre objetos excluyendo variables de instancias o implementación de métodos. Las clases en Java tienen una representación en el runtime que permite a los programadores interrogar por el tipo de clase y enlazar dinámicamente la clase con el resultado de la búsqueda.

-Distribuido

Se ha construido con extensas capacidades de interconexión TCP/IP. Existen librerías de rutinas para acceder e interactuar con protocolos como http y ftp. Esto permite a los programadores acceder a la información a través de la red con tanta facilidad como a los ficheros locales.

Proporciona las librerías y herramientas para que los programas puedan ser distribuidos, es decir, que se corran en varias máquinas, interactuando.

-Robusto

Realiza verificaciones en busca de problemas tanto en tiempo de compilación como en tiempo de ejecución. La comprobación de tipos en Java ayuda a detectar errores, lo antes posible, en el ciclo de desarrollo.

Java obliga a la declaración explícita de métodos, reduciendo así las posibilidades de error. Maneja la memoria para eliminar las preocupaciones por parte del programador de la liberación o corrupción de memoria. También implementa los arrays auténticos, en vez de listas enlazadas de punteros, con comprobación de límites, para evitar la posibilidad de sobrescribir o corromper memoria resultado de punteros que señalan a zonas equivocadas. Estas

características reducen drásticamente el tiempo de desarrollo de aplicaciones en Java.

-Arquitectura neutral

Para establecer Java como parte integral de la red, el compilador Java compila su código a un fichero objeto de formato independiente de la arquitectura de la máquina en que se ejecutará. Cualquier máquina que tenga el sistema de ejecución (run-time) puede ejecutar ese código objeto, sin importar en modo alguno la máquina en que ha sido generado. Actualmente existen sistemas run-time para Solaris 2.x, SunOs 4.1.x, Windows 95, Windows NT, Linux, Irix, Aix, Mac, Apple y probablemente haya grupos de desarrollo trabajando en el porting a otras plataformas.

-Seguro

La seguridad en Java tiene dos facetas. En el lenguaje, características como los punteros o el casting se eliminan para prevenir el acceso ilegal a la memoria. Cuando se usa Java para crear un navegador, se combinan las características del lenguaje con protecciones de sentido común aplicadas al propio navegador. El código Java pasa muchos tests antes de ejecutarse en una máquina. El código se pasa a través de un verificador de byte-codes que comprueba el formato de los fragmentos de código y aplica un probador de teoremas para detectar fragmentos de código ilegal -código que falsea punteros, viola derechos de acceso sobre objetos o intenta cambiar el tipo o clase de un objeto. El Cargador de Clases también ayuda a Java a mantener su seguridad, separando el espacio de nombres del sistema de ficheros local, del de los recursos procedentes de la red. Esto limita cualquier aplicación del tipo Caballo de Troya, ya que las clases se buscan primero entre las locales y luego entre las procedentes del exterior.

Las clases importadas de la red se almacenan en un espacio de nombres privado, asociado con el origen. Cuando una clase del espacio de nombres privado accede a otra clase, primero se busca en las clases predefinidas (del sistema local) y luego en el espacio de nombres de la clase que hace la referencia. Esto imposibilita que una clase suplante a una predefinida.

-Multihebra

Soporta sincronización de múltiples hilos de ejecución (multi threading) a nivel de lenguaje, especialmente útiles en la creación de aplicaciones de red distribuidas. Así, mientras un hilo se encarga de la comunicación, otro puede interactuar con el usuario mientras otro presenta una animación en pantalla y otro realiza cálculos.

1.3.2 Sistemas Gestores de bases de datos (SGBD)

1.3.2.1 MySQL

En el mundo de las base de datos cliente/servidor existe una feroz competencia. Muchos “grandes” compiten por ser la prestación más rápida, más segura, más confiable, más robusta. Los principales colosos de este mundo son, sin dudas Microsoft SQL Server y Oracle, y otros no tan conocidos como DB2, Sybase, Informix y Postgres. Sin embargo, MySQL no se queda atrás y desde hace poco se ha convertido en una importante competencia para estos productos, ya que cuenta con características comparables y muchas veces mejores. La empresa que desarrolla MySQL es MySQL AB, de origen sueco. (Cuenca, 2009)

MySQL es un sistema de administración de bases de datos. Una base de datos es una colección estructurada de tablas que contienen datos. Esta puede ser desde una simple lista de compras a una galería de pinturas o el vasto volumen de información en una red corporativa. Para agregar, acceder a y procesar datos guardados en un computador, usted necesita un administrador como MySQL Server. Dado que los computadores son muy buenos manejando grandes cantidades de información, los administradores de bases de datos juegan un papel central en computación, como aplicaciones independientes o como parte de otras aplicaciones.

MySQL es un sistema de administración relacional de bases de datos. Una base de datos relacional archiva datos en tablas separadas en vez de colocar todos los datos en un gran archivo. Esto permite velocidad y flexibilidad. Las tablas están conectadas por relaciones definidas que hacen posible combinar datos de diferentes tablas sobre pedido.

MySQL es un sistema de gestión de base de datos relacional, multi-hilo y multiusuario, con más de seis millones de instalaciones. MySQL AB desarrolla MySQL como software libre en un esquema de licenciamiento dual. Por un lado lo ofrece bajo la GNU GPL, pero, entidades que quieran incorporarlo en productos privativos pueden comprar a la empresa una licencia que les permita ese uso. Está desarrollado en su mayor parte en ANSIC. Al contrario de proyectos como el Apache, donde el software es desarrollado por una comunidad pública, y el copyright del código está en poder del autor individual, MySQL está poseído y patrocinado por una empresa privada, que posee el copyright de la mayor parte del código. (History of MySQL).

1.3.2.2 PostgreSQL

Es un servidor de base de datos relacional orientada a objetos de software libre, liberado bajo la licencia BSD. Como muchos otros proyectos open source, el desarrollo de PostgreSQL no es manejado por una sola compañía sino que es dirigido por una comunidad de desarrolladores y organizaciones comerciales las cuales trabajan en su desarrollo. Dicha comunidad es denominada el PGDG (PostgreSQL Global Development Group).

Las funciones pueden ser definidas para ejecutarse con los derechos del usuario ejecutor o con los derechos de un usuario previamente definido. El concepto de funciones, en otros DBMS, son muchas veces referidas como "procedimientos almacenados" (stored procedures en inglés).

Mediante un sistema denominado MVCC (Acceso concurrente multiversión por sus siglas en inglés) PostgreSQL permite que mientras un proceso escribe en una tabla, otros accedan a la misma tabla sin necesidad de bloqueos. Cada usuario obtiene una visión consistente de lo último a lo que se le hizo commit. Esta estrategia es superior al uso de bloqueos por tabla o por filas común en otras bases, eliminando la necesidad del uso de bloqueos explícitos. El proyecto PostgreSQL sigue actualmente un activo proceso de desarrollo a nivel mundial gracias a un equipo de desarrolladores y contribuidores de código abierto. Está considerado como el gestor de base de datos de software libre más avanzado del mundo. Es un sistema objeto – relacional pues incluye aspectos del paradigma orientada a objetos, tales como la herencia, tipos de

datos, funciones, restricciones, reglas e integridad transaccional, aunque no llega a ser un gestor con orientación a objetos pura. (PostgreSQL, 2006).

Gracias a su licencia BSD, se permite la utilización del código para ser comercializado. Uno de los casos ejemplo es la de Enterprise DB (Postgresql Plus), la cual incluye varios agregados y una interfaz de desarrollo basada en Java. (PostgreSQL, Award Winning Software, 2008)

1.3.3 Herramientas

1.3.3.1 NetBeans

- Es un proyecto de código abierto de gran éxito con una gran base de usuarios, una comunidad en constante crecimiento, y con cerca de 100 socios en todo el mundo. Sun Micro Systems fundó el proyecto de código abierto NetBeans en junio 2000 y continúa siendo el patrocinador principal de los proyectos. La Plataforma NetBeans es una base modulary extensible usada como una estructura de integración para crear aplicaciones. 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.

El IDE NetBeans 7.0 es una herramienta para programadores, pensada para escribir, compilar, depurar y ejecutar programas. Existe además un número importante de módulos para extender el IDE NetBeans. El IDE NetBeans es un producto libre y gratuito sin restricciones de uso. Soporta el desarrollo de todos los tipos de aplicación Java (J2SE, JavaEE, Web, EJB y aplicaciones móviles).Entre sus características se encuentran un sistema de proyectos basado en Ant, control de versiones y refactoring. El IDE NetBeans IDE 7.0 corre en diferentes sistemas operativos entre los que podemos mencionar Linux, Mac OS X, Solaris y Windows. Como se puede observar se ha lanzado con soporte para la mayoría de las plataformas. Como requerimiento se necesita tener previamente instalado el JDK 6.0 o 7.0. (Eckel, 2000).

1.3.3.2 Visual Paradigm 5.3

Visual Paradigm para UML es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El software de modelado UML ayuda a una más rápida construcción de aplicaciones de calidad, mejores y a un menor coste. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. (Visual Paradigm , 2009)

1.3.3.3 E/R Studio

Es una herramienta de modelado de datos fácil de usar y multinivel, para el diseño y construcción de bases de datos a nivel físico y lógico. Direcciona las necesidades diarias de los administradores de bases de datos, desarrolladores y arquitectos de datos que construyen y mantienen aplicaciones de bases de datos grandes y complejas. ER/Studio está equipado para crear y manejar diseños de bases de datos funcionales y confiables. Ofrece fuertes capacidades de diseño lógico, sincronización bidireccional de los diseños físicos y lógicos, construcción automática de bases de datos, documentación y fácil creación de reportes. (ER/Studio , 2007)

1.4 Metodologías Existentes

Hoy en día, llevar a cabo el desarrollo de un buen software depende de un gran número de actividades y etapas donde elegir la mejor metodología para el equipo influye directamente en el futuro éxito del producto. El papel preponderante de las metodologías es sin duda esencial en un proyecto y en el paso inicial, que debe encajar en el equipo, guiar y organizar actividades que conlleven a las metas trazadas en el grupo.

Una metodología para el desarrollo de un proceso de software es un conjunto de filosofías, fases, procedimientos, reglas, técnicas, herramientas, documentación y aspectos de formación para los desarrolladores de sistemas informáticos. Las metodologías existentes en la actualidad se dividen en dos grandes grupos atendiendo a sus características: las metodologías tradicionales (RUP, MSF) y las metodologías ágiles (XP, SCRUM). Las primeras están pensadas para el uso exhaustivo de documentación durante

todo el ciclo del proyecto mientras que las segundas ponen vital importancia en la capacidad de respuesta a los cambios, la confianza en las habilidades del equipo y al mantener una buena relación con el cliente. (Zulueta ,2010)

1.4.1 RUP

Es un proceso para el desarrollo de un software que define claramente quién, cómo, cuándo y qué debe hacerse en el proyecto. Como tres características esenciales está dirigido por casos de uso: que orientan al proyecto a la importancia para el usuario y lo que se quiere, está centrado en la arquitectura: que relaciona la toma de decisiones que indican cómo tiene que ser construido el sistema y en qué orden, y es iterativo e incremental: donde divide el proyecto en mini-proyectos donde los casos de uso y al arquitectura cumplen sus objetivos de manera depurada. RUP propone cuatro etapas para el desarrollo de un producto: Inicio, Elaboración, Construcción y Transición, cada una de ellas compuesta de una o varias iteraciones. Estas etapas revelan que para producir una versión del producto en desarrollo se emplean todas las actividades de ingeniería pero con diferente énfasis; en las primeras versiones se hace más énfasis en el modelado del negocio, requisitos, análisis y diseño; mientras en las posteriores el énfasis recae sobre las actividades de implementación, pruebas y despliegue. Además contempla flujos de trabajo de soporte que involucran actividades de planificación de recursos humanos tecnológicos y financieros. El Proceso Unificado de Desarrollo tiene 9 flujos de trabajo principales. Los 6 primeros son conocidos como flujos de ingeniería y los tres últimos como de apoyo. Flujos de trabajo:

- Modelamiento del negocio
- Requerimientos
- Análisis y diseño
- Implementación
- Prueba (Testeo)
- Instalación
- Administración del proyecto
- Administración de configuración y cambios
- Ambiente (Zulueta, 2010)

1.4.2 Scrum

Es una metodología ágil de desarrollo de proyectos que toma su nombre y principios de los estudios realizados sobre nuevas prácticas de producción por Hirotaka Takeuchi e Ikujijo Nonaka a mediados de los 80´ Aunque surgió como modelo para el desarrollo de productos tecnológicos también se emplea en entornos que trabajan con requisitos inestables y que requieren rapidez y flexibilidad; situaciones frecuentes en el desarrollo de determinados sistemas de software. Scrum es una metodología de desarrollo muy simple, que requiere trabajo duro porque no se basa en el seguimiento de un plan, sino en la adaptación continua a las circunstancias de la evolución del proyecto (Palacio, 2006).

1.4.3 XP (Extreme Programming).

La Programación Extrema es una metodología ligera de desarrollo de software que se basa en la simplicidad, la comunicación y la realimentación o reutilización del código desarrollado. XP es un enfoque de la ingeniería de software formulado por Kent Beck y De Jean, Extreme Programming Explained: Embrace Change (1999). Es la más destacada de los procesos ágiles de desarrollo de software. Al igual que éstos, la programación extrema se diferencia de las metodologías tradicionales principalmente en que pone más énfasis en la adaptabilidad que en la previsibilidad.

Los defensores de XP consideran que los cambios de requisitos sobre la marcha son un aspecto natural, inevitable e incluso deseable del desarrollo de proyectos. Creen que ser capaz de adaptarse a los cambios de requisitos en cualquier punto de la vida del proyecto es una aproximación mejor y más realista que intentar definir todos los requisitos al comienzo del proyecto e invertir esfuerzos después en controlar los cambios en los requisitos. Se puede considerar la programación extrema como la adopción de las mejores metodologías de desarrollo de acuerdo a lo que se pretende llevar a cabo con el proyecto, y aplicarlo de manera dinámica durante el ciclo de vida del software. (Metodología, 2008).

1.4.3.1 Fases de la metodología XP

Las prácticas que componen la programación extrema se pueden agrupar en cuatro grandes bloques: plan, diseño, codificación y pruebas. Sin embargo, estos bloques no deben realizarse en orden, si no que cada uno consta de una serie de actividades, y todas ellas se irán realizando de manera evolutiva.

Planificación

- Se escriben historias de usuario, cuya idea principal es describir un caso de uso en dos o tres líneas con terminología del cliente (de hecho, se supone que deben ser escritos por el mismo), de tal manera que se creen test de aceptación para el user storie y permita hacer una estimación de tiempo de desarrollo del mismo.

- Se crea un plan de lanzamiento (reléase planning), que debe servir para crear un calendario que todos puedan cumplir y en cuyo desarrollo hayan participado todas las personas involucradas en el proyecto. Se usará como base los user stories, participando el cliente en la elección de los que se desarrollarán, y según las estimaciones de tiempo de los mismos se crearán las iteraciones del proyecto.

- El desarrollo se divide en iteraciones, cada una delas cuales comienza con un plan de iteración para el que se eligen las user stories a desarrollar y las tareas de desarrollo.

- Se cambia el proceso lo que sea necesario para adaptarlo a tu proyecto.

Diseño

- Se eligen los diseños más simples que funcionen.

-Se elige una metáfora del sistema para que el nombrado de clases, etcétera, siga una misma línea, facilitando la reutilización y la comprensión del código. - Se escriben tarjetas CRC de clase-responsabilidades-colaboración para cada objeto, que permiten abstraerse el pensamiento estructurado y que el equipo de desarrollo al completo participe en el diseño.

Codificación

- El cliente está siempre disponible, a ser posible cara a cara. La idea es que forme parte del equipo de desarrollo, y esté presente en todas las fases de XP (escribe las historias de usuarios con la ayuda de los desarrolladores, participa en la elección de los que entrarán en el plan de lanzamientos, prueba

pequeños lanzamientos, participa en las pruebas de funcionalidad...). La idea es usar el tiempo del cliente para estas tareas en vez de para que cree una detalladísima especificación de requisitos, y evitar la entrega de un producto peor que le hará perder tiempo.

- El código se ajustará a unos estándares de codificación, asegurando la consistencia y facilitando la comprensión y refactorización del código.
- Las pruebas unitarias se codifican antes que el código en sí, haciendo que la codificación de este último sea más rápida, y que cuando se afronte la misma se tenga más claro qué objetivos tiene que cumplir lo que se va a codificar.
- La programación del código se realizará en parejas, para aumentarla calidad del mismo. En cada momento, sólo habrá una pareja de programadores integrando código.
- Se integra código y se lanza dicha integración de manera frecuente, evitando divergencias en el desarrollo y permitiendo que todo el mundo trabaje con la última versión del desarrollo. De esta manera, se evitará pasar grandes periodos de tiempo integrando el código al final del desarrollo, ya que las incompatibilidades habrán sido detectadas enseguida.
- Se usa la propiedad colectiva del código, lo que se traduce en que cualquier programador puede cambiar cualquier parte del código. El objetivo es fomentarla contribución de ideas por parte de todo el equipo de desarrollo
- Se deja la optimización para el final
- No se hacen horas extra de trabajo

Pruebas

- Todo el código debe tener pruebas unitarias, y debe pasarlas antes de ser lanzado.
- Cuando se encuentra un error de codificación o bug, se desarrollan pruebas para evitar volver a caer en el mismo.

-Se realizan pruebas de aceptación frecuentemente, publicando los resultados de las mismas. Estas pruebas son generadas a partir de las user stories elegidas para la iteración, y son "pruebas de caja negra", en las que el cliente verifica el correcto funcionamiento de lo que se está probando. Cuando se pasa la prueba de aceptación, se considera que el correspondiente user storie se ha completado.

1.5 Propuesta de solución

Se desarrollará una aplicación de escritorio utilizando como lenguaje de programación Java, la cual guardara sus datos en una base de Datos de PostgresSQL y se utilizará la metodología XP (Programación Extrema) para llevar a cabo el proceso de desarrollo del software.

1.5.1 ¿Por qué Java?

Atendiendo las características de los lenguajes antes mencionados hemos seleccionado Java para el desarrollo de nuestra aplicación por contar con las siguientes características:

Java reduce en un 50% los errores más comunes de programación con respecto a lenguajes como C y C++ al eliminar muchas de las características de éstos, entre las que destacan:

- Aritmética de punteros.
- No existen referencias.
- Registros (struct).
- Definición de tipos (typedef).
- Macros (#define).
- Necesidad de liberar memoria (free).

Aunque, en realidad, lo que hace es eliminar las palabras reservadas (struct, typedef), ya que las clases son algo parecido. Tiene ventajas en cuanto a seguridad ya que al tener la característica de ser interpretado el código Java pasa a través de un verificador de byte-codes que comprueba el formato de los fragmentos de código y aplica un probador de teoremas para detectar fragmentos de código ilegal -código que falsea punteros, viola derechos de acceso sobre objetos o intenta cambiar el tipo o clase de un objeto. El Cargador de Clases, separando el espacio de nombres del sistema de ficheros local, del de los recursos procedentes de la red. Esto limita cualquier aplicación del tipo Caballo de Troya, ya que las clases se buscan primero entre las locales y luego entre las procedentes del exterior.

1.5.2 ¿Por qué PostgresSQL?

Se Escogió a Postgres como Gestor de Base de Datos teniendo en cuenta que es un servidor de base de datos relacional orientada a objetos de software

libre, liberado bajo la licencia BSD y dirigido por una comunidad de desarrolladores y organizaciones comerciales las cuales trabajan en su desarrollo.

Características:

-Alta concurrencia.

-Amplia variedad de tipos nativos.

-Es altamente extensible: soporta operadores, funciones, métodos de acceso y tipos de datos definidos por el usuario.

-Mantiene una ruta a todas las transacciones realizadas por los usuarios de la base de datos.

PostgreSQL soporta funciones que retornan "filas", donde la salida se puede presentar como un conjunto de valores que pueden ser tratados igual a una fila retornada por una consulta (query en inglés).

Las funciones pueden ser definidas para ejecutarse con los derechos del usuario ejecutor o con los derechos de un usuario previamente definido.

El concepto de funciones, en otros DBMS, son muchas veces referidas como "procedimientos almacenados" (stored procedures en inglés). (PostgreSQL, 2006)

1.5.3 ¿Por qué XP (Programación Extrema)?

Se escogió XP para el desarrollo de este proyecto, ya que es una metodología ágil, diseñada para entornos dinámicos, pensada para equipos pequeños (hasta 10 programadores), orientada fuertemente hacia la codificación. Dentro de sus principales roles podemos encontrar el Programador, el Jefe de Proyecto, el Cliente, el Encargado de Pruebas, el Rastreador y el Entrenador.

La metodología consiste en una programación rápida o extrema, cuya particularidad es tener como parte del equipo, al usuario final, pues es uno de los requisitos para llegar al éxito del proyecto. Las características fundamentales son:

-Desarrollo iterativo e incremental: pequeñas mejoras, unas tras otras.

-Pruebas unitarias continuas, frecuentemente repetidas y automatizadas, incluyendo pruebas de regresión. Se aconseja escribir el código de la prueba antes de la codificación.

-Programación por parejas: se recomienda que las tareas de desarrollo se lleven a cabo por dos personas en un mismo puesto. Se supone que la mayor calidad del código escrito de esta manera -el código es revisado y discutido mientras se escribe- es más importante que la posible pérdida de productividad inmediata.

-Corrección de todos los errores antes de añadir nueva funcionalidad.

-Hacer entregas frecuentes.

-Refactorización del código, es decir, reescribir ciertas partes del código para aumentar su legibilidad y mantenibilidad pero sin modificar su comportamiento.

-Las pruebas han de garantizar que en la refactorización no se ha introducido ningún fallo.

-Simplicidad en el código, es la mejor manera de que las cosas funcionen.

-Cuando todo funcione se podrá añadir funcionalidad si es necesario. La programación extrema apuesta que es más sencillo hacer algo simple y tener un poco de trabajo extra para cambiarlo si se requiere, que realizar algo complicado y quizás nunca utilizarlo.

-La simplicidad y la comunicación son extraordinariamente complementarias. Con más comunicación resulta más fácil identificar qué se debe y qué no se debe hacer. Mientras más simple es el sistema, menos tendrá que comunicar sobre este, lo que lleva a una comunicación más completa, especialmente si se puede reducir el equipo de programadores.

Ventajas

- Apropiado para entornos volátiles.

- Estar preparados para el cambio, significa reducir su coste.

- Planificación más transparente para los clientes, ya conocen las fechas de entrega de funcionalidades. Vital para su negocio.

- Permite definir en cada iteración cuales son los objetivos de la siguiente.

-Permite la retroalimentación.

-La presión está a lo largo de todo el proyecto y no en una entrega final. (Subirós, 2009).

1.6 Arquitectura

La Arquitectura es el esqueleto o base de una aplicación. Representa la organización fundamental de un sistema. Desde los pequeños programas hasta

los sistemas más grandes poseen una estructura y un comportamiento que los hace clasificables según su "arquitectura". En la Web es muy común la utilización de la arquitectura “n-capas”, “MVC”, entre otras. El lenguaje de programación JAVA, seleccionado anteriormente, implementa a su vez el patrón arquitectónico MVC, es por ello que adoptamos esta arquitectura para el desarrollo de la propuesta de solución. Modelo Vista Controlador (MVC).Es un patrón de diseño de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos.

Modelo: Representa la información con la que trabaja la aplicación, es decir, su lógica de negocio.

Vista: Presenta el modelo en un formato adecuado, como en una página Web que le permite al usuario interactuar con ella, usualmente un elemento de interfaz de usuario.

Controlador: Responde a eventos, usualmente acciones del usuario y realiza los cambios apropiados en el modelo o en la vista.

La arquitectura MVC separa la lógica de negocio (el modelo) y la presentación (la vista) logrando un mantenimiento más rápido y sencillo de las aplicaciones. Ejemplo, para el caso de la web, si se fuera a mostrar una misma aplicación en un navegador estándar, como en un navegador de un dispositivo móvil, sólo es necesario crear una vista nueva por cada dispositivo; manteniendo el controlador y el modelo original. El controlador se encarga de aislar al modelo y a la vista de los detalles del protocolo utilizado para las peticiones

(Aplicación de escritorio, HTTP, consola de comandos, email, etc.). El modelo se encarga de la abstracción de la lógica relacionada con los datos, haciendo que la vista y las acciones sean independientes de, por ejemplo, el tipo de gestor de bases de datos utilizado por la aplicación.

1.6.1 Ventajas del Modelo Vista Controlador

-La separación del Modelo de la Vista, es decir, separa los datos de la representación visual de los mismos.

-Crea independencia de funcionamiento.

-Facilita el mantenimiento en caso de errores.

-Permite el escalamiento de la aplicación en caso de ser requerido.

1.7 Conclusiones

En este capítulo hemos abordados los conceptos fundamentales asociados al dominio del problema, relacionados con el objeto de estudio y el campo de acción. Además realizamos un estudio de lo más utilizado en cuanto a los múltiples lenguajes, metodologías y tecnologías para el desarrollo de aplicaciones existentes, escogiendo así lo que presenta mayor ventaja con respecto a las características de nuestro sistema. Entre lo seleccionado se encuentra JAVA como lenguaje de programación y NetBeans7.0 como IDE de programación; además, se decide realizar la aplicación sobre la base de la metodología ágil XP, pues nos permitirá obtener resultados funcionales observables a corto plazo.

Capítulo 2 Planeación

2.1 Introducción

El presente capítulo tiene como objetivo hacer una valoración de las principales características del sistema a desarrollar. En este se detallan las necesidades del sistema, a través de la descripción de las funcionalidades que serán objeto de automatización. Se hace alusión a la fase de desarrollo de la etapa de planificación de la metodología utilizada, y se muestran los artefactos generados durante el transcurso de la misma.

2.2 Personas relacionadas con el sistema

Persona(s) relacionadas con el sistema	Justificación
Administrador o Usuario	Esta persona es la que tiene cierto conocimiento de cómo se realizan los diferentes tipos de informes de ensayos así como interpretar sus resultados.

Tabla 1Personasrelacionadas con el sistema.

2.3 Planificación

La planificación es la primera fase de la metodología XP, en la misma quedan definidos los procesos que el cliente quiere automatizar y el tiempo en que terminará la construcción la aplicación.

2.4 Historias de Usuarios.

Las historias de usuario (HU) son la técnica utilizada en XP para especificar los requisitos del software. Estos constituyen el resultado directo de la interacción entre los clientes y desarrolladores a través de reuniones donde el flujo de ideas determina no solo los requerimientos del proyecto sino también las posibles soluciones. De forma general se describen brevemente las características que el sistema debe tener desde el punto de vista del cliente. Ver Historias de Usuarios Anexo1

Para definir las historias de usuario se emplea la siguiente plantilla.

Historia de Usuario	
Número: (Número de la Historia de Usuario)	**Usuario:** (Usuario entrevistado para obtener la función requerida a automatizar)
Nombre historia: (Nombre de la historia de usuario que sirve para identificarla mejor entre los desarrolladores y el cliente)	
Prioridad en negocio:(Importancia de la historia para el cliente) (Alta / Media / Baja)	**Riesgo en desarrollo:** (Dificultad para el programador) (Alto / Medio / Bajo)
Puntos estimados:1-3	**Iteración asignada:**(Iteración a la que corresponde)
Programador responsable: Robin Cortina	
Descripción: (Se especifican las operaciones por parte del usuario y las respuestas que dará el sistema)	
Observaciones: (Algunas observaciones de interés, como glosario, información sobre usuarios, etc.)	

Tabla 2 Representación de una historia de usuario.

A continuación se muestra una de las historias de usuarios confeccionadas por el cliente, teniendo estas las funcionalidades principales a integrar en la aplicación.

Historia de Usuario	
Número: 1	**Usuario:** Beatriz Riverón
Nombre historia: Crear informe de Hinchamiento Controlado	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alta
Puntos estimados:3	**Iteraciónasignada:**1
Programador responsable: Robin Cortina Perdomo	
Descripción: El usuario selecciona en el formulario general la opción “Crear Informe de Hinchamiento Controlado”. Luego se muestra un formulario con los campos a llenar y las opciones de calcular, imprimir, graficar y guardar.	
Observaciones: Para poder realizar algunas de las funcionalidades como graficar o calcular es necesario que los campos especificados estén llenos y	

con los tipos de datos requeridos delo contrario se mostraran mensajes indicando el tipo de error.

Tabla 3 H.U. Crear informe de Hinchamiento Controlado

Ver Historias de Usuarios ANEXO 1

2.5 Planificación de entregas

En esta parte se establece la prioridad de cada historia de usuario así como una estimación del esfuerzo necesario de cada una de ellas con el fin de determinar un cronograma de entregas. Las estimaciones de esfuerzo asociado a la implementación de las historias se establecen utilizando como medida el punto. Un punto, equivale a una semana ideal de programación (6 días). Las historias generalmente valen de 1 a 3 puntos. Por otra parte, se mantiene un registro de la "velocidad" de desarrollo, establecida en puntos por iteración, basándose principalmente en la suma de puntos correspondientes a las historias de usuario que fueron terminadas en la última iteración. La planificación se puede realizar basándose en el tiempo o el alcance. La velocidad del proyecto es utilizada para establecer cuántas historias se pueden implementar antes de una fecha determinada o cuánto tiempo tomará implementar un conjunto de historias. Al planificar por tiempo, se multiplica el número de iteraciones por la velocidad del proyecto, determinándose cuántos puntos se pueden completar. Al planificar según alcance del sistema, se divide la suma de puntos de las historias de usuario seleccionadas entre la velocidad del proyecto, obteniendo el número de iteraciones necesarias para su implementación.

2.5.1 Estimación de esfuerzos por historias de usuario

Historia de Usuario	Numero	Puntos Estimados
1.Crear informe de Hinchamiento Controlado	**1**	**3**
2.Actualizar informe de Hinchamiento Controlado	**2**	**1**
3.Eliminar informe de Hinchamiento	**3**	**1**

Controlado		
4.Listado de informes de Hinchamiento Controlado	**4**	**1**
5.Buscar informe de Hinchamiento Controlado	**5**	**1**
6.Crear informe de Límites de Contracción	**6**	**3**
7.Actualizar informe de Límites de Contracción	**7**	**1**
8.Eliminar informe de Límites de Contracción	**8**	**1**
9.Listar informe de Límites de Contracción	**9**	**1**
10.Buscar informe de Límites de Contracción	**10**	**1**
11.Crear Usuario	**11**	**1**
12.Modificar Usuario	**12**	**1**
13. Listado de Usuarios	**13**	**1**
14. Eliminar Usuario	**14**	**1**

Tabla 4 Estimación de esfuerzo por historias de usuarios

El plan de entregas se realiza teniendo en cuenta las unidades funcionales que se quieren entregar y cada uno de estos módulos abarca un número de historias de usuarios a implementar para dar cumplimiento al funcionamiento del mismo.

2.5.2 Plan de iteraciones

Modulo	Historia(s) de Usuario que abarca
Hinchamiento Controlado Administrar Sistema	1,2,3,4,5 11,12,13,14
Límites de Contracción	6, 7, 8, 9, 10

Tabla 5 Plan de iteraciones

Iteración #1 Hinchamiento Controlado y Administrar Sistema: En esta primera iteración se realizan todas las funcionalidades referentes al cálculo de Hinchamiento Controlado así como la administración del sistema.

Iteración #2: En esta iteración se realizan todas las funcionalidades referentes al cálculo de los Límites de Contracción.

Además en estas iteraciones se realizan las consultas a la base de datos como Insertar, Eliminar, Buscar, Actualizar y Listar los informes de los diferentes tipos de ensayos.

2.5.3 Plan de duración de las iteraciones

Como parte del proceso de desarrollo de un proyecto guiado por la metodología XP, se crea el plan de duración de cada una de las iteraciones que se llevarán a cabo durante el desarrollo del mismo. Este plan tiene como finalidad mostrar la duración de cada iteración, así como el orden en que serán implementadas las HU en cada una de las mismas.

Iteración	Orden de implementación por Historias de Usuario	Duración de la iteración
1	Crear Hinchamiento Controlado(1) Actualizar Hinchamiento Controlado(2) Eliminar Hinchamiento Controlado(3) Listar Hinchamiento Controlado(4) Buscar Hinchamiento Controlado(5) Crear Usuario(11) Modificar Usuario(12) Eliminar Usuario(13) Listar Usuario(14)	11Semanas
2	Límites de Contracción(6) Actualizar Límites de Contracción(7) Eliminar Límites de Contracción(8) Listar Límites de Contracción(9) Buscar Límites de Contracción(10)	7 Semanas

Tabla 6 Duración de las iteraciones

Combinando el plan de entrega y el plan de iteraciones se harán raleases o liberaciones al sistema en las fechas mostradas a continuación:

Iteración/Modulo	Hinchamiento Controlado, Administrar Sistema	Límites de Contracción
Final 1ra Iteración	15/04/2012	
Final 2a Iteración		15/06/2012

Tabla 7 Tabla de releases

2.6 Conclusiones

En este capítulo se abordó la fase de planeación donde se presentaron las distintas HU con la participación del cliente, se llevó a efecto la planificación de iteraciones de cada HU a partir de la estimación del esfuerzo necesario de las mismas, culminando así esta fase y se determina que el equipo de trabajo está listo para pasar a la siguiente etapa de desarrollo.

Capítulo 3: Diseño e Implementación.

3.1 Introducción.

En este capítulo se construye la solución propuesta de forma iterativa, tal y como indica la metodología XP. Una iteración es una entrega del proyecto al cliente que está incompleta, pero tiene implementadas algunas funcionalidades necesarias en ese instante. La siguiente iteración es otra entrega con alguna funcionalidad más y así sucesivamente hasta llegar a la iteración final, en la que se entrega el software acabado. En este proyecto vale destacar que el desarrollo de las iteraciones ha sido ajustado de forma tal que al final de cada una de ellas, se tenga un módulo amplio del sistema. Uno de los artefactos principales es la creación de las tarjetas CRC (Class, Responsabilities and Collaboration) las cuales permiten brindar un mayor enfoque orientado a objetos. Por otra parte se describen cada una de las tareas confeccionadas para llevar cumplir con el desarrollo de cada una de las historias de usuario detectadas.

3.1Diseño

3.1.1 Modelo Vista Controlador (MVC) en Java Swing

Java tiene como características fundamentales su portabilidad a un gran número de plataformas (Java es en sí una plataforma), su simplicidad y su extenso conjunto de librerías. Su arquitectura está profundamente basada en MVC, lo que proporciona un alto grado de extensibilidad y de personalización de los componentes de la librería. Permite “conectar” y “desconectar” estilos de interfaz de usuario (llamados “look and feels”) que modifican la forma en que se muestra y se comporta toda la interfaz de usuario, así, la misma aplicación puede verse como una aplicación Windows o como una aplicación Motif7 simplemente conmutando el look and feel en tiempo de ejecución.

3.1.2 Ventajas

La implementación de Swing del patrón de diseño MVC presenta muchas ventajas:

- Permite la creación de interfaces de usuario de una manera sencilla y rápida, permitiendo el manejo del patrón MVC pero ocultando los detalles de su implementación.

- El mecanismo de eventos de Java se adapta perfectamente al mecanismo de notificaciones de MVC. Al estar los modelos separados de la vista, las posibilidades de extensión de la librería y de personalización de componentes ya existentes son enormes.

- Permite al usuario crear sus propias estructuras de datos y adaptar la interfaz de usuario a ellas y no a la inversa, como sucede con librerías ya implementadas.

3.2 Tarjetas CRC

El uso de las tarjetas C.R.C (Class, Responsabilities and Collaboration) permiten al programador centrarse y apreciar el desarrollo orientado a objetos olvidándose de los malos hábitos de la programación procedural clásica. Las tarjetas C.R.C representan objetos; la clase a la que pertenece el objeto se puede escribir en la parte de arriba de la tarjeta, en una columna a la izquierda se pueden escribir las responsabilidades u objetivos que debe cumplir el objeto y a la derecha, las clases que colaboran con cada responsabilidad. Esta nueva técnica de diseño es adoptada como alternativa a los diagramas UML de las clases, pues en estas se plasman las responsabilidades que tienen cada objeto y las clases con las que tienen que interactuar para darles respuesta brindando así la información que se necesita a la hora de implementar.

Clase	
Description:	
Attributes:	
Name	Description
Responsibilities:	
Name	Collaborator

Tabla8: Plantilla de Tarjeta CRC.

A continuación se muestra una de las tarjetas CRC teniendo estas las funcionalidades principales a integrar en la aplicación.

Datos_Generales	
Description: Esta es la clase que va a almacenar los datos generales del ensayo,independientemente del tipo de experimento que se vaya a realizar	
Attributes:	
Name	Description
codigo	Llave primaria y codigo de la obra
cala	Numero de la cala
muestra	Este es el numero de la muestra a ensayar
registro	Registro de la obra
norma	De acuerdo al tipo de ensayo es la norma a utilizar
referencia	Referencia
descripcion	Una breve descripcion de la propiedades de la muestra
obra	Nombre de la obra que se desea realizar
fecha	Fecha en que se realiza el ensayo
nombre	Nombre de tipo de ensayo
profundidadI	Profundidad a la que se tomo la primera muestra
profundidadF	Profundidad a la que se tomo la ultima muestra
Responsibilities:	
Name	Collaborator
Insertar_Datos_Generales()	
Modificar_Datos_generales()	
Eliminar_Datos_Generales()	
Listar_Datos_Generales()	

Ver tarjetas CRC ANEXO 2

Para una mejor comprensión de las tarjetas C.R.C de nuestro sistema procedemos a agruparlas por los módulos identificados en el Plan de Entregas en el capítulo 2.

3.3 Implementación.

En la metodología XP se convierte en un integrante más del equipo de desarrollo el cliente pues él crea las historias de usuario bajo la supervisión de los desarrolladores. Estas historias quedan confeccionadas cuando el cliente es capaz de identificar con precisión la funcionalidad deseada, además, también debe estar presente cuando se realicen las pruebas de aceptación para cada historia, por lo que su presencia es imprescindible.

En XP generalmente cada historia de usuario se divide en tareas de ingeniería (TI) o tareas de programación. Estas se crean para obtener una mejor

planificación de la historia; estas pretenden cumplir con las funcionalidades básicas que luego conformaran las funcionalidades generales de cada historia.

A continuación se presentan las Tareas de Ingeniería agrupadas por las respectivas historias de usuario a las que pertenecen.

Tareas	
Número de la Tarea:1	**Número de Historia:**1
Nombre de la Tarea: Calcular Hinchamiento Controlado.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:**0.5
Fecha de Inicio: 30/enero/2012.	**Fecha Fin:** 2/febrero/2012.
Programador Responsable: Robin Cortina Perdomo	
Descripción: Se crea un Informe de Hinchamiento Controlado en el cual se introducen los datos y luego se calcula el hinchamiento de la muestra.	

Ver las tareas ANEXO 3

3.4 Conclusiones

Enfocándose en la programación orientada a objetos dentro de la fase de diseño de la metodología XP, se crearon las tarjetas CRC. Para lograr la completa implementación de cada historia de usuario en la fecha acordada con el cliente, estas se dividieron en tareas de ingeniería. A cada TI se le asignó un tiempo de desarrollo que se cumplió de manera eficiente garantizando así el objetivo principal de su confección.

Capítulo 4 Prueba

4.1 Introducción

En el presente capítulo se muestran las pruebas de aceptación confeccionadas por el cliente para comprobar que la aplicación funcione de forma correcta. Estas pruebas fueron llevadas a cabo antes de cada entrega que se realizó durante todo el desarrollo del proyecto.

4.2 Pruebas de Software.

Las pruebas de software son elemento crítico para la garantía de calidad del software y representa una revisión final de las especificaciones, del diseño y de la codificación. La creciente percepción del software como un elemento del sistema y la importancia de los costos asociados a un fallo del propio sistema, están motivando la creación de pruebas minuciosas y bien planificadas. No es raro que una organización de desarrollo de software emplee entre el treinta y cuarenta por ciento del esfuerzo total del proyecto en las pruebas. Una vez generado el código fuente, el software debe ser probado para descubrir y corregir el máximo de errores posibles antes de su entrega al cliente. El objetivo es diseñar una serie de casos de prueba que tengan una alta probabilidad de encontrar errores. Aquí es donde se aplican las técnicas de pruebas del software. Estas técnicas facilitan una guía sistemática para diseñar pruebas que:

- Comprueban la lógica interna de los componentes software.
- Verifican los dominios de entrada y salida del programa para descubrir errores en la funcionalidad, el comportamiento y rendimiento (Pressman, 2002).

En la metodología XP es esencial el desarrollo de las pruebas, permitiendo probar constantemente el código. Cada vez que se quiere implementar las funcionalidades que tendrá el software, XP propone una redacción sencilla de prueba, para ser pasada por el código posteriormente. XP posee dos tipos de pruebas; las unitarias o TDD (desarrollo dirigido por pruebas, del inglés Test Driven Development) desarrolladas por los programadores verificando su código de forma automática y las pruebas de aceptación que son las que se utilizaran en este capítulo, dichas pruebas serán evaluadas luego de culminar

una iteración verificando así si se cumplió la funcionalidad requerida por el cliente.

4.2.1 Desarrollo Dirigido por Pruebas

El desarrollo dirigido por pruebas (TDD), se enfoca en la implementación orientada apruebas. El código debe ser probado paso a paso y obtener un resultado, aunque aún no con lógica para el negocio, pero sí funcional. Algunas personas confunden este término con las nombradas “pruebas de caja blanca”, las cuales son pruebas que se realizan a los métodos u operaciones para medir la funcionalidad del mismo desde la perspectiva de la validez para el cliente. Sin embargo el TDD se aplica antes de comenzar a implementar cada paso de la tarea en desarrollo asumiendo que la prueba es insatisfactoria desde un inicio. Solo una vez que se haya cumplido de la forma más sencilla posible la lógica del código a probar se asume como cumplida. Luego se realiza un proceso conocido informalmente como “refactorización” de código, el cual consiste en limpiarlo, organizarlo y adaptarlo a los patrones. En esencia, TDD se centra en la lógica del código y las pruebas de caja blanca en la del negocio.

4.2.2 Pruebas de Aceptación

Las pruebas de aceptación en la metodología XP, se pueden asociar con las pruebas de caja negra que se aplican en otras metodologías de desarrollo, sólo que en XP se crean a partir de las Historias de Usuario y no por un listado de requerimientos. Durante las iteraciones las Historias de Usuarios seleccionadas serán traducidas a pruebas de aceptación. En ellas se especifican, desde la perspectiva del cliente, los escenarios para probar que una Historia de Usuario ha sido implementada correctamente. Una Historia de Usuario puede tener todas las pruebas de aceptación que necesite para asegurar su correcto funcionamiento. El objetivo final de éstas es garantizar que las funcionalidades requeridas por el cliente han sido cumplidas. Una Historia de Usuario no se considera completa hasta que no ha pasado por sus pruebas de aceptación.

4.3 Objetivos de las pruebas

El objetivo de las pruebas, expresado de forma sencilla, es encontrar el mayor número posible de errores con una cantidad razonable de esfuerzo, aplicado

sobre un lapso de tiempo realista. Se debe ejecutar el programa antes de que llegue al cliente, con la intención especifica de descubrir todos los errores, de manera que el cliente no experimente la frustración asociada con un producto de baja calidad. Con el propósito de encontrar el mayor número posible de errores, las pruebas deben conducirse sistemáticamente.

Otro de sus objetivos son que:

- Un buen caso de prueba es aquel que tiene una alta probabilidad de mostrar un error no descubierto hasta entonces.
- Una prueba tiene éxito si descubre un error no detectado hasta entonces.
- Reducir costos de mantenimiento.
- Obtener información concreta acerca de fallas, que pueda usarse como apoyo en la mejora de procesos, y en la de los desarrolladores. Los objetivos anteriores suponen un cambio dramático de punto de vista. Nos quitan la idea que, normalmente, tenemos de que una prueba tiene éxito si no descubre errores. Nuestro objetivo es diseñar pruebas que sistemáticamente saquen a la luz diferentes clases de errores, haciéndolo con la menor cantidad de tiempo y de esfuerzo. Si la prueba se lleva a cabo con éxito (de acuerdo con el objetivo anteriormente establecido), descubrirá errores en el software. Como ventaja secundaria, la prueba demuestra hasta qué punto las funciones del software parecen funcionar de acuerdo con las especificaciones y parecen alcanzarse los requisitos de rendimiento.

4.4 Alcance de las pruebas

La prueba de software tiene limitantes, tanto teóricos como prácticos. Desde el punto de vista teórico, la prueba es un problema que llamamos no-decidible; esto implica, de algún modo, que no podemos escribir un programa que pruebe los programas sin intervención humana. Sin embargo, como mencionábamos anteriormente, la prueba sí es automatizable en muchos aspectos. Desde el punto de vista práctico, la cantidad de posibilidades para probar exhaustivamente un sistema es sencillamente inmanejable; es necesario entonces utilizar técnicas adecuadas para maximizar la cantidad de fallas importantes encontradas con los recursos asignados. Cada método que se utilice para detectar defectos deja un residuo de defectos más sutiles contra los cuales ese método es ineficaz (la llamada “Paradoja del Pesticida”). La prueba

de software implica pues, la aplicación de técnicas y herramientas apropiadas en el marco de un proceso bien definido, determinado por el tipo de proyectos de desarrollo de software que se abordan.

4.5 Plantilla de Prueba de Aceptación

La plantilla de Prueba de Aceptación, se genera de la etapa de pruebas. El objetivo de las pruebas de aceptación es validar que un sistema cumple con el funcionamiento esperado y permitir al usuario de dicho sistema que determine su aceptación, desde el punto de vista de su funcionalidad y rendimiento.

Para realizar las pruebas de aceptación el cliente utiliza la siguiente plantilla.

Prueba de aceptación
HU: Nombre de la historia de usuario que va a comprobar su funcionamiento.
Nombre: Nombre del caso de prueba.
Descripción: Descripción del propósito de la prueba.
Condiciones de ejecución: Precondiciones para que la prueba se pueda realizar.
Entrada/Pasos de ejecución: Pasos para probar la funcionalidad.
Resultado esperado: Resultado que se desea de la prueba
Evaluación de la prueba: Aceptada o Denegada.

Tabla Plantilla de Pruebas de Aceptación

Ver Tablas de Pruebas ANEXO 4

4.6 Conclusiones

Con la realización de las pruebas de aceptación el cliente se asegura de que las funciones implementadas cumplan su objetivo satisfactoriamente, probando individualmente cada módulo y asignándole la evaluación correspondiente. Todas las pruebas que se realizaron fueron positivas y el cliente estuvo conforme, cumpliendo entonces la aplicación con las historias de usuarios definidas inicialmente.

Capítulo 5 Estudio de Factibilidad

Introducción

En la actualidad con el desarrollo de los proyectos viene incluido el estudio de factibilidad del sistema, el cual es vital pues se tienen en cuenta los costos a incurrir, deduciéndose si el proyecto realizado será factible o no llevarlo a cabo. Hay muchas formas de calcular el costo, pero para nuestro caso se utilizará la Metodología Costo Efectividad, la cual sugiere que la conveniencia de la ejecución de un proyecto se determina por la observación de ciertos factores en conjunto, estos son:

- El costo que involucra la implementación de la solución informática, adquisición y puesta en marcha del sistema hardware / software y los costos de operación asociados.
- La efectividad que se entiende como capacidad del proyecto para satisfacer la necesidad, solucionar el problema o lograr el objetivo por el cual se ideó, es decir, un proyecto será más o menos efectivo con relación al mayor o menor cumplimiento que alcance en la finalidad para la cual fue ideado (costo por unidad del cumplimiento del objetivo).

Esta parte es fundamental en la elaboración de cualquier proyecto pues haciendo un estudio correcto de factibilidad se puede ahorrar semanas, meses e incluso años de trabajo, hasta evitar poner en duda la reputación profesional si se realiza un sistema mal planificado desde una etapa temprana. (Ficha de costo ,2000)

5.1 Efectos Económicos

-Efectos directos

-Efectos indirectos

-Efectos externos

-Intangibles

5.1.1 Efectos directos:

- POSITIVOS:

-Se facilitará la realización de informes de ensayos

-Se facilitará la interpretación de los resultados mediante la graficación.

-Se mejorará la eficiencia y explotación del Sistema de Gestión de Informes de Ensayos (SIGIE 1.0).

- NEGATIVOS:

Para usar la aplicación es imprescindible el uso de un ordenador, aparejado a los gastos que este trae de consumo de corriente eléctrica y mantenimiento.

5.1.2 Efecto indirecto:

Los efectos económicos observados que pudiera repercutir sobre otros mercados no son perceptibles, aunque este proyecto no está construido con la finalidad de comercializarse.

5.1.3 Intangibles:

En la valoración económica siempre hay elementos perceptibles por una comunidad como perjuicio o beneficio, pero al momento de ponderar en unidades monetarias esto resulta difícil o prácticamente imposible. A fin de medir con precisión los efectos, deberán considerarse tres situaciones:

5.1.3.1 Situación sin proyecto

Los informes de ensayos se realizan en hoja de cálculo de Excel o de forma manual, realizando los cálculos y la graficación en un período de tiempo excesivamente largo y muchas veces perdiendo la información por no contar con una base de datos potente.

5.1.3.2 Situación con proyecto

Con la realización del Sistema de Gestión de Informes de Ensayos SIGIE 1.0 se ha logrado disminuir el tiempo de realización de un informe de ensayo además mejora el trabajo de los especialistas en ensayos debido a que esta herramienta facilita los cálculos y la graficación del experimento.

5.2 Beneficios Y Costos Intangibles en el proyecto

Costos:

-Resistencia al cambio.

Beneficios:

-Mayor comodidad para los usuarios.

-Mejora en la calidad de la información.

-Menor tiempo empleado en la introducción de los datos.

-Facilidad a la hora de interpretar los resultados.

5.3 Ficha de costo

Para determinar el costo económico del proyecto se utilizará el procedimiento para elaborar Una Ficha De Costo de un Producto Informático [Dra. Ana Ma. Gracia Pérez, UCLV]. Para la elaboración de la ficha se consideran los siguientes elementos de costo, desglosados en moneda libremente convertible y moneda nacional.

5.3.1Costos en Moneda Libremente Convertible:

-Costos Directos.

1. Compra de equipos de cómputo: No procede.
2. Alquiler de equipos de cómputo: No procede.
3. Compra de licencia de Software: No procede.
4. Depreciación de equipos: $ 50.00.
5. Materiales directos: No procede.

Total: $ 50.00.

- Costos Indirectos.

1. Formación del personal que elabora el proyecto: No procede.
2. Gastos en llamadas telefónicas: No procede.
3. Gastos para el mantenimiento del centro: No procede.
4. Know How: No procede.
5. Gastos en representación: No procede.

Total: $0.00.

- Gastos de distribución y venta.

1. Participación en ferias o exposiciones: No procede.
2. Gastos en transportación: No procede.
3. Compra de materiales de propagandas: No procede.

Total: $0.00.

5.3.2 Costos en Moneda Nacional:

- Costos Directos.

1. Salario del personal que laborará en el proyecto: $100.00.

2. El 12% del total de gastos por salarios se dedica a la seguridad social: No procede.

3. El 0.09% de salario total, por concepto de vacaciones a acumular: No procede.

4. Gasto por consumo de energía eléctrica: $ 5.50.

5. Gastos en llamadas telefónicas: No procede.

6. Gastos administrativos: No procede.

- Costos Indirectos.

1. Know How: No procede.

Total: $ 105.50.

Como se hizo referencia anteriormente, la técnica seleccionada para evaluar la factibilidad del proyecto es la Metodología Costo-Efectividad. Dentro de esta metodología la técnica de punto de equilibrio aplicable a proyectos donde los beneficios tangibles no son evidentes el análisis se basa exclusivamente en los costos. Para esta técnica es imprescindible definir una variable discreta que haga variar los costos. Teniendo en cuenta que el costo para este proyecto es despreciable, tomaremos como costo el tiempo en minutos empleado desde que se introducen los datos hasta el tiempo en que se muestran los resultados y el grafico de comportamiento.

Valores de la variable (Solución manual):

- Gestionar los datos de Hinchamiento Controlado por la vicedecana de la facultad de Geología en los laboratorios de Mecánica de Suelo y Rocas del ISMMM (60 min).
- Calcular los ensayos e interpretar los resultados de este proceso (30 min).
- Gestionar los datos de Límites de Contracción por la vicedecana de la facultad de Geología en los laboratorios de Mecánica de Suelos y Rocas del ISMMM (50 min).
- Calcular los ensayos de este proceso (20 min).

Valores de la variable (Solución con el software):

- Gestionar los datos de Hinchamiento Controlado por la vicedecana de la facultad de Geología en los laboratorios de Mecánica de Suelo y Rocas del ISMMM (12 min).
- Calcular los ensayos e interpretar los resultados de este proceso (4 min).
- Gestionar los datos de Límites de Contracción por la vicedecana de la facultad de Geología en los laboratorios de Mecánica de Suelos y Rocas del ISMMM (10 min).
- Calcular los ensayos de este proceso (3 min).

A continuación se muestra un gráfico que muestra la solución de las actividades sin proyecto comparado con la solución con proyecto.

Tiempo de la realización de las actividades

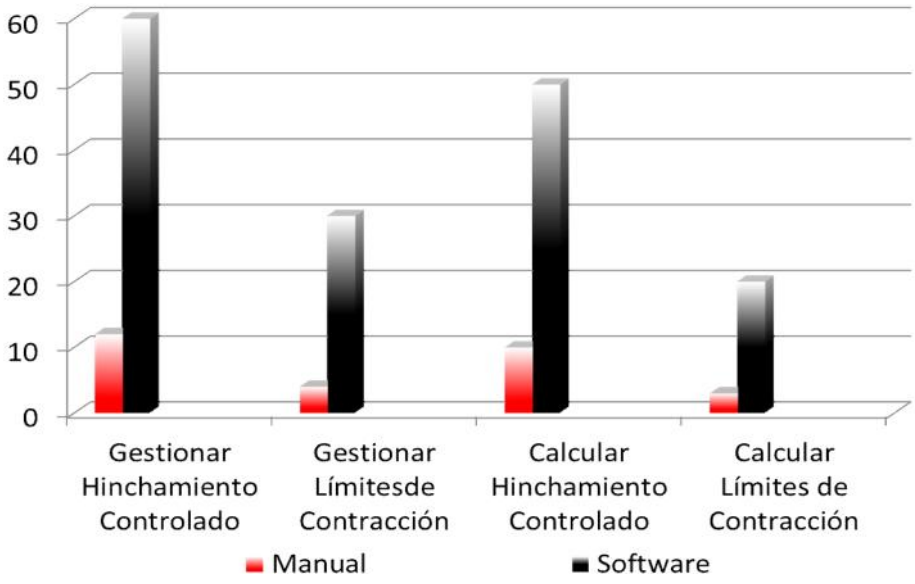


Fig. Gráfica de la solución sin el producto y solución con el producto.

5.4 Conclusiones

En este capítulo se realizó el estudio de factibilidad mediante La Metodología Costo Efectividad (Beneficio), se analizó los efectos económicos, los beneficios y costos intangibles, así como se calculó el costo de ejecución del proyecto mediante la ficha de costo arrojando como resultado $50.00 CUC y $105.50 CUP demostrándose la factibilidad del proyecto.

Conclusiones Generales

Durante la realización de este trabajo se dio cumplimiento a las tareas concebidas para el desarrollo de la aplicación dando así desempeño al objetivo general del trabajo. Se hizo un análisis detallado de las diferentes metodologías para la elaboración del sistema, teniendo como resultado el uso de la metodología XP en las que se les dio cumplimiento a las Historias de Usuarios del sistema, se realizaron las tarjetas CRC, las tareas de ingenierías para cada Historia de Usuario y las pruebas de aceptación realizadas por el cliente. Luego del análisis de factibilidad podemos concluir que el trabajo responde como propuesta de solución al problema planteado. A demás el sistema se encuentra disponible para ser usado.

Recomendaciones

-Se recomienda la explotación de las funcionalidades que brinda el software en los Laboratorios de Mecánica de Suelo y Rocas del Instituto Superior Minero Metalúrgico de Moa.

-Que se implementen más funcionalidades al sistema además de crear una nueva versión la cual abarque los diferentes tipos de informes que se realizan en el Instituto Superior Minero Metalúrgico de Moa (ISMMM).

Bibliografías

C++. [en línea], 2004. [Consultado: 2012-03-20]. Disponible en: http://wikipedia.uo.edu.cu/es/articles/c/+/+/C++.html.

Delphi. [en línea], 2004. [Consultado: 2012-03-20]. Disponible en:

http://wikipedia.uo.edu.cu/es/articles/d/e/l/Delphi.html.

http://es.wikipedia.org/w/index.php?title=PostgreSQL&oldid=54219210

Eaprende.com Aprende.com. [En línea] // Gestor de Base de Datos: MySQL, PostgreSQL, SQLite. - 2001. - enero de 2011. http://www.eaprende.com/gestor-de-basededatos-mysql-postresql sqlite.html.

Sitio oficial metodología XP http://www.extremeprogramming.org (Consultado: 2/febrero/2012).

Java. [en línea], 2004. [Consultado: 2012-03-20]. Disponible en: http://wikipedia.uo.edu.cu/es/articles/j/a/v/Java.html.

Jeffries, Canos Letelier Penades, Ferrer, C. “Extreme Programming”, AddisonWesley.2001.

Monografias.com Monografias.com [En línea] // Definición arquitectura cliente servidor. -2007. - enero de 2011.

http://www.monografias.com/trabajos24/arquitectura cliente-servidor/arquitectura-cliente-servidor.shtml.

Roger S. Pressman. Un enfoque práctico. Ciudad Madrid, editorial Mc Graw Hill, 2002. 640 páginas

Sitio oficial de MySQL http://www.mysql.org (Consultado: 2/febrero/2010).

SUBIRÓS MUÑOZ, Dariel Raúl. Desarrollo de una interfaz gráfica de usuario para el preprocesador meteorológico AERMET. Trabajo de Diploma. Instituto Superior Minero Metalúrgico de Moa “Dr. Antonio Núñez Jiménez”, 2009.

Wikipedia.org Wikipedia, la enciclopedia libre [En línea]. - marzo de 2012. - http://www.eswikipedia.org.

ZULUETA TORRES, Agustín. Modulo para la Extracción, Preprocesamiento, Descripción y Almacenaje en formato XML, de la información recuperada por el Sistema Automatizado de Información Virtual del ISMMM. Trabajo de Diploma. Instituto Superior Minero Metalúrgico de Moa “Dr. Antonio Núñez Jiménez”, 2010.

Glosario de términos

Bugs: Errores presentas en la aplicación que atentan contra su correcto funcionamiento

Herramientas: Son los ambientes de apoyo necesario para automatizar las prácticas de Ingeniería de Software.

IDE: Integrated **D**evelopment **E**nvironment / Entorno de Desarrollo Integrado. Entorno de programación que ha sido empaquetado como un programa de aplicación, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica GUI.

Iteraciones: En el contexto de un proyecto se refieren a la técnica de desarrollar y entregar componentes incrementales de funcionalidades de un negocio. Una iteración resulta en uno o más paquetes atómicos y completos del trabajo del proyecto que pueda realizar alguna función tangible del negocio. Múltiples iteraciones contribuyen a crear un producto completamente integrado.

Metodología Ágil: Constituyen un nuevo enfoque en el desarrollo de software, mejor aceptado por los desarrolladores de proyectos que las metodologías convencionales debido a la simplicidad de sus reglas y prácticas, su orientación a equipos de desarrollo de pequeño tamaño, su flexibilidad ante los cambios y su ideología de colaboración.

Metodologías de Desarrollo: Se define como un conjunto de filosofías, etapas, procedimientos, reglas, técnicas, herramientas, documentación y aspectos de formación para los desarrolladores de sistemas de información.

Multiplataforma: Es un término usado para referirse a los programas, sistemas operativos, lenguajes de programación, u otra clase de software, que puedan funcionar en diversas plataformas.

Programación Extrema(XP): Es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo.

Release: Versión candidata definitiva de un producto de software y se refiere a un producto final, preparado para lanzarse como versión definitiva a menos que aparezcan errores que lo impidan.

RUP: El Proceso Unificado Racional o RUP (Rational Unified Process), es un proceso desarrollo de software y junto con el Lenguaje Unificado de Modelado UML, constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos. Se caracteriza por ser iterativo e incremental, estar centrado en la arquitectura y guiado por los casos de uso. Incluye artefactos y roles.

Software: Es la suma total de los programas de cómputo, procedimientos, reglas, documentación y datos asociados que forman parte de las operaciones de un sistema de cómputo.

Software Libre: Es el software que, una vez obtenido, puede ser usado, copiado, estudiado, modificado y redistribuido libremente, aunque conserve su carácter de libre, puede ser vendido comercialmente.

Testing: Proceso de pruebas usado para identificar posibles fallos de implementación, calidad, o usabilidad de un programa de ordenador o videojuego. Básicamente es una fase en el desarrollo de software consistente en probar las
aplicaciones construidas.

Historias de Usuarios

Historia de Usuario	
Número: 1	**Usuario:** Beatriz Riverón
Nombre historia: Crear informe de Hinchamiento Controlado	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alta
Puntos estimados: 3	**Iteración asignada:** 1
Programador responsable: Robin Cortina Perdomo	
Descripción: El usuario selecciona en el formulario general la opción "Crear Informe de Hinchamiento Controlado". Luego se muestra un formulario con los campos a llenar y las opciones de calcular, imprimir, graficar y guardar.	
Observaciones: Para poder realizar algunas de las funcionalidades como graficar o calcular es necesario que los campos especificados estén llenos y con los tipos de datos requeridos de lo contrario se mostraran mensajes indicando el tipo de error.	

Tabla Historia de Usuario #1 Crear Informe de Hinchamiento Controlado

Historia de Usuario	
Número: 2	**Usuario:** Beatriz Riverón
Nombre historia: Actualizar Informe de Hinchamiento Controlado.	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Media
Puntos estimados: 1	**Iteración asignada:** 1
Programador responsable: Robin Cortina Perdomo	
Descripción: El usuario selecciona en el formulario general la opción "Modificar Informe de Hinchamiento Controlado". Primeramente se muestra un formulario en el que el usuario debe especificar el informe a modificar, luego se muestra un formulario con los campos a llenar y las opciones de calcular, imprimir, graficar y guardar.	
Observaciones: Para poder modificar un informe primero se debe verificar que el informe este en la base de datos de no estar se muestra un mensaje especificando que no existe.	

Tabla Historia de Usuario #2 Actualizar Informe de Hinchamiento Controlado

Historia de Usuario	
Número: 3	**Usuario:** Beatriz Riverón
Nombre historia: Listar Informes de Hinchamiento Controlado.	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Baja
Puntos estimados: 1	**Iteración asignada:** 1
Programador responsable: Robin Cortina Perdomo	
Descripción: El usuario selecciona en el formulario general la opción "Listado de Informes de Hinchamiento Controlado". Luego se muestra un formulario con la lista de informes que se encuentran guardados en la base de datos.	

Tabla Historia de Usuario #3 Listar Informe de Hinchamiento Controlado

Historia de Usuario	
Número: 4	**Usuario:** Beatriz Riverón
Nombre historia: Eliminar Informe de Hinchamiento Controlado.	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Media
Puntos estimados: 1	**Iteración asignada:** 1
Programador responsable: Robin Cortina Perdomo	
Descripción: El usuario selecciona en el formulario general la opción "Eliminar Informe de Hinchamiento Controlado". Luego se muestra un formulario con la lista de informes que se encuentran guardados en la base de datos, el usuario selecciona el informe a eliminar y presiona la opción de eliminar.	
Observaciones: Para eliminar un informe se selecciona y se acepta luego se muestra un mensaje de confirmación para la eliminación segura.	

Tabla Historia de Usuario #4 Eliminar Informe de Hinchamiento Controlado

Historia de Usuario	
Número: 5	**Usuario:** Beatriz Riverón
Nombre historia: Buscar Informe de Hinchamiento Controlado.	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Baja
Puntos estimados: 1	**Iteración asignada:** 1
Programador responsable: Robin Cortina Perdomo	
Descripción: El usuario selecciona en el formulario general la opción "Buscar Informe de Hinchamiento Controlado". Se muestra un formulario con una lista de informes, el usuario selecciona el informe a buscar y se muestra los datos generales de este informe.	

Tabla Historia de Usuario #6 Buscar Informe de Hinchamiento Controlado

Historia de Usuario	
Número: 6	**Usuario:** Beatriz Riverón
Nombre historia: Crear Informe de Límites de Contracción	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alta
Puntos estimados: 3	**Iteración asignada:** 2
Programador responsable: Robin Cortina Perdomo	
Descripción: El usuario selecciona en el formulario general la opción "Crear Informe de Límites de Contracción". Luego se muestra un formulario con los campos a llenar y las opciones de calcular, imprimir y guardar.	
Observaciones: Para poder realizar algunas de las funcionalidades como graficar o calcular es necesario que los campos especificados estén llenos y con los tipos de datos requeridos de lo contrario se mostraran mensajes indicando el tipo de error.	

Tabla Historia de Usuario #6 Crear Informe de Límites de Contracción

Historia de Usuario	
Número: 7	**Usuario:** Beatriz Riverón
Nombre historia: Actualizar Informe de Límites de Contracción	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alta
Puntos estimados: 1	**Iteración asignada:** 2
Programador responsable: Robin Cortina Perdomo	
Descripción: El usuario selecciona en el formulario general la opción "Modificar Informe de Límites de Contracción". Primeramente se muestra un formulario en el que el usuario debe especificar el informe a modificar, luego se muestra un formulario con los campos a llenar y las opciones de calcular, imprimir y guardar.	
Observaciones: Para poder modificar un informe primero se debe verificar que el informe este en la base de datos de no estar se muestra un mensaje especificando que no existe.	

Tabla Historia de Usuario #7 Actualizar Informe de Límites de Contracción

Historia de Usuario	
Número: 8	**Usuario:** Beatriz Riverón
Nombre historia: Listar Informes de Límites de Contracción.	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Baja
Puntos estimados: 1	**Iteración asignada:** 2
Programador responsable: Robin Cortina Perdomo	
Descripción: El usuario selecciona en el formulario general la opción "Listado de Informes de Límites de Contracción". Luego se muestra un formulario con la lista de informes que se encuentran guardados en la base de datos.	

Tabla Historia de Usuario #8 Listar Informe de Límites de Contracción

Historia de Usuario	
Número: 9	**Usuario:** Beatriz Riverón
Nombre historia: Eliminar Informe de Límites de Contracción.	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Media
Puntos estimados: 1	**Iteración asignada:** 2
Programador responsable: Robin Cortina Perdomo	
Descripción: El usuario selecciona en el formulario general la opción "Eliminar Informe de Límites de Contracción". Luego se muestra un formulario con la lista de informes que se encuentran guardados en la base de datos, el usuario selecciona el informe a eliminar y presiona la opción de eliminar.	
Observaciones: Para eliminar un informe se selecciona y se acepta luego se muestra un mensaje de confirmación para la eliminación segura.	

Tabla Historia de Usuario #9 Eliminar Informe de Límites de Contracción

Historia de Usuario	
Número: 10	**Usuario:** Beatriz Riverón
Nombre historia: Buscar Informe de Límites de Contracción.	
Prioridad en negocio: Baja	**Riesgo en desarrollo:** Baja
Puntos estimados: 1	**Iteración asignada:** 2
Programador responsable: Robin Cortina Perdomo	
Descripción: El usuario selecciona en el formulario general la opción "Buscar Informe de Límites de Contracción". Se muestra un formulario con una lista de informes, el usuario selecciona el informe a buscar y se muestra los datos generales de este informe.	

Tabla Historia de Usuario #10 Buscar Informe de Límites de Contracción

Historia de Usuario	
Número: 11	**Usuario:** Beatriz Riverón
Nombre historia: Crear Usuario	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Baja
Puntos estimados: 1	**Iteración asignada:** 1
Programador responsable: Robin Cortina Perdomo	
Descripción: El Administrador selecciona en el formulario general la opción "Crear Usuario". Luego se muestra un formulario con los campos a llenar	

Tabla Historia de Usuario #11 Crear Usuario

Historia de Usuario	
Número: 12	**Usuario:** Beatriz Riverón
Nombre historia: Actualizar Usuario.	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Baja
Puntos estimados: 1	**Iteración asignada:** 1
Programador responsable: Robin Cortina Perdomo	
Descripción: El Administrador selecciona en el formulario general la opción "Modificar Usuario". Primeramente se muestra un formulario en el que el Administrador debe especificar el usuario a modificar, luego se muestra un formulario con los campos a llenar.	
Observaciones: Para poder modificar un usuario primero se debe verificar que este en la base de datos de no estar se muestra un mensaje especificando que no existe.	

Tabla Historia de Usuario #12 Actualizar Usuario

Historia de Usuario	
Número: 13	**Usuario:** Beatriz Riverón
Nombre historia: Listar Usuarios	
Prioridad en negocio: Media	**Riesgo en desarrollo:** Baja
Puntos estimados: 1	**Iteración asignada:** 1
Programador responsable: Robin Cortina Perdomo	
Descripción: El usuario selecciona en el formulario general la opción "Listado de Usuarios". Luego se muestra un formulario con la lista de usuarios que se encuentran guardados en la base de datos.	

Tabla Historia de Usuario #13 Listar Usuario

Historia de Usuario	
Número: 14	**Usuario:** Beatriz Riverón
Nombre historia: Eliminar Usuario.	
Prioridad en negocio: Media	**Riesgo en desarrollo:** Baja
Puntos estimados: 1	**Iteración asignada:** 1
Programador responsable: Robin Cortina Perdomo	
Descripción: El administrador selecciona en el formulario general la opción "Eliminar Usuario". Luego se muestra un formulario con la lista de usuarios que se encuentran guardados en la base de datos, el administrador selecciona el usuario a eliminar y presiona la opción de eliminar.	
Observaciones: Para eliminar un usuario se selecciona y se acepta luego se muestra un mensaje de confirmación para la eliminación segura.	

Tabla Historia de Usuario #14 Eliminar Usuario.

Tarjetas CRC

Datos_Generales	
Description: Esta es la clase que va a almacenar los datos generales del ensayo,independientemente del tipo de experimento que se vaya a realizar	
Attributes:	
Name	Description
codigo	Llave primaria y codigo de la obra
cala	Numero de la cala
muestra	Este es el numero de la muestra a ensayar
registro	Registro de la obra
norma	De acuerdo al tipo de ensayo es la norma a utilizar
referencia	Referencia
descripcion	Una breve descripcion de la propiedades de la muestra
obra	Nombre de la obra que se desea realizar
fecha	Fecha en que se realiza el ensayo
nombre	Nombre de tipo de ensayo
profundidadI	Profundidad a la que se tomo la primera muestra
profundidadF	Profundidad a la que se tomo la ultima muestra
Responsibilities:	
Name	Collaborator
Insertar_Datos_Generales()	
Modificar_Datos_generales()	
Eliminar_Datos_Generales()	
Listar_Datos_Generales()	

Tabla Tarjeta CRC #1 Datos_Generales

Carga_Brazo	
Description: Guarda la carga que se aplica al brazo durante el ensayo	
Attributes:	
Name	Description
codigo	Llave de datosgenerales
carga_brazo1	Cargas aplicadas en el brazo 1-n
carga_brazo2	
carga_brazo3	
carga_brazo4	
carga_brazo5	
carga_brazo6	
Responsibilities:	
Name	Collaborator
Insertar_CargaB()	datosgenerales
Modificar_CargaB()	
Eliminar_CargaB()	

Tabla Tarjeta CRC #2 Carga_Brazo

Carga_Muestra	
Description: Guarda la carga que se le aplica a la muestra	
Attributes:	
Name	Description
codigo	Llave de la tabla datosgenerales
carga_muestra1	Diferentes cargas aplicadas a la muestra 1-n
carga_muestra2	
carga_muestra3	
carga_muestra4	
carga_muestra5	
carga_muestra6	
Responsibilities:	
Name	Collaborator
Insertar_Carga()	datosgenerales
Modificar_Carga()	
Eliminar_Carga()	

Tabla Tarjeta CRC #3 Carga_Muestra

Contraccion	
Description: Guarda los datos que determinan la contraccion de la muestra	
Attributes:	
Name	Description
codigo	Llave de datosgenerales
mzsahumedainicial	Muestra acabada de extraer
mzsasecafinal	Muestra sin el contenido de humedad
taramasaresipiente	Peso del resipiente donde se ensaya la muestra
volumenini	Volumen de la muestra con humedad
volumenfin	Volumen de la muestra sin humedad
Responsibilities:	
Name	Collaborator
Crear_Contraccion()	datosgenerales
Modificar_Contraccion()	
Eliminar_Contraccion()	

Tabla Tarjeta CRC #4 Contracción

Deformacion	
Description: Guarda la deformacion que presenta la muestra durante el tiempo del ensayo	
Attributes:	
Name	Description
codigo	Llave de datos generales
deformacion1	Deformaciones de la muestra 1-n
deformacion2	
deformacion3	
deformacion4	
deformacion5	
deformacion6	
Responsibilities:	
Name	Collaborator
Insertar_Deformacion()	datosgenerales
Modificar_Deformacion()	
Eliminar_Deformacion()	

Tabla Tarjeta CRC #5 Deformación

Hinchamiento	
Description: Esta clase almacena los datos que determinan el hinchamiento	
Attributes:	
Name	Description
codigo	llave de la tabla datosgenerales
lecturapordivision	Lectura que se lleva de acuerdo a cada division del ensayo
relacionbrazo	Relacion de hichamiento en el brazo
Responsibilities:	
Name	Collaborator
Crear_Hinchamiento()	datosgenerales
Modificar_Hinchamiento()	
Eliminar_Hinchamiento()	

Tabla Tarjeta CRC #6 Hinchamiento

Humedades	
Description: Esta guarda los datos de las humedades de la muestra	
Attributes:	
Name	Description
codigo	Llave de la tabla datosgenerales
masahumedamastarainicial	Muestra humeda mas el peso del recipiente
masahumedamastarafinal	Segunda muestra mas el peso de resipiente
masacecamastarainicial	Muestra seca mas el peso del recipiente
masacecamastarafinal	Segunda muestra mas el peso del resipiente
tarainicial	Peso inicial
tarafinal	Peso final
masasecainicial	Masa seca inicial de la muestra
masasecafinal	Masa seca final de la muestra
masadelaguainicial	Masa de agua inicial
masadelaguafinal	Masa del agua final
humedadinicial	Humedadinicial de la muestra
humedadfinal	Humedad final de la muestra
Responsibilities:	
Name	Collaborator
Insertar_Humedades()	datosgenerales
Modificar_Humedades()	
Eliminar_Humedades()	

Tabla Tarjeta CRC #7 Humedades

Muestra	
Description: Guarda los datos de la muestra	
Attributes:	
Name	Description
codigo	Lave de datosgenerales
muestraa	Si la muestra esta alterada o no
aparato	Numero del aparato
anillo	Numero de anillo
Responsibilities:	
Name	Collaborator
Insertar_Muestra()	datosgenerales
Modificar_Muestra()	
Eliminar_Muestra()	

Tabla Tarjeta CRC #8 Muestra

Presion	
Description: Guarda la presiones que se suminnistarn a las muestras	
Attributes:	
Name	Description
codigo	Llave de la tabla datosgenerales
presion1	Diferentes tipos de presiones 1-n
presion2	
presion3	
presion4	
presion5	
presion6	
Responsibilities:	
Name	Collaborator
Insertar_Presion()	datosgenerales
Modificar_Presion()	
Eliminar_Presion()	

Tabla Tarjeta CRC #9 Presión

Propiedades	
Description: Guarda los datos de las propiedades de las muestras	
Attributes:	
Name	Description
codigo	Llave de datosgenerales
masahumedamastaraini	Masa humeda mas recipiente inicial
masahumedamastarafin	Masa humeda mas recipiente final
taraanilloini	Resipiente mas la masa del anillo inicial
taraanillofin	Resipiente mas la masa del anillo final
masahumedaini	Masa con el contenido de humedad
masahumedafin	Masa con el conenido de humedad final
alturaini	Altura del suelo inicial
alturafin	Altura del suelo final
diametroini	Diametro de la muestra inicial
diametrofin	Diametro de la muestra final
Responsibilities:	
Name	Collaborator
Insertar_Propiedades()	datosgenerales
Modificar_Propiedades()	
Eliminar_Propiedades()	

Tabla Tarjeta CRC #10 Propiedades

Realizado	
Description: Guarda los datos de la personas que realizaron el informe	
Attributes:	
Name	Description
codigo	Lave de la tabla datosgenerales
nom_realiza	Nombre del que realiza el ensayo
nom_reviza	Nombre del que reviza y aprueba el ensayo
nom_calcula	Nombre del que clcula el ensayo
cargo_realiza	Cargo del que realiza el ensayo
cargo_calcula	Cargo del que realiza los calculos
cargo_reviza	Cargo del que reviza el ensayo
Responsibilities:	
Name	Collaborator
Insertar_Realizado()	datosgenerales
Modificar_Realizado()	
Eliminar_Realoizado()	

Tabla Tarjeta CRC #11 Realizado

Resultado_Contraccion	
Description: Guarda los resultados del experimento	
Attributes:	
Name	Description
codigo	Lave de datosgenerales
masahumeda	Masa con el contenido de humedad
masaseca	Masa sin humedad
contenidohumedad	Humedad que posee la muestra
limitescontraccion	Límite de contraccion de la muestra
relacioncontraccion	Relacion entre la contraccion y la muestra
contraccionvolumetrica	Contraccion en el volumen
contraccionlineal	Contracion en la longitud
pesoespesifico	Peso de la muestra despues del ensayo
Responsibilities:	
Name	Collaborator
Insertar_Resultado_Contraccion()	datosgenerales
Modificar_Resultado_Contraccion()	
Eliminar_Resultado_Contraccion()	

Tabla Tarjeta CRC #12 Resultado_Contraccion

Resultado_Hinchamiento	
Description: Guarda el resultado del ensayo de hinchamiento	
Attributes:	
Name	Description
codigo	Llave de datosgenerales
resultado_hinchamiento	Es el hinchamiento de la muestra despues del ensayo
Responsibilities:	
Name	Collaborator
Insertar_Resultado_Hinchamiento()	datosgenerales
Modificar_Resultado_Hinchamiento()	
Eliminar_Resultado_Hinchamiento()	

Tabla Tarjeta CRC #13 Resultado_Hinchamiento

Tareas

Tareas	
Número de la Tarea:1	**Número de Historia:**1
Nombre de la Tarea: Calcular Hinchamiento Controlado.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:**0.5
Fecha de Inicio: 30/enero/2012.	**Fecha Fin:** 2/febrero/2012.
Programador Responsable: Robin Cortina Perdomo	
Descripción: Se crea un Informe de Hinchamiento Controlado en el cual se introducen los datos y luego se calcula el hinchamiento de la muestra.	

Tabla Tarea #1 Calcular Hinchamiento Controlado

Tareas	
Número de la Tarea:2	**Número de Historia:**1
Nombre de la Tarea: Graficar.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:**1
Fecha de Inicio: 2/febrero/2012.	**Fecha Fin:** 9/febrero/2012.
Programador Responsable: Robin Cortina Perdomo	
Descripción: Se crea un Informe de Hinchamiento Controlado en el cual se introducen los datos y después de calculado el hinchamiento de la muestra se presiona la opción de graficar y el sistema muestra la gráfica de presión contra logaritmo del tiempo.	

Tabla Tarea #2 Graficar

Tareas	
Número de la Tarea:3	**Número de Historia:**1, 2
Nombre de la Tarea: Imprimir.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:**0.5
Fecha de Inicio: 9/febrero/2012.	**Fecha Fin:** 13/febrero/2012.
Programador Responsable: Robin Cortina Perdomo	
Descripción: Después de haber calculado el hinchamiento de la muestra y realizado el grafico del comportamiento de esta se procede a realizar el impresión del informe.	

Tabla Tarea #3 Imprimir

Tareas	
Número de la Tarea:4	**Número de Historia:**1
Nombre de la Tarea: Guardar Informe de Hinchamiento Controlado.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:**1
Fecha de Inicio: 13/febrero/2012.	**Fecha Fin:** 21/febrero/2012.
Programador Responsable: Robin Cortina Perdomo	
Descripción: Una vez calculado, imprimido y realizado el gráfico se procede a guardar el informe terminado en la base de datos.	

Tabla Tarea #4 Guardar Informe de Hinchamiento Controlado

Tareas	
Número de la Tarea:5	**Número de Historia:**2

Nombre de la Tarea: Actualizar Informe Hinchamiento Controlado.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:**1
Fecha de Inicio: 22/febrero/2012.	**Fecha Fin:** 1/marzo/2012.
Programador Responsable: Robin Cortina Perdomo	
Descripción: Se realiza una búsqueda del Hinchamiento deseado por el nombre de Obra mostrando una Interfaz con los datos creados, de esta forma modificamos y continuamos con el cálculo ,la graficación e impresión de los resultados modificados.	

Tabla Tarea #5 Actualizar Informe de Hinchamiento Controlado

Tareas	
Número de la Tarea:6	**Número de Historia:**3
Nombre de la Tarea: Buscar Informe Hinchamiento Controlado.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:**1
Fecha de Inicio: 2/marzo/2012.	**Fecha Fin:** 9/marzo/2012.
Programador Responsable: Robin Cortina Perdomo	
Descripción: Se busca un informe por el nombre de la Obra y se muestra un formulario con los datos de esa Obra.	

Tabla Tarea #6 Buscar Informe de Hinchamiento Controlado

Tareas	
Número de la Tarea:7	**Número de Historia:**4
Nombre de la Tarea: Listar Informe Hinchamiento Controlado.	

Tipo de Tarea: Desarrollo.	**Puntos Estimados:**1
Fecha de Inicio: 9/marzo/2012.	**Fecha Fin:** 16/marzo/2012.
Programador Responsable: Robin Cortina Perdomo	
Descripción: Se muestra una Interfaz con la Lista existente de Hinchamiento Controlados.	

Tabla Tarea #7 Listar Informe de Hinchamiento Controlado

Tareas	
Número de la Tarea:8	**Número de Historia:**5
Nombre de la Tarea: Eliminar Informe Hinchamiento Controlado.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:**1
Fecha de Inicio: 16/marzo/2012.	**Fecha Fin:** 23/marzo/2012.
Programador Responsable: Robin Cortina Perdomo	
Descripción: Se muestra un formulario con una lista existente de Hinchamiento Controlado el cual al seleccionar uno se activa un botón "Eliminar". Cuando seleccionamos en él se elimina correctamente este Hinchamiento.	

Tabla Tarea #8 Eliminar Informe de Hinchamiento Controlado

Tareas	
Número de la Tarea:9	**Número de Historia:**6
Nombre de la Tarea: Crear Informe de Límites de Contracción.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:**3

Fecha de Inicio: 23/marzo/2012.	**Fecha Fin:** 13/abril/2012.
Programador Responsable: Robin Cortina Perdomo	
Descripción: Se crea un Informe de Límites de Contracción en el cual se calculan los procesos de este ensayo así la Impresión de estos resultados.	

Tabla Tarea #9 Crear Informe de Límites de Contracción.

Tareas	
Número de la Tarea:10	**Número de Historia:**7
Nombre de la Tarea: Actualizar Informe de Límites de Contracción.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:**1
Fecha de Inicio: 13/abril/2012.	**Fecha Fin:** 20/abril/2012.
Programador Responsable: Robin Cortina Perdomo	
Descripción: Se realiza una búsqueda del Límite de Contracción deseado por el nombre de Obra mostrando una Interfaz con los datos creados, de esta forma modificamos y continuamos con el cálculo ,la graficación e impresión de los resultados modificados.	

Tabla Tarea #10 Actualizar Informe de Límites de Contracción.

Tareas	
Número de la Tarea:11	**Número de Historia:**8
Nombre de la Tarea: Buscar Informe de Límites de Contracción.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:**1
Fecha de Inicio: 20/abril/2012.	**Fecha Fin:** 27/abril/2012.

Programador Responsable: Robin Cortina Perdomo
Descripción: Se busca un informe por el nombre de la Obra y se muestra un formulario con los datos de esa Obra.

Tabla Tarea #11 Buscar Informe de Límites de Contracción.

Tareas	
Número de la Tarea:12	**Número de Historia:**9
Nombre de la Tarea: Listar Informe de Límites de Contracción.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:**1
Fecha de Inicio: 27/abril/2012.	**Fecha Fin:** 4/mayo/2012.
Programador Responsable: Robin Cortina Perdomo	
Descripción: Se muestra una Interfaz con la Lista existente de Límites de Contracción.	

Tabla Tarea #12 Listar Informe de Límites de Contracción.

Tareas	
Número de la Tarea:13	**Número de Historia:**10
Nombre de la Tarea: Eliminar Informe de Límites Contracción.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:**1
Fecha de Inicio: 4/mayo/2012.	**Fecha Fin:** 11/mayo/2012.
Programador Responsable: Robin Cortina Perdomo	
Descripción: Se muestra un formulario con una lista existente de Hinchamiento Controlado el cual al seleccionar uno se activa un botón	

"Eliminar". Cuando seleccionamos en él se elimina correctamente este Límite.

Tabla Tarea #13 Eliminar Informe de Límites de Contracción.

Tareas	
Número de la Tarea:14	**Número de Historia:11**
Nombre de la Tarea: Crear Usuarios.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:**1
Fecha de Inicio: 11/mayo/2012.	**Fecha Fin:** 18/mayo/2012.
Programador Responsable: Robin Cortina Perdomo.	
Descripción: Se crea Usuario el cual va a ser parte de la lista de usuarios que pueden acceder al sistema, con un ROL de Usuarios o Administrador.	

Tabla Tarea #14 Crear Usuarios

Tareas	
Número de la Tarea:15	**Número de Historia:12**
Nombre de la Tarea: Actualizar Usuarios.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:**1
Fecha de Inicio: 18/mayo/2012.	**Fecha Fin:** 21/mayo/2012.
Programador Responsable: Robin Cortina Perdomo.	
Descripción: Se realiza una búsqueda del Usuario deseado por el usuario mostrando una Interfaz con una Lista de los usuarios, de esta forma modificamos y Actualizamos los datos del Usuario deseado.	

Tabla Tarea #15 Actualizar Usuarios

Tareas	
Número de la Tarea:16	**Número de Historia:**13
Nombre de la Tarea: Eliminar Usuarios.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:**1
Fecha de Inicio: 21/mayo/2012.	**Fecha Fin:** 28/mayo/2012.
Programador Responsable: Robin Cortina Perdomo.	
Descripción: Se muestra un formulario con una lista existente de los Usuarios a eliminar, el cual al seleccionar uno se activa un botón “Eliminar”. Cuando seleccionamos en él se elimina correctamente este Usuario.	

Tabla Tarea #16 Eliminar Usuarios

Tareas	
Número de la Tarea:17	**Número de Historia:**14
Nombre de la Tarea: Listar Usuarios.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:**1
Fecha de Inicio: 28/mayo/2012.	**Fecha Fin:** 2/junio/2012.
Programador Responsable: Robin Cortina Perdomo.	
Descripción: Se muestra un formulario con una lista existente de los Usuarios existentes en la base de datos.	

Tabla Tarea #16 Listar Usuarios

Pruebas de Aceptación

Prueba de aceptación Crear informe de Hinchamiento Controlado
HU: Crear informe de Hinchamiento Controlado.
Nombre: Prueba para crear un informe de hinchamiento controlado
Descripción: El propósito de esta prueba es para determinar si se calcula el hinchamiento correctamente, así como la realización del gráfico, se logre imprimir y se guarde correctamente en la base de datos.
Condiciones de ejecución: El usuario debe llenar los campos especificados.
Entrada/Pasos de ejecución: Primeramente el usuario introduce los datos y luego presiona la opción calcular, si se calculó correctamente se habilita el botón “Ver Grafica” y se presiona si se generó bien la gráfica se selecciona el botón “Imprimir“ y luego la opción de guardar.
Resultado esperado: Se muestran los resultados esperados así como el grafico correcto además se logra imprimir el informe demás de ser guardado en la base de datos
Evaluación de la prueba: Aceptada.

Tabla Prueba de Aceptación #1 Crear informe de Hinchamiento Controlado

Prueba de aceptación Actualizar informe de Hinchamiento Controlado
HU: Actualizar informe de Hinchamiento Controlado.
Nombre: Prueba para modificar un informe de hinchamiento controlado
Descripción: El propósito de esta prueba es para determinar si se modifica un informe que se encuentre en la base de datos.
Condiciones de ejecución: **-** El usuario debe especificar el informe a modificar **-** El informe debe existir en la base de datos
Entrada/Pasos de ejecución: Primeramente el usuario introduce el nombre del informe que desea modificar, y cuando se muestre modifica los campos deseados y presiona el botón “Actualizar” si se actualizó correctamente se habilita el botón guardar y se presiona.
Resultado esperado: Se modifican los datos del informe en la base de datos
Evaluación de la prueba: Aceptada.

Tabla Prueba de Aceptación #2 Actualizar informe de Hinchamiento Controlado

Prueba de aceptación Listar informes de Hinchamiento Controlado
HU: Listar informe de Hinchamiento Controlado.
Nombre: Prueba para listar los informes de hinchamiento controlado
Descripción: El propósito de esta prueba es para determinar si se listan todos los informes que se encuentran en la base de datos.
Condiciones de ejecución: El usuario presiona la opción Listar Informe de Hinchamiento Controlado
Entrada/Pasos de ejecución: Una vez presionado la opción de Listar Informe de Hinchamiento Controlado se muestra un formulario con una tabla en la que se muestran todos los informes de Hinchamiento Controlado.
Resultado esperado: Se muestran los datos de todos los informes que se encuentran en la base de datos
Evaluación de la prueba: Aceptada.

Tabla Prueba de Aceptación #3 Listar informe de Hinchamiento Controlado

Prueba de aceptación Buscar informe de Hinchamiento Controlado
HU: Buscar informe de Hinchamiento Controlado.
Nombre: Prueba para buscar un informe de hinchamiento controlado.
Descripción: El propósito de esta prueba es para determinar si se busca un informe que se encuentra en la base de datos.
Condiciones de ejecución: **-**El usuario presiona la opción Buscar Informe de Hinchamiento Controlado **-**Selecciona de la lista el informe que desea buscar
Entrada/Pasos de ejecución: Una vez seleccionado el informe se muestran todos los datos referentes a este informe.
Resultado esperado: Se muestran los datos del informe que fue selecciona do en la lista de informes.
Evaluación de la prueba: Aceptada.

Tabla Prueba de Aceptación #4 Buscar informe de Hinchamiento Controlado

Prueba de aceptación Eliminar informe de Hinchamiento Controlado
HU: Eliminar informe de Hinchamiento Controlado.
Nombre: Prueba para eliminar un informe de hinchamiento controlado que se encuentre en la base de datos.
Descripción: El propósito de esta prueba es para determinar si se elimina correctamente un informe que se encuentra en la base de datos.
Condiciones de ejecución: **-**El usuario presiona la opción Eliminar Informe de Hinchamiento Controlado **-**Selecciona de la lista el informe que desea buscar
Entrada/Pasos de ejecución: Cuando se muestra el formulario con la lista de todos los informes que se encuentran en la base de datos se selecciona el informe que se desea eliminar y se presiona el botón “Eliminar” el sistema pide confirmación y lo elimina.
Resultado esperado: Se eliminan todos los datos del informe que fue seleccionado en la lista de informes.
Evaluación de la prueba: Aceptada.

Tabla Prueba de Aceptación #5 Eliminar informe de Hinchamiento Controlado

Prueba de aceptación Crear informe de Límites de Contracción
HU: Crear informe de Límites de Contracción.
Nombre: Prueba para crear un informe de Límites de Contracción
Descripción: El propósito de esta prueba es para determinar si se calculan los límites de contracción de la muestra correctamente, así como imprimir y se guarde correctamente en la base de datos.
Condiciones de ejecución: El usuario debe llenar los campos especificados.
Entrada/Pasos de ejecución: Primeramente el usuario introduce los datos y luego presiona la opción calcular, si se calculó correctamente se selecciona el botón “Imprimir” y luego la opción de guardar.
Resultado esperado: Se muestran los resultados esperados, además se logra imprimir el informe demás de ser guardado en la base de datos
Evaluación de la prueba: Aceptada.

Tabla Prueba de Aceptación #6 Crear informe de Límites de Contracción

Prueba de aceptación Actualizar informe de Límites de Contracción
HU: Actualizar informe de Límites de Contracción.
Nombre: Prueba para modificar un informe de Límites de Contracción
Descripción: El propósito de esta prueba es para determinar si se modifica un informe que se encuentre en la base de datos.
Condiciones de ejecución: **-** El usuario debe especificar el informe a modificar **-** El informe debe existir en la base de datos
Entrada/Pasos de ejecución: Primeramente el usuario introduce el nombre del informe que desea modificar, y cuando se muestre modifica los campos deseados y presiona el botón “Actualizar” si se actualizó correctamente se habilita el botón guardar y se presiona.
Resultado esperado: Se modifican los datos del informe en la base de datos
Evaluación de la prueba: Aceptada.

Tabla Prueba de Aceptación #7 Actualizar informe de Límites de Contracción

Prueba de aceptación Listar informes de Límites de Contracción
HU: Listar informe de Límites de Contracción.
Nombre: Prueba para listar los informes de Límites de Contracción
Descripción: El propósito de esta prueba es para determinar si se listan todos los informes que se encuentran en la base de datos.
Condiciones de ejecución: El usuario presiona la opción Listar Informe de Límites de Contracción
Entrada/Pasos de ejecución: Una vez presionado la opción de Listar Informe de Límites de Contracción se muestra un formulario con una tabla en la que se muestran todos los informes de Límites de Contracción.
Resultado esperado: Se muestran los datos de todos los informes que se encuentran en la base de datos
Evaluación de la prueba: Aceptada.

Tabla Prueba de Aceptación #8 Listar informe de Límites de Contracción

Prueba de aceptación Buscar informe de Límites de Contracción

HU: Buscar informe de Límites de Contracción.
Nombre: Prueba para buscar un informe de Límites de Contracción.
Descripción: El propósito de esta prueba es para determinar si se busca un informe que se encuentra en la base de datos.
Condiciones de ejecución: **-**El usuario presiona la opción Buscar Informe de Límites de Contracción **-**Selecciona de la lista el informe que desea buscar
Entrada/Pasos de ejecución: Una vez seleccionado el informe se muestran todos los datos referentes a este informe.
Resultado esperado: Se muestran los datos del informe que fue selecciona do en la lista de informes.
Evaluación de la prueba: Aceptada.

Tabla Prueba de Aceptación #9 Buscar informe de Límites de Contracción

Prueba de aceptación Eliminar informe de Límites de Contracción
HU: Eliminar informe de Límites de Contracción.
Nombre: Prueba para eliminar un informe de Límites de Contracción que se encuentre en la base de datos.
Descripción: El propósito de esta prueba es para determinar si se elimina correctamente un informe que se encuentra en la base de datos.
Condiciones de ejecución: **-**El usuario presiona la opción Eliminar Informe de Límites de Contracción **-**Selecciona de la lista el informe que desea buscar
Entrada/Pasos de ejecución: Cuando se muestra el formulario con la lista de todos los informes que se encuentran en la base de datos se selecciona el informe que se desea eliminar y se presiona el botón “Eliminar” el sistema pide confirmación y lo elimina.
Resultado esperado: Se eliminan todos los datos del informe que fue seleccionado en la lista de informes.
Evaluación de la prueba: Aceptada.

Tabla Prueba de Aceptación #10 Eliminar informe de Límites de Contracción