Universidad de las Ciencias Informáticas

Facultad # 6

<u>Título:</u>

Análisis del núcleo del sistema Geólogo Minero (GeolMin) del proyecto de Minería.

<u>Trabajo para optar por el Título de Ingeniero en Ciencias Informáticas</u>

<u>Autor</u>: Yirian Castell Mena

<u>Tutor</u>: Ing. Armando Ortíz Cabrera

Ciudad de la Habana, Junio del 2011
"Año 53 de la Revolución"

Frase

"Sean capaces siempre de sentir, en lo más hondo, cualquier injusticia realizada contra cualquiera, en cualquier parte del mundo. Es la cualidad más linda del revolucionario."

Ernesto Che Guevara.

Dedicatoria

A mis abuelos y mis padres que son mi razón de ser, sin ellos hubiera sido imposible llegar donde estoy hoy.

Los quiero con la vida, son lo más importante que tengo y a ustedes va dedicado este momento tan importante.

Yiri.

Agradecimientos

Mis más sinceros agradecimientos a todos los que colaboraron con el desarrollo del presente trabajo y a todas las personas que influyeron en mi formación como profesional.

Mi especial agradecimiento a mis padres queridos por la confianza que siempre depositaron en mí, por su apoyo y dedicación.

A mis abuelos que son la luz de mis ojos y sin su apoyo no hubiera podido llegar a la meta final.

A toda mi familia, mis tíos y tías, mis primas, mis primos, por ayudarme siempre.

A mi tutor por haberme apoyado en todo lo que fue necesario. Gracias por el tiempo que me has dedicado.

A mis amigos con los que he compartido alegrías y tristezas.

A la Revolución y a la UCI por brindarme la oportunidad de ser hoy una profesional.

A todos, muchas gracias.

Los quiero mucho

Yiri.

Declaración de autoría

Declaro que soy la única autora de este trabajo y autorizo a la Facultad 6 de la Universidad de las Ciencias Informáticas; así como a dicho centro para que hagan el uso que estimen pertinente con este trabajo.

Para que así conste firmo la presente a los ___ días del mes de _____ del año 2011.

Yirian Castell Mena **Ing. Armando Ortiz Cabrera**

_____ _____

Firma del Autor Firma del Tutor

Resumen

Con el desarrollo del presente trabajo de diploma se pretende informatizar la actividad minera en Cuba. Dicha actividad presenta algunos inconvenientes, los cuales imposibilitan que el desarrollo productivo se realice con eficiencia y calidad. Varios de los problemas identificados, como los software utilizados son de compañías extranjeras, los costos por concepto de pago de licencias y actualizaciones de software son elevados, tienen solución desde el punto de vista informático.

Este documento expone los resultados de todo el trabajo investigativo realizado. Se describen los conceptos necesarios para lograr una mejor comprensión del tema. Se analizan dos sistemas que fueron de gran uso para la industria minera cubana. Se realiza un análisis de las tecnologías y tendencias que existen en la actualidad a nivel mundial que son útiles en el desarrollo de la propuesta de solución. Se describen las actividades desarrolladas por el rol de analista de software y los artefactos que se construyen durante el desarrollo del producto para así darle una solución a la problemática planteada. Se confecciona el diagrama de casos de uso del sistema, el diagrama de paquetes, la matriz de trazabilidad entre requisitos funcionales y casos de uso, así como los diagramas de clases de análisis correspondientes a cada caso de uso y los diagramas de colaboración, finalizando la investigación con el análisis de factibilidad, donde queda estimado el esfuerzo del módulo núcleo.

Palabras clave: analista, esfuerzo, factibilidad, industria, licencias, módulo, núcleo.

Índice

Índice de tablas

Índice de figuras

INTRODUCCIÓN

Los hombres, desde los tiempos más remotos, habían tenido ya en gran aprecio muchos minerales por su transparencia, espléndido color y dureza, los mismos que aún hoy tienen gran valor como piedras preciosas, fueron entre aquellas personas considerados como algo sobrenatural. Desde ese entonces el hombre ha trabajado fuertemente en este campo convirtiéndolo en el sector principal para el desarrollo de la economía y el avance de la sociedad.

En la actualidad, con el rápido incremento de las tecnologías informáticas, hasta las más pequeñas empresas ponen a disposición de los centros desarrolladores de software la informatización de sus procesos empresariales. El país se encuentra inmerso en dicho proceso, con el propósito de extender el desarrollo informático hacia todos los sectores de la sociedad. No cabe duda de que un elevado número de las empresas cubanas van camino a la adaptación de los modelos de negocio basados en las Tecnologías de la Información y las Comunicaciones (TIC).

Para una empresa u organización la implementación de las tecnologías de la información se ha vuelto una necesidad y una cuestión de sostenimiento dentro de un mundo tan competitivo y desarrollado; además, es un paso que lleva a una actividad dinámica y eficaz, no importa el sector en el que esté operando dentro del sistema económico. No existe organización alguna que no pueda obtener un beneficio mediante la introducción de las TIC en sus procesos de producción, gestión, control, planificación y otros de vital importancia.

El avance de las TIC[1] está provocando una migración en el funcionamiento interno de las organizaciones hacia sistemas electrónicos y digitales. Esto se logra mediante la instalación de redes informáticas y la implementación de aplicaciones y sistemas que mejoran la rapidez de las operaciones inherentes que a su vez están protegidas por mecanismos de seguridad.

Cuba ha tratado de mantenerse, a pesar de su estatus de país subdesarrollado y de todas las barreras que le impone el bloqueo, en contacto directo con el desarrollo de las nuevas tecnologías y los diferentes

[1]TIC: Tecnologías de la Información y las Comunicaciones.

procesos por los que ha transitado la informática; es por ello que prepara a su población en el estudio de esta ciencia y al mismo tiempo extiende la utilización de estos medios a otras esferas de la vida.

La Universidad de las Ciencias Informáticas (UCI) desarrolla conjuntamente con el MINBAS[2], la Oficina Nacional de Recursos Minerales (ONRM) y el Centro de Investigación de Geología y Paleontología un proyecto de informatización que constituye un paso hacia el aprovechamiento racional del conocimiento geológico formando parte de este, el petróleo y la minería, siendo esta última el objeto de estudio.

El proyecto que se ejecuta lleva por nombre “Minería” y pretende informatizar la actividad minera en el país. Dicha actividad presenta algunos inconvenientes, los cuales imposibilitan que el desarrollo productivo se realice con eficiencia y calidad. Varios de los problemas identificados tienen solución desde el punto de vista informático y se le ha dado la tarea al centro de desarrollo Geoinformática y Señales Digitales (GEySED) de la UCI de desarrollar un software que resuelva tal situación.

Problemas identificados:

1. El software utilizado es de compañías extranjeras.
2. Altos costos por concepto de pago de licencias y actualizaciones de software.
3. La estructura y el comportamiento del software es rígido debido a que el código fuente no se encuentra disponible para cualquier usuario, es decir no se hace público y no se le pueden realizar nuevas modificaciones o extender su desarrollo por terceras empresas.

Dada la explicación expuesta anteriormente, se determinó como problema a resolver:

¿Cómo obtener la representación técnica del núcleo del sistema GeolMin de forma tal que permita crear su diseño?

El objeto de estudio lo constituyen el análisis de la estructura y el comportamiento del núcleo de los sistemas informáticos. El campo de acción es el análisis de la estructura y el comportamiento del núcleo de los sistemas informáticos utilizados en la Minería.

[2]MINBAS: Ministerio de la industria básica.

El objetivo general de este trabajo es: Diseñar la propuesta de los casos de uso y requisitos del núcleo del sistema GeolMin del proyecto Minería.

Para cumplir con el objetivo general propuesto y darle solución a la situación problémica planteada, se proponen las siguientes tareas investigativas:

1. Describir los conceptos asociados al dominio del problema.
2. Argumentar las tecnologías a utilizar.
3. Construir el modelo de negocio o dominio según corresponda.
4. Construir la matriz de trazabilidad de los requisitos del sistema.
5. Elaborar el análisis de los requisitos propuesto para el sistema.
6. Documentar los artefactos asociados al rol de analista utilizando las plantillas del expediente de proyecto.
7. Validar los modelos generados.

La idea a defender plantea que con el desarrollo del modelo de sistema y los requisitos del núcleo del sistema GeolMin, se podrá contar con un marco de trabajo que facilite la integración de las funcionalidades básicas y complementarias que soporten los nuevos componentes desarrollados en el proyecto de minería del departamento GEySED.

Para lograr un mejor entendimiento y obtener información valiosa sobre el tema investigado se utilizan los siguientes métodos científicos:

Métodos teóricos

- El Método Histórico – Lógico: se utilizó para investigar si existen proyectos informáticos de este tipo implementados en el país e internacionalmente, y en caso de existir, conocer cómo es su funcionamiento.
- Analítico-Sintético: permitió el estudio de la bibliografía utilizada y posibilitó que se efectuara una recapitulación de la misma.
- Modelación: se empleó para la modelación de diagramas, logrando un mejor entendimiento de lo que se va a diseñar e implementar en futuras iteraciones.

Métodos empíricos

- Entrevistas: se aplicó a especialistas de la minería del Instituto de Geopaleontología (IGP), para recopilar la información necesaria sobre las aplicaciones informáticas utilizadas y conceptos relacionados con la minería.

La presente investigación queda estructurada de la siguiente forma:

- **Capítulo 1: Fundamentación teórica**

En este capítulo se realiza un análisis del objeto de estudio, la posible existencia de sistemas vinculados al campo de acción, se reflejan algunas tendencias y tecnologías actuales seleccionadas para el desarrollo de la solución propuesta. Se describen los conceptos que intervienen en el proceso de esta investigación que ayudan al entendimiento del problema.

» **Capítulo 2: Características del sistema**

Descripción de la solución propuesta a través del modelado del negocio y sistema. Descripción de los requisitos funcionales (RF) y casos de uso, así como la elaboración de la matriz de trazabilidad. Incluye la definición del modelo de análisis del sistema, diagramas de colaboración y la organización del sistema en paquetes.

» **Capítulo 3: Estudio de factibilidad y validación del sistema**

Descripción de la planificación y la gestión de proyecto basado en el método de estimación de puntos de función de casos de uso sin ajustar. Además de la descripción de los resultados del estudio de factibilidad.

CAPÍTULO #1
FUNDAMENTACIÓN TEÓRICA

INTRODUCCIÓN

En el presente capítulo se describen y clasifican los conceptos necesarios para lograr una mejor comprensión del tema, además se identifican los principales problemas que motivan la investigación. Se analizan dos sistemas que fueron de gran uso para la industria minera cubana. Se realiza un análisis de las tecnologías y tendencias que existen en la actualidad a nivel mundial y que pudieran ser útiles en el desarrollo de la propuesta de solución como herramientas, metodología de desarrollo de software[3] y UML[4]como lenguaje de modelado. Se describen las actividades desarrolladas por el rol de analista de software y los artefactos[5] que se construyen durante el desarrollo del producto para así darle solución a la problemática planteada.

1.1. CONCEPTOS ASOCIADOS AL DOMINIO DEL PROBLEMA

1.1.1. Geología

Es una ciencia multifacética, la cual tiene por objeto investigar la composición, estructura, origen y evolución de la tierra. Etimológicamente, la palabra geología viene del griego geo “Tierra”, y logos “Estudio”, es decir, trata el estudio y el conocimiento de la Tierra. Su estudio abarca el proceso de la formación de las montañas, las rocas, los minerales, los restos orgánicos fosilizados. (Mis respuestas, 2005).

Como ciencia mayor, la geología no sólo implica el estudio de la superficie terrestre, también se interesa por el interior del planeta. Este conocimiento es de interés científico básico y está al servicio de la

[3]Software: Término genérico que se aplica a los componentes no físicos de un sistema informático. Eje. los programas, sistemas operativos, que permiten a este ejecutar sus tareas.

[4] UML: lenguaje para especificar, visualizar, construir y documentar el proceso de desarrollo de software.

[5] Artefactos: es una información que es utilizada o producida mediante un proceso de desarrollo de software.

humanidad. De esta forma, la geología aplicada se centra en la búsqueda de minerales útiles en el interior de la tierra, la identificación de entornos estables, en términos geológicos, para las construcciones humanas y la predicción de desastres naturales asociados con las fuerzas geodinámicas que se describen más adelante. La geología física incluye campos como geofísica, petrología y mineralogía, está enfocada hacia los procesos y las fuerzas que dan forma al exterior de la Tierra y que actúan en su interior.

Según lo mencionado anteriormente se puede llegar a la conclusión de que la geología es el campo de la ciencia que se interesa por el origen del planeta Tierra, su historia, su forma, la materia que lo configura y los procesos que actúan o han actuado sobre él. Es una de las muchas materias relacionadas como ciencias de la Tierra, o geociencias.

1.1.2. Mineralogía

La ciencia de la mineralogía trata de los minerales de la corteza terrestre y de los encontrados fuera de la Tierra, como las muestras lunares o los meteoritos; de la identificación de esos minerales y del estudio de sus propiedades, origen y clasificación. (Villajos Carmona, y otros, 2004).

Ramas de la mineralogía

- Mineralogía descriptiva: estudia las propiedades y clasificación de los minerales individuales, su localización, sus formas de aparición y sus usos.
- Mineralogía determinativa: se dedica a la identificación de los minerales en función de sus propiedades químicas, físicas y cristalográficas.

1.1.3. Mineral

Los minerales son los bloques constructores de las rocas. Son sólidos y, como toda materia, están hechos de átomos de elementos. (Tierra, 2000).

Un mineral es una sustancia natural que se diferencia del resto por su origen inorgánico, homogeneidad, composición química preestablecida y que corrientemente ostenta una estructura de cristal, entre sus funciones principales se cuenta la de ser un componente decisivo y fundamental para la conservación y la

salud de los seres vivos, ya que su presencia resulta determinante para la actividad de las distintas células. (ABC, 2008).

Se está de acuerdo con los conceptos proporcionados anteriormente por los diferentes autores, en esta investigación se va a tomar por mineral al conjunto de sustancia sólida, natural, de origen inorgánico, de composición química. Tienen gran importancia por sus múltiples aplicaciones en los diversos campos de la actividad humana. La industria moderna depende directa o indirectamente de los minerales; se usan para fabricar múltiples productos de la moderna civilización. Así, de distintos tipos de cuarzo y silicatos, se produce el vidrio. Los nitratos y fosfatos son utilizados como abono para la agricultura. Como por ejemplo (Níquel, cobre, oro, diamante y mármol).

1.1.4 Sistema

Los sistemas se componen de otros sistemas llamándose subsistemas. En la mayoría de los casos, se pude pensar en sistemas más grandes, los cuales comprenden otros sistemas que se llaman sistema total y sistema integral. (Soto, 2003).

Según el coronel Carlos Edgardo Tejada, del ejército argentino, en su investigación referente a: “Ideas Orientadoras sobre el Sistema Logístico del Componente Ejército del Teatro de Operaciones” (Edgardo Tejada, noviembre-diciembre 2001), plantea que sistema es el conjunto de medios interconectados (objetos, seres humanos, informaciones), utilizados según un proceso dinámico con el fin de alcanzar los objetivos señalados. Se considera a un todo formado por partes, estas partes son afectadas por factores internos y externos a ese todo. Además, cada parte debe interrelacionarse e interactuar entre sí y con otras provenientes del medio externo. También, este todo debe, a su vez, interactuar como un conjunto lógico para dar respuesta a sus propios requisitos. Si cada parte de este todo, actúa e interrelaciona interna y externamente mediante plan, método y orden, se estará entonces, en presencia de un sistema integrado totalmente. La ausencia de plan, método y orden para actuar e interrelacionarse provoca que se coloque frente a un caos. Por otra parte, sistema indica orden; lo opuesto a sistema es el caos.

Carlos Edgardo Tejada muestra el concepto de sistema gráficamente mediante la siguiente imagen:

Figura #1Descripción de Sistema

Dado estos planteamientos se puede definir sistema como el conjunto de procesos o elementos interrelacionados (definiciones, nombres, símbolos) con un medio para formar una totalidad encauzada hacia un objetivo común. “Tienen tres características estructurales básicas: Los elementos que lo componen, las relaciones entre los mismos y los límites que determinan los elementos que pertenecen o no al sistema.”(Ponjuán Dante, 2004).

El sistema es el conjunto de elementos dinámicamente interrelacionados que tienen un propósito determinado. De esta definición se desprende una implicación básica; la influencia mutua entre sus componentes, es decir, que los cambios experimentados en cualquiera de sus elementos repercuten y afectan invariablemente al resto, para modificar en parte o en todo al propio sistema. (Ramírez, 2000).

Después de haber dado un conjunto de definiciones de sistema se puede concluir que un sistema es un conjunto de procesos o elementos interrelacionados con un medio para formar una totalidad encauzada hacia un objetivo común.

1.1.5 Sistema Informático

Un sistema Informático resulta de la interacción entre los componentes físicos que se denominan Hardware y los lógicos que se denominan Software. A estos hay que agregarles el recurso humano, parte fundamental de un sistema informático. Este componente es llamado humanware. (IH, 2007).

En un sistema informático, la información es introducida a través de los periféricos de entrada, luego es procesada y mostrada por los periféricos de salida. Una simple computadora es un "**sistema informático**", ya que al menos dos componentes deben trabajar conjuntamente.

Informática: El término "informática" proviene de la fusión de los términos "INFORmación" y "autoMÁTICA". La informática es una ciencia que estudia el tratamiento automático de la información.

La Real Academia Española (RAE) de la Lengua da la siguiente definición:

"Conjunto de conocimientos científicos y técnicas que hacen posible el tratamiento automático de la información por medio de computadoras electrónicas". (RAE, 2001).

De estas definiciones se pude deducir que hay tanto una ciencia informática como unas técnicas informáticas. **Sistema informático:** Sistema de procesamiento de la información basado en computadoras.

1.1.6 Software

En el libro "Ingeniería de Software. Un enfoque práctico" (Pressman, 2007) define que: Una definición de software en un libro de texto pude tener la siguiente forma: el software se forma con 1) las instrucciones (programas de computadora) que al ejecutarse proporcionan las características, funciones y el grado de desempeño deseados; 2) las estructuras de datos que permiten que los programas manipulen información de manera adecuada; y la 3) los documentos que describen la operación y el uso de los programas.
En el libro "Ingeniería de Software" (Sommerville, 2005) define que: software es un programa de ordenador y la documentación asociada. Los productos de software se pueden desarrollar para algún cliente en particular o para un mercado general.
Según la RAE (RAE, 2001) es un conjunto de programas, instrucciones y reglas informáticas para ejecutar ciertas tareas en una computadora.

Se puede acotar que un software es una aplicación o conjunto intangible de datos y programas que operan dentro de una computadora y pueden ser utilizados por persona u otros sistemas informáticos que así lo requieran.

1.1.7 Núcleo

En informática, el núcleo (también conocido en español con el anglicismo[6] kernel, de raíces germánicas) es la parte fundamental de un sistema operativo. Es el software responsable de facilitar a los distintos programas acceso seguro al hardware de la computadora. Como hay muchos programas y el acceso al hardware es limitado, el núcleo también se encarga de decidir qué programa podrá hacer uso de un dispositivo de hardware y durante cuánto tiempo, lo que se conoce como multiplexado. Acceder al hardware directamente puede ser realmente complejo, por lo que los núcleos suelen implementar una serie de abstracciones del hardware. Esto permite esconder la complejidad, y proporciona una interfaz limpia y uniforme al hardware subyacente, lo que facilita su uso para el programador. (babylon, 1997).

Existen cuatro grandes tipos de núcleos:

- Los núcleos monolíticos facilitan abstracciones del hardware subyacente realmente potentes y variadas.
- Los micronúcleos (en inglés microkernel) proporcionan un pequeño conjunto de abstracciones simples del hardware, y usan las aplicaciones llamadas servidores para ofrecer mayor funcionalidad.
- Los núcleos híbridos (micronúcleos modificados) son muy parecidos a los micronúcleos puros, excepto porque incluyen código adicional en el espacio de núcleo para que se ejecute más rápidamente. Son los que reciben o dan salida a señales analógicas que son procesadas digitalmente. Esto puede realizarse gracias a los conversores analógicos/digitales que, como su nombre indica, convierte señales analógicas a digitales.
- Los exonúcleos no facilitan ninguna abstracción, pero permiten el uso de bibliotecas que proporcionan mayor funcionalidad gracias al acceso directo o casi directo al hardware.

[6]Anglicismo: Giro o modo de hablar propio de la lengua inglesa.

1.1.8 Componente

Existen varias definiciones de componentes realizadas por expertos que han sido los encargados del desarrollo de las tecnologías, ellos han tomado como base la programación orientada a objeto y el modelado a través de UML:

Un componente de software es una unidad de composición con interfaces contractualmente especificadas y explícitas sólo con dependencias dentro de un contexto. Un componente de software puede ser desplegado independientemente y es sujeto a la composición de terceros. (Szyperski, 1998).

En el libro “Ingeniería de Software. Un enfoque práctico” (Pressman, 2007) se refiere que: un componente es un bloque de construcción modular para el software de cómputo. De manera más formal, la especificación unificada del lenguaje de modelado de OMG define un componente como ´´una parte modular, desplegable y reemplazable de un sistema que encapsula implementación y expone un conjunto de interfaces´´.

A partir de los conceptos expresados anteriormente se concluye que se está de acuerdo con las definiciones planteadas por Pressman y Szyperski referente a un componente, el cual es una parte no trivial, casi independiente y reemplazable de un sistema que llena claramente una funcionalidad dentro de un contexto en una arquitectura bien definida. Un componente se conforma y provee la realización física por medio de un conjunto de interfaces.

1.1.9 *Plugin*

En castellano se le llama complemento. Programas que se agregan a un navegador web para realizar funciones determinadas. Es una ampliación de las funciones del navegador. Esta aplicación adicional (normalmente muy específica) es ejecutada por la aplicación principal e interactúan por medio de la API[7]. Resultan muy prácticos ya que permiten expandir las posibilidades de un programa, de forma que no afecte a lo ya instalado. La forma más común en que un *plugin* se manifiesta, es cuando se intenta abrir un archivo que tiene una extensión que el sistema no tiene instalada. Normalmente se pone a disposición del usuario un enlace para poder conseguir ese complemento. (Hooping, 2009).

[7] API: Interface de Programación de Aplicaciones.

Son pequeñas aplicaciones que se incorporan a los navegadores y permiten que estos presenten posibilidades multimedia. El concepto *"plugin",* que desde ahora lleva por nombre módulo, es originario de Netscape, creado para sus navegadores. Microsoft, para Explorer, utiliza el término de controles Active X, aunque, en realidad el funcionamiento es muy similar. (Animación, 2000).

Un *plugin* (también llamado greffon en francés) es un software que permite cambiar, mejorar o agregar funcionalidades en SPIP[8]. Está escrito específicamente para SPIP y respeta todo un formalismo (una API) que les permite interactuar. El *plugin* debe su nombre al inglés *toplug* (enchufar) porque debe ser muy fácil enchufarlo con SPIP, pero también desenchufarlo. La existencia de *plugin* responde principalmente a la necesidad de evitar una hipertrofia del «núcleo» de SPIP en particular por razones de mantenimiento al mismo tiempo que se facilitan ampliamente las posibilidades de personalización avanzada de su funcionamiento. (SPIP, 2008).

Se puede concluir que los *plugins* son paquetes o clases de código implementados y se conectan de forma dinámica en tiempo de ejecución al sistema existente, mejorando las prestaciones del mismo. Además pueden ser empaquetados en componentes que cumplen un conjunto de funcionalidades que el núcleo de una aplicación informática puede operar mediante una interfaz bien definida para atender o dar respuesta a una petición del usuario del sistema.

1.2. APLICACIONES EXISTENTES EN EL CAMPO DE LA MINERÍA

En la actualidad en el mercado existen al menos tres soluciones que informatizan los procesos de la industria minera. Uno de ellos está representado en compañías como GEMCOM, SURPAC y DATAMINE, cuyas principales características son ofrecer soluciones globales, capaces de cubrir desde el área de la geología hasta el control operacional de una faena en producción, pasando por el modelado, la estimación de recursos, la optimización y el diseño. Cuba para desarrollar e incrementar la producción de minerales utiliza dos software, los cuales llevan el mismo nombre de su empresa, describiéndose a continuación.

[8] SPIP: sistema de publicación para Internet.

GEMCOM

Desde su fundación en 1985, GEMCOM ha estado a la vanguardia de la innovación. En la actualidad es uno de los proveedores mundiales de soluciones de software de minería. Proporciona un conjunto de servicios informativos y profesionales sobre la ingeniería y análisis de negocios; gestión de programas e implementación de soluciones mineras. (Chilena, 2010).

En geología y planificación cuenta con Surpac, sistema que mejora la eficiencia y la precisión a través de sus aplicaciones, gráficos 3D y la habilidad de automatizar tareas y flujos de trabajo. Otra de las soluciones de las compañías es MineSched, para la programación en minería subterránea y el trabajo abierto. Esta herramienta única permite a los planificadores probar múltiples escenarios rápidamente y generar ajustes a las planificaciones en un corto tiempo.

SIMUVIMA proporciona las capacidades adecuadas para el trabajo a cielo abierto y los profesionales de minería subterránea en la exploración, modelado, diseño de la mina, la planificación a largo plazo y la programación de la producción.

DATAMINE

Fundada en 1981, dedicada a la provisión de software especializado y servicios para la industria minera a través del mundo. Presenta gran flexibilidad al sistema, con un sistema de archivos que conforman una base de datos abierta y relacional, además de una estructura modular que permite su ampliación o cambios. Es aplicable a todas las áreas del quehacer minero, permitiendo obtener grandes avances en la industria minera del Hierro, del Oro, Níquel, Fosfatos, Diamantes, Cobre, Bauxita, Carbón, Lignite, Platino, Petróleo, y otros minerales. Los usos más comunes del sistema son; la captura y análisis de la información, exploración, geología, geoquímica, mecánica de rocas, topografía, modelado geológico, diseño de mina a cielo abierto y subterráneas, planeamiento minero y áreas relacionadas a los estudios ambientales.

De esta forma, las alternativas propuestas por la compañía para suministrar soluciones efectivas a estos exigentes requisitos cuentan con todas las características fundamentales para asegurar el éxito del negocio minero, tales como Integrabilidad, Versatilidad, Potencialidad y Confiabilidad, permitiendo un

trabajo rápido, eficiente, que considera el riesgo asociado, generando resultados auditables y repetibles a lo largo del tiempo, o en cualquier sitio de la compañía. Entre estos productos destaca el software DATAMINE Studio. (mineria, 2000).

Beneficios que originaron el uso de estas aplicaciones en la minería

GEMCOM y DATAMINE trajeron consigo grandes beneficios en la actividad minera de Cuba, desarrollando todos los modelos, caracterización y estudio minero-geológico de las principales actividades como el Níquel, cobre y oro.

Algunos alcances que se obtuvieron:

1. Conformación de modelos geológicos y de leyes 2D y 3D que facilitan un acercamiento más intuitivo y natural a los geólogos y planificadores implicados.
2. Utilización de modelos digitales del terreno aunque no siempre de buena calidad.
3. Mayor capacidad de validación de los modelos creados.
4. Posibilidad de validación de los errores de estimación y con ello del riesgo asociado al modelo creado.

Motivos por los cuáles se dejaron de usar estas aplicaciones

1. Adquisición de licencias de software de terceras compañías comenzando con GEMCOM y más tarde SURPAC, DATAMINE.
2. Impusieron una dependencia tecnológica.
3. Costoso pago de versiones, licencias, capacitación y servicios.

1.3. ESTADO ACTUAL DEL NEGOCIO

1.3.1. FUNDAMENTACIÓN DEL PROBLEMA

La adquisición de software propietarios del mercado, la premura de los empresarios ante los compromisos que planteaba la apertura de las negociaciones con capital extranjero y el mito imperante en el mundo minero, de que la marca del software garantizaba la calidad de la estimación, inclinó el vector de la decisión hacia la adquisición de licencias que comenzando con GEMCOM, SURPAC, DATAMINE y otros fueron adquiridos por entidades nacionales o introducidos por los socios extranjeros en las empresas mixtas formadas para el níquel y otros minerales, que si bien respondieron a aquellas urgencias impusieron una dependencia tecnológica que además del oneroso pago de versiones, licencias, capacitación y servicios se comprometieron con el derrotero futuro de las compañías productoras de estos sistemas.

Los elementos expuestos anteriormente permiten constatar que la solución a la problemática minera es la creación de un software cubano de minería. Siendo este el objetivo principal del proyecto de Minería. En esta investigación se le va a dar solución a una parte del sistema que se encuentra recogido en el problema a resolver: ¿Cómo obtener la representación técnica del núcleo del sistema GeolMin de forma tal que permita crear su diseño?

1.3.2. OBJETO DE AUTOMATIZACIÓN

Es preciso tener en cuenta los aspectos técnicos que recoge todo núcleo de un sistema informático, desde las funcionalidades básicas hasta las más complejas, la forma de gestionar los componentes y *plugin* logrando así un núcleo conformado por componentes reutilizables para su utilización por el sistema y en otros similares.

1.3.3. FUNDAMENTACIÓN DEL OBJETIVO

Con el diseño de la propuesta de los casos de uso y requisitos del núcleo del sistema GeolMin del proyecto Minería, se contará con un conjunto de modelos, documentos, requisitos funcionales y

descripciones del núcleo que serán la base para el futuro diseño e implementación del núcleo del sistema del proyecto de Minería que será desarrollado por el departamento de Geoinformática de la facultad 6.

» Primero: Lograr que los desarrolladores del proyecto utilicen la información que se va a generar en esta investigación para futuras mejoras del núcleo del sistema GeolMin.

» Segundo: Que los desarrolladores dominen los aspectos y conceptos básicos del núcleo del sistema GeolMin.

1.3.4. PROPUESTA DE SOLUCIÓN

Después de haber realizado un análisis de los sistemas y conceptos asociados al dominio del problema, se determinó la situación actual sobre el objeto de estudio que tiene este trabajo, se consideró la necesidad de diseñar las funciones básicas del núcleo del sistema que soporte las funcionalidades de los nuevos componentes que se desarrollaron por otros desarrolladores del proyecto.

1.4. TENDENCIAS Y TECNOLOGÍAS A UTILIZAR

Para garantizar la calidad del proceso de desarrollo de software es necesario contar con una base tecnológica como lenguaje de modelado, herramientas, metodologías de desarrollo de software que resuelvan el problema planteado en la investigación.

1.4.1. LENGUAJE DE MODELADO

UML se ha convertido en la notación visual estándar para el modelado orientado a objeto. Comenzó como una iniciativa de Grady Booch y Jim Rumbaugh en 1994 para combinar las notaciones visuales de sus dos populares métodos, los métodos de Booch y OMT (*Object Modeling Technique*). Más tarde se les unió Ivar Jacobson, el creador del método Objectory, y el grupo comenzó a ser conocido como los tres amigos.

UML es un lenguaje para especificar, visualizar construir y documentar los artefactos de los sistemas software, así como para el modelado del negocio y otros sistemas no software. (Larman, 2000).

UML fue adoptado en 1997 como estándar por el OMG (*Object Management Group*, organización que promueve estándares para la industria), y continúa siendo refinado en nuevas versiones.

Algunas de las propiedades de UML como lenguaje de modelado son:(Rumbauhg, y otros, 2000)

1. Es un lenguaje de representación Visual.
2. Remplaza a decenas de notaciones empleadas por otros lenguajes.
3. Modela estructuras complejas.
4. Permite combinar diversos elementos gráficos y crear diagramas.
5. Permite visualizar, especificar, construir, y documentar los artefactos que se crean durante el proceso de desarrollo.
6. Organiza los diagramas en grupos estáticos, comportamiento y de implementación.
7. Las estructuras más importantes que soportan tienen su fundamento en las tecnologías orientada a objeto, tales como objetos, clase, componentes y nodos.

Por decisión del equipo de arquitectura del proyecto y teniendo en cuenta la explicación anterior se llegó a la conclusión que el lenguaje de modelado a utilizar en el desarrollo de la investigación es UML 2.1, porque es un estándar de la industria del software, pero no solo de la industria sino en general de cualquier industria que requiera la construcción de modelos como condición previa para el diseño y posteriormente para la construcción de prototipos. Estos modelos ayudan a comprender mediante gráficos, modelos, diagrama o pequeñas estructuras el comportamiento y la organización de una aplicación informática, además de proporcionar un lenguaje común entre los integrantes del equipo y de otros proyectos de desarrollo.

1.4.2. METODOLOGÍA DE DESARROLLO UTILIZADA

Uno de los principales problemas en la actualidad en el desarrollo de software es seleccionar la metodología más adecuada que posibilite obtener los resultados óptimos que se desean; o sea, cómo

trabajar eficientemente evitando las catástrofes que conllevan al fracaso de un gran porcentaje de proyectos a nivel mundial. Una metodología tiene como principal objetivo aumentar la calidad del software y guiar las actividades que se realizan dentro de un proyecto de desarrollo de software.

Actualmente el proceso de desarrollo de software ha tenido que enfrentar la construcción de sistemas de software mucho más complejos y grandes cada día. Debido al auge de las computadoras y a la vez el aumento del rigor de los usuarios, esto ha traído un rápido incremento en Internet para el intercambio de todo tipo de información.

James Rumbaugh, Grady Booch e Ivar Jacobson, autores de "El proceso unificado de desarrollo de software", opinan que *"El problema del software se reduce a la dificultad que afrontan los desarrolladores para coordinar las múltiples cadenas de trabajo de un gran proyecto de software".* (Pressman, 2007).

El proceso unificado de desarrollo, RUP (ver Anexo #1), es el resultado de la evolución e integración de diferentes metodologías de desarrollo de software que permite sacar el máximo provecho de los conceptos asociados a la orientación a objetos y al modelado visual. Cuenta con las mejoras prácticas del modelo de desarrollo de un software en particular. (Pressman, 2007).

1. Desarrollo de software de forma iterativa.
2. Manejo de requisitos.
3. Utiliza arquitectura basada en componentes.
4. Modela el software de forma visual, usando UML.
5. Verifica la calidad del software.
6. Controla los cambios.
7. Dirige las tareas de cada desarrollador por separado y del equipo como un todo.
8. Especifica los artefactos que deben desarrollarse en cada fase de desarrollo del software.

RUP:(Flores, 2011)

- Forma disciplinada de asignar tareas y responsabilidades en una empresa de desarrollo (quién hace qué, cuándo y cómo).
- Requiere un grupo grande de programadores para trabajar con esta metodología y es el proceso de desarrollo más general de los existentes actualmente.
- Es un marco del proyecto que describe una clase de los procesos que son iterativos e incrementales.
- Los procesos estiman tareas y horario del plan midiendo la velocidad de iteraciones concerniente a sus estimaciones originales.
- Proporciona muchas ventajas sobre XP le da énfasis en los requisitos y el diseño.
- Realiza un levantamiento exhaustivo de requerimientos.
- Busca detectar defectos en las fases iniciales.
- Intenta reducir al número de cambios tanto como sea posible.
- Realiza el Análisis y diseño, tan completo como sea posible.
- Diseño genérico, intenta anticiparse a futuras necesidades.
- Las necesidades de clientes no son fáciles de discernir.
- Existe un contrato prefijado con los clientes.
- El cliente interactúa con el equipo de desarrollo mediante reuniones a diferencia de la metodología XP que el cliente es parte del equipo.
- La ventaja principal es que se basa todo en las mejores prácticas que se han intentado y se han probado en el campo. (En comparación con XP que se basa en las prácticas inestables que utilizaron juntas se evita que se derribe).

Define nueve disciplinas a realizar en cada fase del proyecto:

- Modelado del negocio
- Análisis de requisitos
- Análisis y diseño
- Implementación
- Prueba
- Distribución
- Gestión de configuración y cambios
- Gestión del proyecto
- Gestión del entorno

De acuerdo a las condiciones, características del proyecto y del personal de desarrollo, el equipo de arquitectura decidió utilizar la metodología RUP como guía del proceso de desarrollo. Por contar con los mejores reconocimientos a nivel mundial en las empresas desarrolladoras de software, los roles están bien definidos y el grupo de desarrollo presenta un nivel de conocimiento y familiarización alto. Además de ser una metodología que está muy bien documentada, basada en el estándar UML y que presenta características como son: dirigido por caso de uso, iterativo e incremental y centrado en la arquitectura.

1.4.3. HERRAMIENTA DE MODELADO

En la actualidad con el avance en el desarrollo de las tecnologías de la informáticas y las comunicación, cada vez los clientes son más exigente en sus pedidos y los productos a desarrollar cada vez son más complejos y requieren el uso de aplicaciones integrales que proporcionen un desarrollo con calidad, en un menor tiempo y que cumpla con las necesidades o expectativas de los interesados, para lograr lo

anteriormente planteado se utilizan las herramientas CASE[9], con el fin de automatizar los aspectos clave de todo el proceso de desarrollo de un sistema, desde el principio hasta el final.

Facilita a los ingenieros de software diseñar, integrar y modelar visualmente los distintos diagramas que se generan a lo largo del desarrollo del software. Las herramientas CASE liberan al programador de parte de su trabajo y aumentan la calidad del programa a la vez que disminuyen sus posibles errores. Se incluyen una serie de herramientas, lenguajes y técnicas de programación que permiten la generación de aplicaciones de manera semiautomática.

Visual Paradigm es una herramienta profesional que soporta el ciclo de vida completo del desarrollo de software: negocio, requisitos, análisis y diseño orientado a objeto, construcción, pruebas y despliegue. Ayuda a una más rápida construcción de aplicaciones de calidad, mejores y a un menor coste. Permite crear todos los tipos de diagramas basado en UML en su versión 2.1. (Sitio de descargas de software, 2007).

Lista de características:

- Soporte de UML versión 2.1.
- Diagramas de Procesos de Negocio – Proceso.
- Modelado colaborativo con CVS y Subversión (Trabajo en equipo y a distancia).
- Ingeniería inversa de código a diagrama.
- Ingeniería inversa Java, C++, Esquemas XML, .NET, exe/dll, CORBA IDL.
- Editor de Detalles de Casos de Uso.
- Diagramas EJB - Visualización de sistemas EJB.
- Generación de código y despliegue.
- Soporte ORM - Generación de objetos desde la base de datos.

[9] CASE: Ingeniería de Software Asistida por Computadoras.

- Generación de bases de datos. Transformación de diagramas de Entidad-Relación en tablas de base de datos.
- Ingeniería inversa de bases de datos.
- Generador de informes para generación de documentación.

Una vez, adoptado como lenguaje de modelado UML y metodología de desarrollo RUP, se seleccionó por el equipo de arquitectura del proyecto una herramienta CASE que soportara UML y que cubriera el desarrollo de software desde el inicio hasta el final como lo propone RUP; por lo que se ha elegido Visual Paradigm. Esta herramienta es de vital importancia para el desarrollo de la actividad realizada por el rol de analista que defiende esta investigación ya que proporciona además de las ventajas anteriormente descritas, realizar un diseño guiado por casos de uso, permitir el trabajo en equipo realizando una rápida construcción ordenada de la aplicación con calidad, modelar utilizando como soporte UML 2.1, además es multiplataforma y el equipo de desarrollo presenta un dominio en uso de la herramienta. Permite establecer una trazabilidad real entre el modelo (análisis y diseño) y el código ejecutable, facilita el desarrollo de un proceso cooperativo en el que todos los agentes tienen sus propias vistas de información.

1.4.4. HERRAMIENTA DE GESTIÓN DE REQUISITOS

Los productos de software son cada vez más complejos y grandes ya que las necesidades del cliente son más complejas y se convierten en requisitos que el sistema debe cumplir como por ejemplo graficar información en forma de barra, crear reportes en formatos pdf,[10] registrar información hasta autenticar los usuarios en el sistema.

Todas estas características que el sistema debe cumplir hacen referencia a los requisitos funcionales (RF), para lograr un software con calidad, altamente competitivo, además que cubra las necesidades y expectativas de los interesados.

[10] pdf: formato de documento portable.

Para lograr cumplir lo planteado, el equipo de arquitectos del proyecto decidió utilizar la herramienta de gestión de requisitos Rational RequisitePro. Es una herramienta que proporciona un apoyo sustancial para el desarrollo de las actividades realizadas por el rol de analista que contribuyen a obtener y gestionar los requisitos funcionales (RF) con eficiencia y calidad, además de brindar la facilidad de tracearlos durante las disciplinas de trabajo propuestas por RUP fundamentalmente (Requisitos, Análisis y Diseño, Implementación y Prueba).

1.4.5. HERRAMIENTA O TÉCNICAS DE ESTIMACIÓN

Para cumplir con las expectativas del cliente y los cronogramas de trabajo se debe utilizar al menos una de las técnicas de estimación, permitiendo estimar cuánto va a durar el desarrollo de producto. En este tipo de actividad vale mucho la experiencia que se tenga en desarrollar aplicaciones informáticas. Se investigaron tres técnicas de estimación y se seleccionó una de ellas el análisis de puntos de casos de uso. A continuación se describe en qué consiste cada una de las técnicas de estimación.

Método de Estimación UCI para el desarrollo de software: En la Universidad de las Ciencias Informáticas (UCI) se ha definido un método de estimación. El desarrollo del método parte de la necesidad que existe en la universidad de estimar el tamaño, costo y esfuerzo requerido para desarrollar un producto de software. Los métodos que existen actualmente no abarcan todas las características de producción en la UCI. Actualmente la universidad no cuenta con una base histórica, se tuvo en cuenta los datos dispersos de algunos proyectos y la evaluación de algunos factores que, según criterio de expertos, pueden influir en las estimaciones del proyecto. Este método se basa en diferentes métricas permitiendo calcular el tamaño, costo, esfuerzo, tiempo y cantidad de recursos humanos; específicamente para calcular el tamaño se identifican la cantidad de módulos que potencialmente tendrá el sistema a desarrollar y su clasificación. Estos módulos engloban una serie de funcionalidades y estarán compuestos por una determinada cantidad de CU que será estimada por el método. (Eva, 2005).Este método se encuentra en proceso de refinamiento a partir del levantamiento de información real sobre los proyectos productivos por tal motivo no se utilizará en esta investigación.

Análisis de Puntos de Función y COCOMO II: Este método está basado en ecuaciones matemáticas que permiten calcular el esfuerzo a partir de ciertas métricas de tamaño estimado, como el Análisis de Puntos de Función y las líneas de código fuente (en inglés *SLOC*, *Source Line Of Code*).(Estudio de Técnicas

basadas en Puntos de Función para la Estimación, 2005). Este método no será utilizado porque hay que contar con una cantidad de líneas de código fuente (*SLOC*) a implementar por casos de uso. Para lograr este indicador se debe tener experiencia en el desarrollo de aplicaciones.

Análisis de Puntos de Casos de Uso: Es un método propuesto originalmente por Gustav Karner de *Objectory AB* (Academic, 2000-2010), y posteriormente refinado por muchos otros autores. Se trata de un método de estimación del tiempo de desarrollo de un proyecto mediante la asignación de "pesos" a un cierto número de factores que lo afectan, para finalmente, contabilizar el tiempo total estimado del proyecto y a través de la cantidad de personal que se tenga en el proyecto se cuenta con una métrica que permite brindar un aproximado de tiempo de desarrollo por horas-hombre.

1.5. TENDENCIA DEL ROL DE ANALISTA DE SOFTWARE

1.5.1. FUNDAMENTACIÓN DEL ROL DE ANALISTA DE SOFTWARE

Los analistas cumplen un rol vital en el proceso de desarrollo, son responsables de investigar, planear, coordinar y recomendar opciones de software para cumplir los requisitos del usuario. Para ser exitoso debe adquirir cuatro habilidades:(Argudo Vásconez, y otros, 2010)

- Analítica: permiten entender a la organización y sus funciones, las cuales le ayudan a identificar oportunidades, analizar y resolver problemas.
- Técnica: ayudan al analista de sistemas a entender el potencial y las limitaciones de las tecnologías de la información. El analista de sistemas debe ser capaz de trabajar con varios lenguajes de programación, sistemas operativos, y plataformas hardware de computadoras.
- Gerencial: ayudan al analista de sistemas a administrar proyectos, recursos, riesgos, y cambio.
- Interpersonal: ayudan al analista de sistemas a trabajar con los usuarios finales así como con analistas, programadores, y otros profesionales de los sistemas, es decir, son un enlace entre el equipo de desarrollo y los usuarios.

1.5.2. ARTEFACTOS CONSTRUIDOS

» Documento Modelo Negocio: Este documento recoge todo lo relacionado con el negocio desde las reglas, actores, trabajadores, casos de uso, diagrama de casos de uso, diagrama de actividad y modelo de objetos.

» Documento Modelo Dominio: Este documento recoge el diagrama de clases del dominio, descripción de clases que participan y las restricciones del modelo.

» Documento Especificación de Requisitos Funcionales: Este documento recoge las descripciones de los requisitos funcionales y no funcionales del sistema.

» Documento Modelo Sistema: Este documento recoge el diagrama de casos de uso del sistema, actores, descripción textual de los casos de uso, diagrama de paquetes de casos de uso y prototipo de interfaz de usuario.

» Documento Modelo Análisis: Este documento recoge clases de análisis, diagramas de clases de análisis y los diagramas de interacción (Secuencia y Colaboración).

» Documento de Matriz de Trazabilidad: Este documento recoge los requisitos funcionales y su relación con su especificación, casos de uso del sistema, realizaciones de casos de uso del análisis y diseño, en la clase de código fuente que lo implementa y con las pruebas realizadas.

CONCLUSIONES PARCIALES

A partir del estudio realizado se arribó a las siguientes conclusiones: mediante el estudio de los conceptos asociados al dominio y el objeto de estudio se logró un mayor entendimiento del problema. .El empleo de la Metodología RUP permitió obtener los artefactos necesarios para la modelación de la solución informática y su grado de detalle en cada una de las Fases y Flujos de Trabajo establecidos; contribuyendo a la clarificación y comprensión de los Modelos. Su característica de ser iterativo – incremental, facilitó la organización, obtención y perfeccionamiento de los mismos. Se tomó UML como la mejor variante para modelar, especificar, describir y documentar artefactos que se desarrollen durante el proyecto, además de contar con una base tecnológica de herramientas multiplataforma que mejoran el proceso de desarrollo de software.

CAPÍTULO #2
CARACTERÍSTICAS DEL SISTEMA

INTRODUCCIÓN

En este capítulo se abordarán temas específicos de Ingeniería de software, es decir artefactos generados en la fase de inicio, elaboración específicamente en los flujos modelado del negocio, levantamiento de requisitos, análisis y diseño, que permitirá comprender las características del núcleo del sistema GeolMin. Además de la especificación de los requisitos funcionales (RF) y casos de uso del sistema, así como la matriz de trazabilidad, el modelo de análisis y diagrama de interacción.

2.1. PROPÓSITO DEL MODELADO DE NEGOCIO

Un sistema por pequeño que sea generalmente es complicado, por eso es necesario dividirlo en piezas si se pretende comprender y gestionar su complejidad, esas piezas se pueden representar a través de modelos que permitan abstraer sus características esenciales. Existen varias técnicas como modelo de negocio, modelo de procesos o modelo de dominio, siendo esta última la que se desarrollará en la investigación producto a que no se tienen identificados los procesos del negocio, actores y trabajadores.

Por lo anteriormente expresado un modelo de dominio es: una representación visual de los conceptos u objetos del mundo real en un dominio de interés. Este modelo agrupa los conceptos de un dominio y permite comprender el problema y establecer conceptos comunes. Además ayuda a comprender los conceptos que utilizan los usuarios, los conceptos con los que trabajan y con los que deberá trabajar la aplicación.

2.2. MODELO DE DOMINIO

2.2.1. DESCRIPCIÓN DE LOS CONCEPTOS

Concepto de dominio	Descripción
Mineros	Personal que opera con las aplicaciones informáticas para la explotación, planificación y

	control de depósitos minerales.
Aplicaciones Informáticas	Son las aplicaciones que utilizan los geólogos mineros para controlar, planificar y explotar depósitos de minerales.
Adquisición	Consiste en la toma de muestras del mundo real para generar datos que puedan ser manipulados por aplicaciones informáticas.
Procesamiento	Es la aplicación sistemática de una serie de operaciones sobre un conjunto de datos mineros mediante software especializado o aplicaciones informáticas sobre ordenadores.
Visualizar	Se centra en la representación de las características de los depósitos de minerales.
Datos	Son sucesos que describen hechos y entidades.
2 Dimensiones	Es la animación que representa elementos y características de minerales que se despliegan en un sistema de coordenadas de dos dimensiones, es decir sobre el plano.
3 Dimensiones	Es mucho más compleja que la bidimensional y requiere por lo general una gran potencia de cálculo para ser elaborada con calidad, y un elevado tiempo de diseño para producir efectos realistas de movimiento de objetos que representan estructuras mineras.
Modelos terrestres	Es la representación de un área terrestre donde se encuentran ubicados objetos minerales.

Tabla #1 Descripción de los conceptos del dominio

2.2.2. DESCRIPCIÓN DE LAS RELACIONES

Relaciones	Descripción
Asociación	Es una relación estructural que describe una conexión entre objetos.

Tabla #2 Descripción de las relaciones utilizadas en dominio

2.2.3. DIAGRAMA DE CLASES DEL DOMINIO

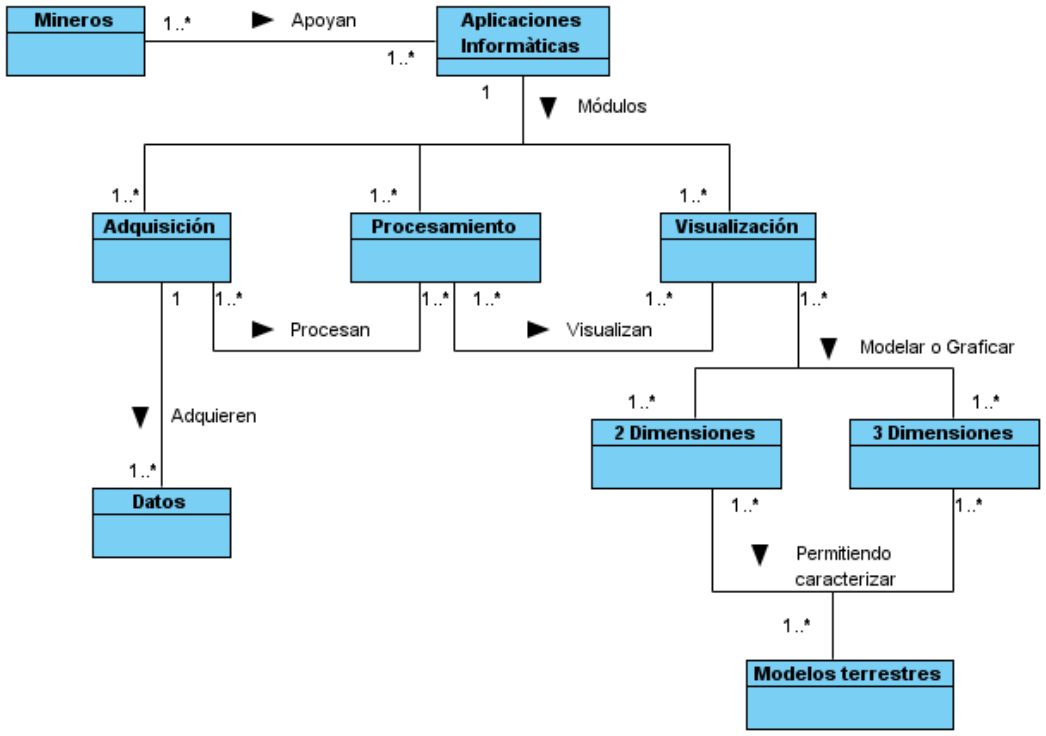


Figura #2 Diagrama de clases del dominio

2.3. ESPECIFICACIÓN DE LOS REQUISITOS DE SOFTWARE

Requisitos funcionales (RF): describen lo que el sistema debe hacer en términos de funcionalidades. Especifican la manera en que éste debe reaccionar a determinadas entradas.

Requisitos no funcionales (RNF): son aquellos que se refieren directamente a las propiedades emergentes del sistema como la fiabilidad, el tiempo de respuesta, la capacidad de almacenamiento entre otros.

2.3.1. DESCRIPCIÓN DE LOS REQUISITOS FUNCIONALES

RF1: Iniciar la Aplicación.

RF2: Cargar configuración base.

RF3: Cargar plugin.

RF4: Cargar I18N.

RF5: Configurar barra de tarea.

RF6: Configurar menú.

RF7: Configurar hojas de estilo.

RF8: Configurar estilo de aplicación.

RF9: Configurar I18N.

RF10: Crear proyecto.

RF11: Abrir proyecto.

RF12: Guardar proyecto.

RF13: Guardar como.

RF14: Modificar propiedades del proyecto.

RF15: Pegar objeto de información.

RF16: Rehacer objeto de información.

RF17: Deshacer objeto de información.

RF18: Cortar objeto de información.

RF19: Copiar objeto de información.

RF20: Mostrar contenido de ayuda.

RF21: Actualizar sistema.

RF22: Exportar configuración.

RF23: Cerrar proyecto.

RF24: Salir Aplicación.

2.4. PROPUESTA DE SOLUCIÓN

Con la realización del diagrama de casos de uso del sistema se evidencia una relación entre los actores y requisitos del sistema. Donde los actores representan usuarios y otras aplicaciones que interaccionan con el sistema. Los requisitos son una condición o capacidad que debe estar presente en un sistema o componentes de sistema para satisfacer un contrato, estándar, especificación u otro documento formal. A los casos de uso identificados se le aplicaron patrones permitiendo estructurar los diagramas, conllevando a obtener una organización y reflejar con mayor precisión las necesidades reales del cliente.

2.4.1. DESCRIPCIÓN DE LOS ACTORES DEL SISTEMA

Los actores no representan solamente a personas, estos también pueden ser otros sistemas o hardware externos que interactúan con el sistema. El modelo de casos de uso del módulo Núcleo del proyecto Minería incluye un actor, el actor Minero el cual se describe a continuación.

Actores	Descripción
Minero	Persona que opera con las aplicaciones informáticas para la explotación, planificación y control de depósitos minerales.

Tabla #3 Descripción de los actores del sistema

2.4.2. FUNDAMENTACIÓN DE LOS PATRONES DE CASOS DE USO UTILIZADOS

Un patrón es una solución a un problema en un contexto, codifica conocimiento específico acumulado por la experiencia en un dominio. Tiene un formato bien definido, donde sus atributos frecuentes son: nombre del patrón, problema que resuelve, solución propuesta, contexto, ejemplos.

Los patrones dan nombre y forma a heurísticas abstractas, reglas y buenas prácticas de técnicas orientadas a objetos. Están estrechamente relacionados unos con otros y para un mejor aprovechamiento

de estos no deben aplicarse de manera aislada. En la construcción del diagrama de casos de uso del sistema se utilizaron dos patrones:

» Inclusión concreta.

» CRUD Parcial.

2.4.3. LISTADO DE CASOS DE USO DEL SISTEMA

Los casos de uso son descripciones de un conjunto de secuencias de acciones que ejecuta un sistema y que produce un resultado observable para un actor particular. (Prof. Juan Carlos Gutiérrez Lázaro, 2008-2009).

Casos de uso del sistema

» Iniciar la Aplicación.

» Cargar configuración base.

» Cargar plugin.

» Cargar I18N.

» Gestionar Proyecto.

» Configurar barra de tarea.

» Configurar menú.

» Configurar hojas de estilo.

» Configurar estilo de aplicación.

» Configurar I18N.

» Editar Proyecto.

» Mostrar contenido de ayuda.

» Actualizar sistema.

» Exportar configuración.

» Salir Aplicación.

2.4.4. DIAGRAMA DE CASOS DE USO DEL SISTEMA

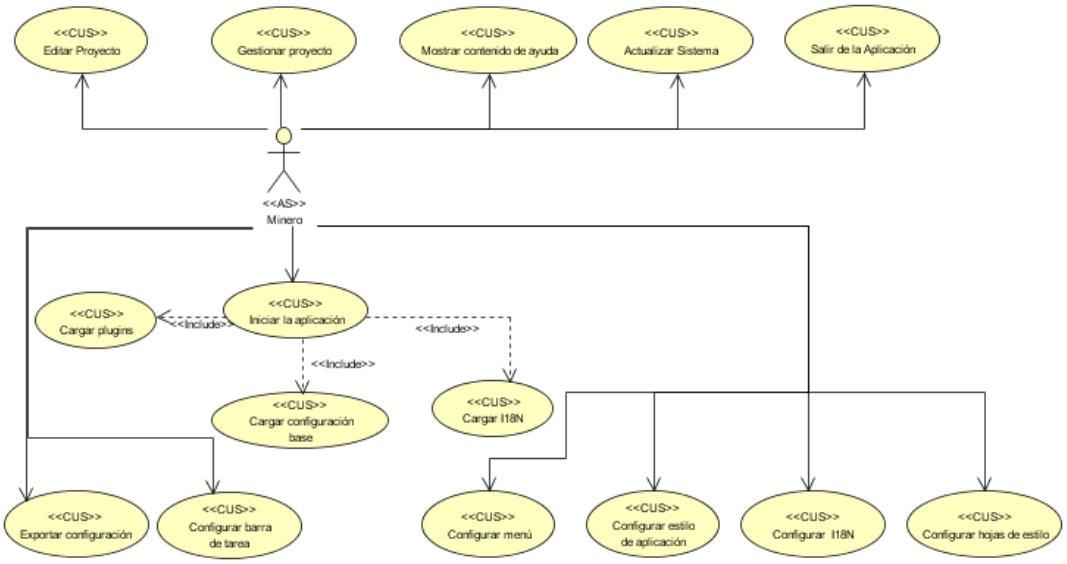


Figura #3 Diagrama_CUS

2.4.5. DIAGRAMA DE PAQUETES

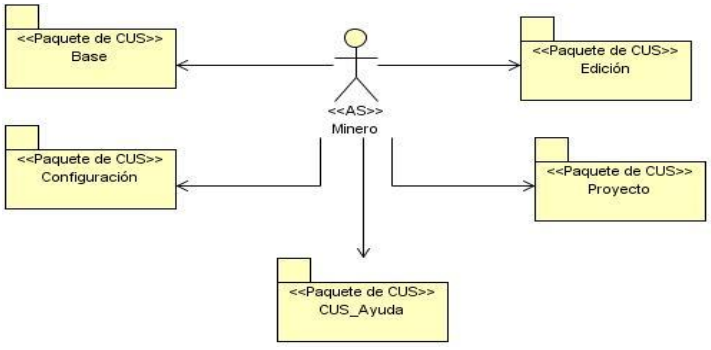


Figura #4 Diagrama de paquetes

2.4.6. DESCRIPCIÓN TEXTUAL DE LOS CASOS DE USO DEL SISTEMA

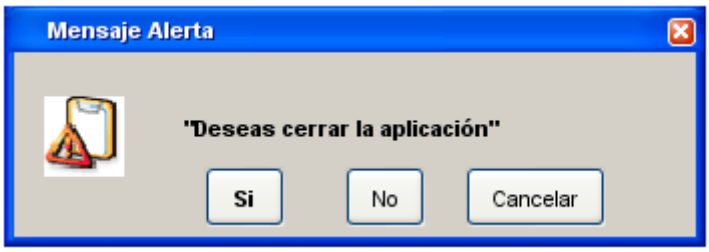
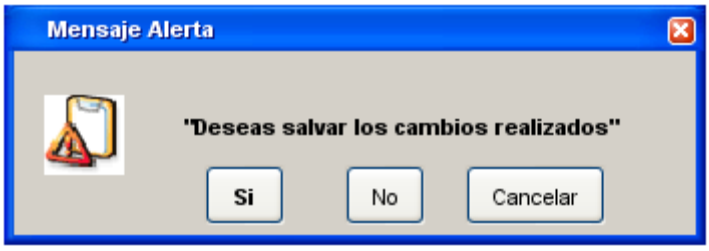
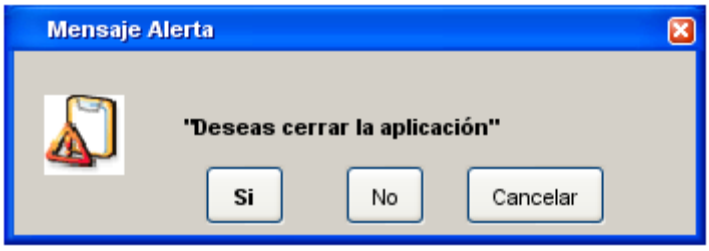
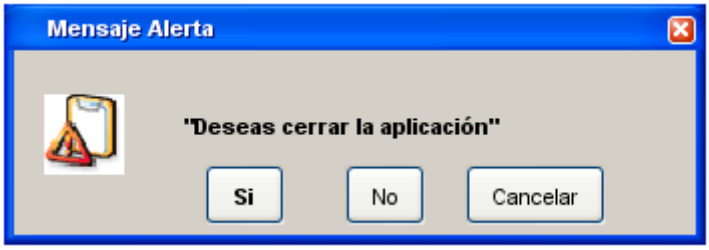
Las descripciones textuales de los casos de uso (CU) se realizaron con el objetivo de lograr una mejor comprensión de los procesos a automatizar. En ellas se describen los diferentes flujos de sucesos entre el actor y el sistema, auxiliados de prototipos de interfaz de usuario. A continuación se presentan algunas especificaciones de CU del módulo Núcleo del proyecto Minería.

Caso de Uso:	Iniciar Aplicación.
Actores:	Minero
Resumen:	El caso de uso se inicia cuando el actor decide ejecutar la aplicación y termina cuando el sistema crea la aplicación y la visualiza.
Precondiciones:	
Referencias	RF1
Prioridad	Crítico
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El actor selecciona el ejecutable de la aplicación (GeolMin.exe).	2. El sistema invoca el caso de uso Cargar Configuración Base.
	3. El sistema invoca el caso de uso Cargar los Plugin (“Dirección del directorio de plugin”).
	4. El sistema invoca el caso de uso Cargar I18N (“Dirección del directorio de Idioma”).
	5. El sistema lee la configuración y extrae el tipo de área de trabajo.
	6. El sistema configura el área de trabajo con los plugin instalados.
	7. El sistema muestra la interfaz visual de la

	aplicación y termina el CU.
Prototipo de Interfaz	
Flujos Alternos	
Acción del Actor	**Respuesta del Sistema**
Prototipo de Interfaz	
Poscondiciones	El actor accede a la aplicación.

Tabla #4 Descripción textual del CUS Iniciar Aplicación

Caso de Uso:	Salir de la aplicación.
Actores:	Minero
Resumen:	El caso de uso se inicia cuando el usuario desea salir de la aplicación y termina cuando este logra salir de la aplicación.
Precondiciones:	Que se haya ejecutado el caso de uso iniciar aplicación.
Referencias	RF25
Prioridad	Crítico
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El caso de uso se inicia cuando el actor selecciona la opción de Salir de la Aplicación.	2. El sistema muestra un mensaje de alerta. **Mensaje:** (*Deseas cerrar la aplicación*).
3. El actor selecciona el botón “Si”.	4. El sistema verifica que existen cambios en las estructuras abiertas (Ventanas, objetos, etc.).
	5. El sistema muestra un mensaje al actor. **Mensaje:** (*Deseas salvar los cambios realizados*)
6. El actor selecciona el botón “Si”.	7. El sistema localiza los datos y los guarda en los ficheros correspondientes a proyecto creado.
	8. El sistema cierra aplicación y termina el CU.

<table>
<tr><td colspan="2"> 
</td></tr>
<tr><td colspan="2"> 
</td></tr>
<tr><td colspan="2">Flujos Alternos</td></tr>
<tr><td>Acción del Actor</td><td>Respuesta del Sistema</td></tr>
<tr><td>3*a. El actor selecciona el botón “No”.</td><td>3*a. Ir a la acción 5 del sistema.</td></tr>
<tr><td colspan="2"> 
</td></tr>
<tr><td>3*b. El actor selecciona el botón “Cancelar”.</td><td>3*b. El sistema cancela la acción de salir la aplicación. Dejando la aplicación con el mismo estado.</td></tr>
<tr><td colspan="2"> 
</td></tr>
<tr><td>6*a. El actor selecciona el botón “No”.</td><td>6*a. Ir a la acción 5 del sistema.</td></tr>
</table>

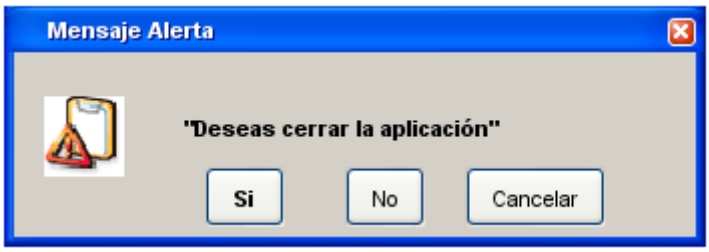
	
Poscondiciones	Aplicación cerrada.

Tabla #5 Descripción textual del CUS Salir de la aplicación

Caso de Uso:	Cargar configuración base.
Actores:	Iniciar Aplicación(include)
Resumen:	Se inicia cuando es invocado por el caso de uso iniciar aplicación. Localiza el fichero de configuración y extrae la información creando una estructura de configuración. Cierra el fichero y termina el caso de uso.
Precondiciones:	Que se haya ejecutado el caso de uso iniciar aplicación.
Referencias	RF2
Prioridad	Crítico
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
	1. El sistema verifica si el fichero existe.
	2. El sistema abre el fichero.
	3. El sistema lee la estructura del fichero y almacena la información en la estructura de configuración.
	4. El sistema cierra el fichero.
	5. El sistema devuelve la estructura y termina el caso de uso.
Prototipo de Interfaz	

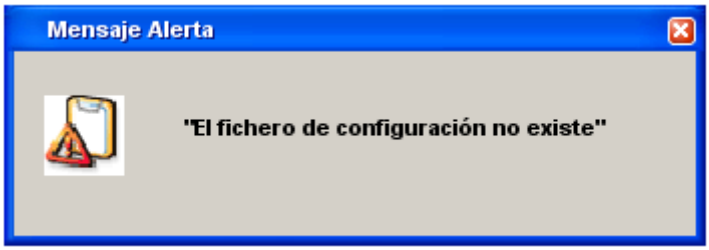
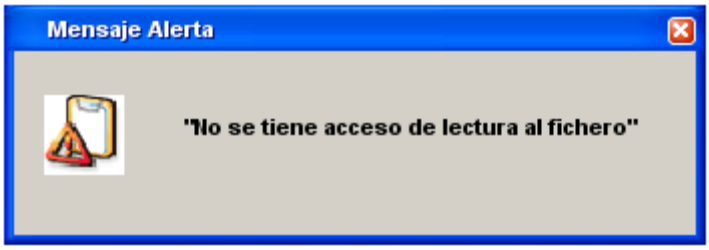
Flujos Alternos	
Acción del Actor	**Respuesta del Sistema**
	1*a. El sistema muestra un mensaje de error. **Mensaje:** (*El fichero de configuración no existe*). El sistema se cierra automáticamente.
	
	2*a. El sistema muestra un mensaje de alerta. **Mensaje:** (*No se tiene acceso de lectura al fichero*). El sistema carga la I18N por defecto.
	
Poscondiciones	Configuración cargada.

Tabla #6 Descripción textual del CUS Cargar configuración base

Caso de Uso:	Cargar los plugin.
Actores:	Iniciar Aplicación(include)
Resumen:	Se inicia cuando es invocado por el caso de uso iniciar aplicación. Localiza el directorio de los plugin, carga los que están instalados por el sistema, devolviendo un objeto del mismo y termina el caso de uso.

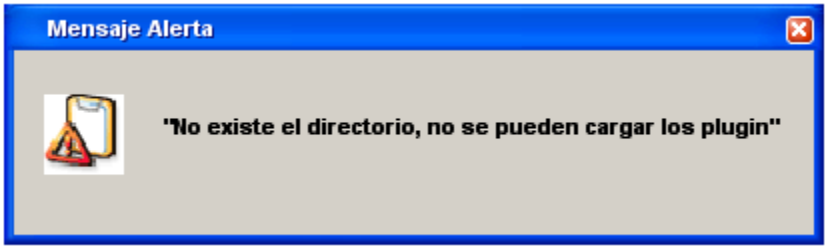
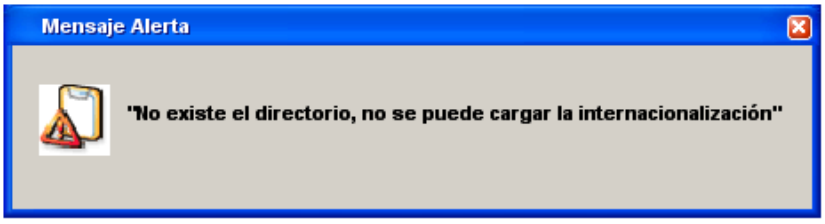
Precondiciones:	Que se haya ejecutado el caso de uso iniciar aplicación.
Referencias	RF3
Prioridad	Crítico
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
	1. El sistema localiza el directorio de los plugin.
	2. El sistema carga los plugin que están instalados.
	3. El sistema devuelve una instancia de cada plugin.
	4. Termina el caso de uso.
Prototipo de Interfaz	
Flujos Alternos	
Acción del Actor	**Respuesta del Sistema**
	1*a. Muestra un mensaje de error. **Mensaje:**(*No existe el directorio, no se pueden cargar los plugin*). El sistema se cierra automáticamente.
	
Poscondiciones	Plugin cargado.

Tabla #7 Descripción textual del CUS Cargar los plugin

Caso de Uso:	Cargar I18N.
Actores:	Iniciar Aplicación(include)
Resumen:	Se inicia cuando es invocado por el caso de uso iniciar aplicación.

	Localiza el fichero de I18N y extrae la información creando una estructura de I18N. Cierra el fichero y termina el caso de uso.
Precondiciones:	Que se haya ejecutado el caso de uso iniciar aplicación.
Referencias	RF4
Prioridad	Crítico

Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
	1. El sistema localiza el directorio de ficheros de I18N.
	2. El sistema abre los ficheros y lee la estructura de información.
	3. El sistema extrae la información creando una estructura de I18N.
	4. El sistema cierra el fichero.
	5. El sistema devuelve la estructura creada y termina el CU.
Prototipo de Interfaz	
Flujos Alternos	
Acción del Actor	**Respuesta del Sistema**
	1*a. El sistema muestra un mensaje de error. **Mensaje:**(*No existe el directorio, no se puede cargar la I18N*). El sistema carga la I18N por defecto.
	
	2*a. El sistema muestra un mensaje de alerta.

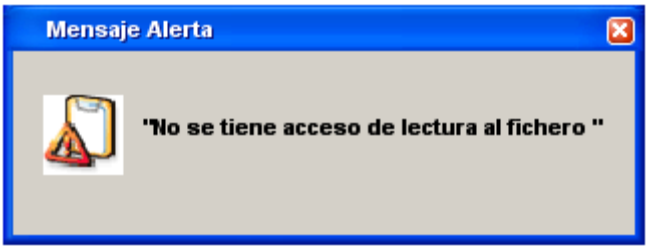
	Mensaje:(*No se tiene acceso de lectura al fichero*). El sistema carga la I18N por defecto.
	
Poscondiciones	I18N cargada.

Tabla #8 Descripción textual del CUS Cargar I18N

2.5. MATRIZ DE TRAZABILIDAD

La matriz de trazabilidad está conformada por las representaciones gráficas de las relaciones entre dos o más productos del proceso de desarrollo, generalmente identificadas en las intersecciones de líneas verticales y horizontales. Por ejemplo, para representar la relación entre los requisitos y el diseño de un componente del software. (código, 2008).

Ventajas de su utilización:

- Indica el avance en el desarrollo de los requisitos.
- Indica las dependencias entre los productos.
- Indica los productos y requisitos afectados al evaluar un cambio.
- Herramienta para verificar si los productos desarrollados son consistentes con los requisitos.
- Tiene una relación entre requisitos funcionales y casos de uso.
- Se pueden configurar y dar seguimiento a las relaciones entre requisitos para verificar que los requisitos de alto nivel están representados dentro de las especificaciones detalladas de requisitos de software.

2.6. MATRIZ DE TRAZABILIDAD ENTRE REQUISITOS FUNCIONALES Y CASOS DE USO DEL SISTEMA

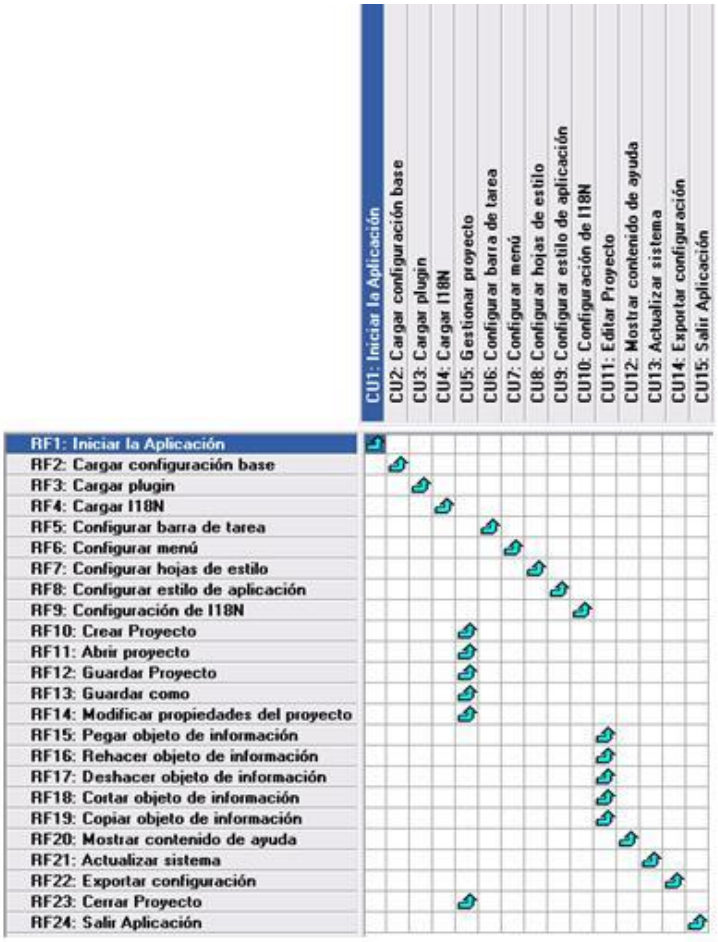


Tabla #9 Matriz de trazabilidad (RF y CUS)

2.6.1. FUNDAMENTACIÓN DEL USO DE LA HERRAMIENTA DE GESTIÓN DE REQUISITOS

En el desarrollo de esta investigación se utilizó la herramienta de Rational RequisitePro para construir la matriz de trazabilidad siendo esta una poderosa herramienta, fácil de usar e integrada a la administración de requisitos que promueve una mejor comunicación.

Rational RequisitePro permite: (IBM, 2010)

- Gestionar los requisitos de los equipos de proyecto.
- Escribir buenos casos prácticos.
- Mejorar la rastreabilidad.
- Reforzar la colaboración.
- Reducir las tareas de remodelación de los proyectos.
- Aumentar la calidad.
- Evitar repeticiones y duplicaciones gracias a la integración avanzada en tiempo real con Microsoft Word.
- Gestionar la complejidad con vistas de rastreabilidad detalladas que muestran relaciones padre-hijo.
- Reducir los riesgos de los proyectos con la visualización de requisitos que pueden verse afectados por cambios en los requisitos en sentido ascendente o descendente.
- Lograr la colaboración de equipos distribuidos geográficamente a través de una interfaz web escalable totalmente funcional e hilos de debate.
- Mejorar la productividad haciendo un seguimiento de los cambios mediante comparaciones de las versiones del proyecto con líneas base de proyecto basadas en XML.
- Ajustar los objetivos empresariales con los productos finales del proyecto mediante la integración con varias herramientas en la plataforma de desarrollo y distribución de software de IBM Rational.
- Como sistema operativo Windows.

2.7. PROPUESTA DE ANÁLISIS

El modelo de análisis se utiliza para obtener una visión del sistema y se preocupa por ver qué hace, por lo que solo se interesa por los RF, a pesar de que aquí hay un refinamiento de los requisitos, no se especifica el lenguaje de programación que se utilizará ni la plataforma en que se ejecutará la aplicación, tampoco se toma en cuenta los componentes prefabricados o reutilizables de otras aplicaciones, ya que el objetivo del análisis es comprender perfectamente los requisitos funcionales del software en un lenguaje más técnico, sin precisar, como se implementa la solución.

Ventajas de su aplicación:

- Suaviza la transición al diseño.
- Apoya el cambio a otra plataforma de programación.
- Sirve para tener una visión general de la propuesta del sistema.
- Apoya la aplicación de reingenierías a aplicaciones existentes.
- Planifica y divide el diseño e implementación en pequeños módulos.

El modelo de clases de análisis se divide en dos partes, diagrama de clases de análisis y diagrama de interacción.

Las clases del Análisis están estructuradas por Interfaz, Controly Entidad.

Interfaz:

Modela la interacción Actor – Sistema, además de la comunicación con otros sistemas o dispositivos a través de ventanas, formularios.

Control:

Coordinan el trabajo de las clases, encapsulan comportamiento de un CU y posee funciones complejas.

Entidad:

Modelan la información del sistema y el comportamiento asociado a una información.

Los diagramas de interacción se dividen en secuencia y colaboración, se recomienda que para las actividades realizadas en el análisis se debe utilizar colaboración debido a que representa un grafo donde

los nodos son los objetos y los arcos los mensajes que se mandan, en los cuales se destaca la organización de los objetos que participan, permitiendo una mejor comprensión de la funcionalidades internas del sistema.

2.7.1. DIAGRAMA DE CLASES DEL ANÁLISIS

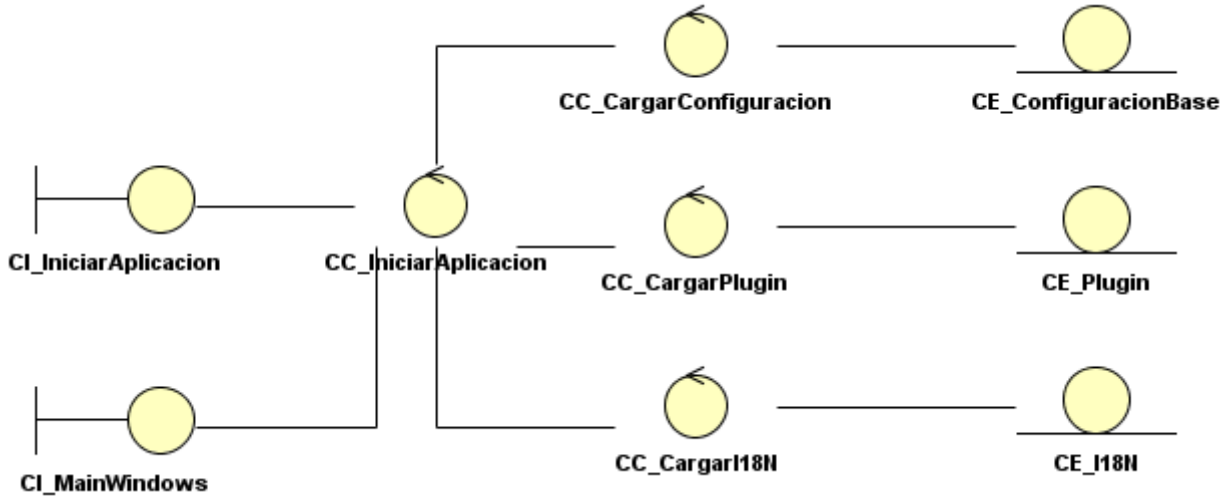


Figura #5 Diagrama de Clase del Análisis del CUS Iniciar Aplicación

Figura #6 Diagrama de Clase del Análisis del CUS Salir de la Aplicación

2.7.2. DIAGRAMA DE COLABORACIÓN

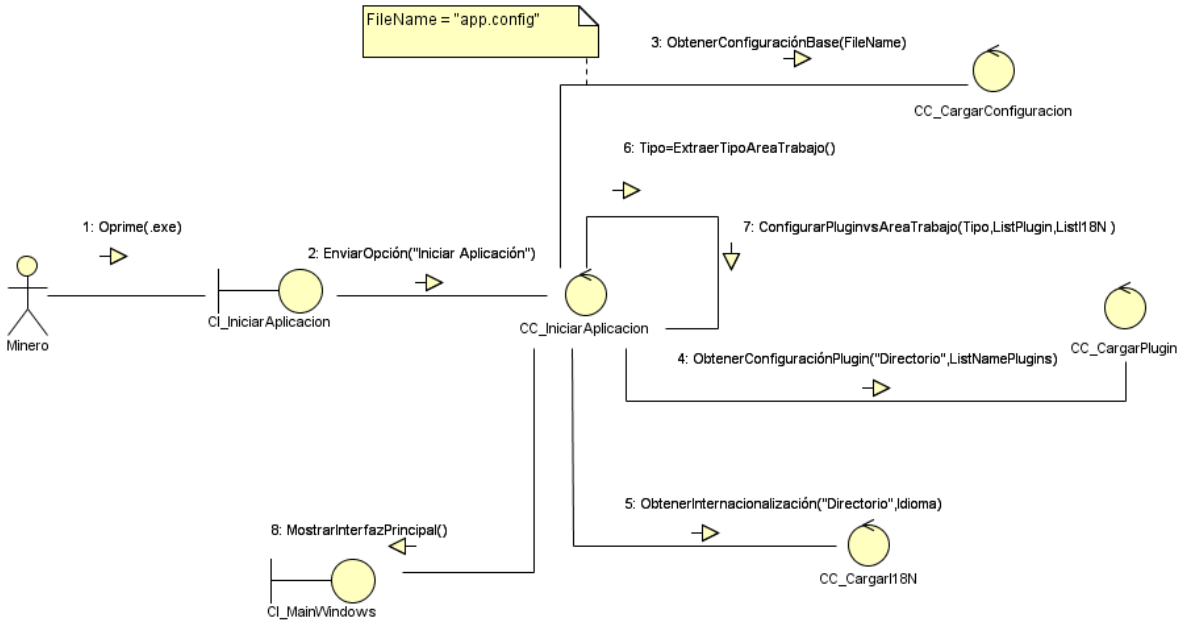


Figura #7 Diagrama de colaboración del CUS Iniciar Aplicación

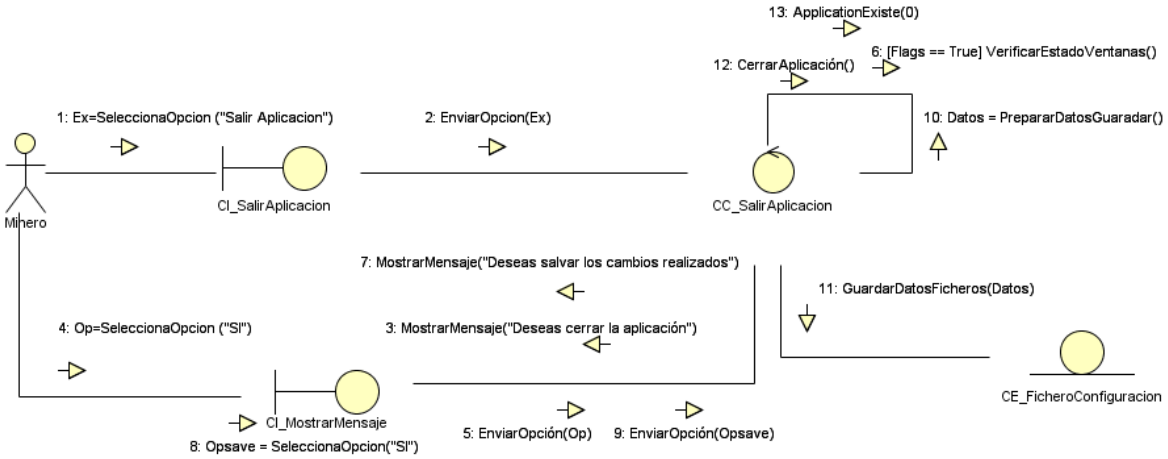


Figura #8 Diagrama de colaboración del CUS Salir Aplicación

2.8. CONCLUSIONES PARCIALES

A partir de la investigación realizada se puede concluir que la modelación del sistema permitió identificar los requisitos funcionales, además de la matriz de trazabilidad que ofrece un seguimiento durante las distintas fases del proyecto, tomando siempre en cuenta las necesidades y expectativas del cliente. Se utilizaron un conjunto de patrones que ayudaron a organizar y estructurar el diagrama de casos de uso del sistema, además se elaboraron los diagramas de clases de análisis y de colaboración que proporcionaron una mejor comprensión de la vista interna del sistema, lográndose identificar las clases que intervienen en el análisis (interfaz, controladora y entidad).

CAPÍTULO #3
ESTUDIO DE FACTIBILIDAD Y VALIDACIÓN DEL SISTEMA

INTRODUCCIÓN

El estudio de factibilidad y la validación del sistema presentan una suma importancia ya que se realizan con el objetivo de auxiliar a una organización a lograr sus metas y a su vez de cubrir sus fines con los recursos actuales. El objetivo fundamental de la planificación y el análisis de la factibilidad es establecer planes razonables para desarrollar la Ingeniería de Software y manejar los cambios de los proyectos de Software. Es pertinente realizar un estudio de factibilidad para determinar los costos, beneficios y grado de aceptación que genera el nuevo sistema. En el presente capítulo para llevar a cabo el análisis de factibilidad del sistema se emplea el método Análisis de Puntos de Casos de Uso para la estimación del esfuerzo mediante la asignación de "pesos" a un cierto número de factores que lo afectan, para finalmente, contabilizar el tiempo total estimado para el proyecto a partir de esos factores.

3.1. ESTUDIO DE FACTIBILIDAD DEL SISTEMA

3.1.1. APLICACIÓN DE LA TÉCNICA ANÁLISIS DE PUNTOS DE CASOS DE USO

A continuación, se detallan los pasos a seguir para la aplicación de éste método:

PASO 1. Cálculo de Punto de Casos de Usos sin ajustar

Se calcula a partir de la siguiente ecuación:

UUCP = UAW + UUCW

Donde:

UUCP: Puntos de Casos de Uso sin ajustar.

UAW: Factor de Peso de los Actores sin ajustar.

UUCW: Factor de Peso de los Casos de Uso sin ajustar.

Este valor del **Factor de Peso de los Actores sin ajustar (UAW)** se calcula mediante un análisis de la cantidad de Actores presentes en el sistema y la complejidad de cada uno de ellos. La complejidad de los Actores se establece teniendo en cuenta en primer lugar si se trata de una persona o de otro sistema y en segundo lugar, la forma en la que el actor interactúa con el sistema. Calculándose de la siguiente manera:

Tipo de actor	Descripción	Factor de peso	Actores	Total
Simple	Sistema con sistema a través de interfaz de programación.	1	0	0
Medio	Sistema con sistema mediante protocolo de interfaz basada en texto.	2	0	0
Complejo	Persona que interactúa con el sistema mediante interfaz gráfica.	3	1	3
Total			1	3

Tabla #10 Factor de Peso de los Actores sin ajustar

Por tanto:

UAW = Σ cant actores $*$ peso

UAW = 3

El valor **Factor de Peso de los Casos de Uso sin ajustar (UUCW)** se calcula mediante un análisis de la cantidad de Casos de Uso presentes en el sistema y la complejidad de cada uno de ellos. La complejidad de los Casos de Uso se establece teniendo en cuenta la cantidad de transacciones efectuadas en el mismo, donde una transacción se entiende como una secuencia de actividades atómica, es decir, se efectúa la secuencia de actividades completa, o no se efectúa ninguna de las actividades de la secuencia. Los criterios se muestran a continuación:

Tipo de CU	Descripción	Peso	Cantidad de CU	Total
Simple	El caso de uso tiene de 1 a 3 transacciones.	5	5	25
Medio	El caso de uso tiene de 4 a 7 transacciones.	10	9	90

Complejo	El caso de uso tiene más de 8 transacciones.	15	1	15
Total			15	130

Tabla #11 Factor de peso de los casos de uso sin ajustar.

Por tanto:

UUCW= Σ cant CU ∗ Peso

UUCW=130

Una vez calculado el valor del **Factor de Peso de los Actores sin ajustar** y el **Peso de los Casos de Usos sin ajustar** se puede calcular los **Puntos de Casos de Uso sin ajustar**:

UUCP = UAW + UUCW

UUCP = 3 + 130

UUCP =133

PASO 2. Cálculo de los Puntos de casos de uso ajustados.

Una vez que se tienen los **Puntos de Casos de Uso sin ajustar**, se debe ajustar éste valor mediante la siguiente ecuación:

UCP = UUCP x TCF x EF

Donde:

UCP: Puntos de casos de uso ajustados.

UUCP: Puntos de casos de uso sin ajustar.

TCF: Factor de complejidad técnica.

EF: Factor de ambiente.

El **Factor de Complejidad Técnica (TCF)** se calcula mediante la cuantificación de un conjunto de factores que determinan la complejidad técnica del sistema. Cada factor se cuantifica en un valor desde 0 (aporte irrelevante) hasta 5 (aporte muy relevante).

Factor	Descripción	Peso	Valor asignado	Total
T1	Sistema distribuido	2	0	0
T2	Objetivos de performance o tiempo de respuesta	1	5	5
T3	Eficiencia del usuario final	1	3	3
T4	Procesamiento interno complejo	1	5	5
T5	El código debe ser reutilizable	1	3	3
T6	Facilidad de instalación	0.5	5	2.5
T7	Facilidad de uso	0.5	5	2.5
T8	Portabilidad	2	5	10
T9	Facilidad de cambio	1	4	4
T10	Concurrencia	1	4	4
T11	Incluye objetivos especiales de seguridad	1	0	0
T12	Provee acceso directo a terceras partes	1	0	0
T13	Se requieren facilidades especiales de entrenamiento a usuarios	1	4	4
Total			43	43

Tabla#12 Factor de complejidad técnica.

Significado de los valores:

0: No presente o sin influencia.

1: Influencia o presencia incidental.

2: Influencia o presencia moderada.

3: Influencia o presencia media.

4: Influencia o presencia significativa.

5: Influencia o presencia fuerte.

Comentarios

T1: El sistema no es distribuido.

T2: Los tiempos de respuesta deben ser muy altos, debe ofrecer un buen rendimiento.

T3: El usuario final requiere la adecuada preparación para usar el sistema.

T4: El procesamiento interno es complejo.

T5: El código es reutilizable, ya que puede ser utilizado en desarrollos similares.

T6: Aunque son varios los módulos a instalar, el trabajo de instalación es muy sencillo.

T7: El sistema cuenta con una interfaz gráfica y amigable, muy sencilla de utilizar.

T8: El sistema debe ser multiplataforma, siendo usado bajo los Sistemas Operativos Windows y Linux.

T9: Se le podrá realizar algún cambio específico, no todos los cambios de forma general.

T10: Algunos procesos a los que accede el usuario dependen de que esté funcionando otro.

T11: No se hará uso de cuentas de usuario.

T12: No provee acceso a terceras partes.

T13: No es tan necesario dale preparación a los usuarios para poder usar el sistema.

Por tanto:

$TCF = 0.6 + 0.01 * \Sigma (peso * valor\ asignado)$

$TCF = 0.6 + 0.01 * 43$

$TCF = 0.6 + 0.43$

TCF = 1.03

El **Factor de Ambiente (EF)** está relacionado con las habilidades y entrenamiento del grupo de desarrollo que realiza el sistema. Cada factor se cuantifica con un valor desde 0 (aporte irrelevante) hasta 5 (aporte muy relevante). Es importante tener en cuenta:

» Para los factores E1 al E4, un valor asignado de 0 significa sin experiencia, 3 experiencia media y 5 amplia experiencia (experto).

» Para el factor E5, 0 significa sin motivación para el proyecto, 3 motivación media y 5 alta motivación.

» Para el factor E6, 0 significa requisitos extremadamente inestables, 3 estabilidad media y 5 requisitos estables sin posibilidad de cambios.

» Para el factor E7, 0 significa que no hay personal *part-time* (es decir todos son *full-time*), 3 significa mitad y mitad y 5 significa que todo el personal es *part-time* (nadie es *full-time*).

» Para el factor E8, 0 significa que el lenguaje de programación es fácil de usar, 3 medio y 5 que el lenguaje es extremadamente difícil.

Factor	Descripción	Peso	Valor asignado	Total
E1	Familiaridad con el modelo de proyecto utilizado	1.5	5	7.5
E2	Experiencia en la aplicación	0.5	3	1.5
E3	Experiencia en orientación a objetos	1	5	5
E4	Capacidad del analista líder	0.5	5	2.5
E5	Motivación	1	5	5
E6	Estabilidad de los requisitos	2	3	6
E7	Personal part-time	-1	3	-3
E8	Dificultad del lenguaje de programación	-1	3	-3
Total			47	21.5

Tabla #13 Factor de ambiente.

Comentarios:

E1: El equipo se encuentra familiarizado con el modelo utilizado.

E2: Se tiene experiencia media en el trabajo con aplicaciones similares.

E3: Hay una amplia experiencia con la programación orientada a objetos.

E4: El analista líder es una persona capacitada.

E5: Hay una alta motivación en el equipo de trabajo para la construcción del producto final.

E6: Los requisitos pueden cambiar en el desarrollo del sistema.

E7: Los miembros no trabajan a tiempo completo debido a que parte del grupo de trabajo son estudiantes.

E8: Se programa en QT C++, que tiene un nivel de complejidad media para los desarrolladores.

Por tanto:

EF = 1.4 − 0.03* Σ (peso*valor asignado)

EF = 1.4 – 0,03 * 21.5

EF = 1.4 – 0.645

EF = 0.755

Una vez que se tiene el **Factor de Ambiente** y **el Factor de Complejidad Técnica**, conjuntamente con los **Puntos de Casos de Uso sin ajustar** que se habían calculado en el paso anterior, se puede calcular entonces los **Puntos de casos de uso ajustados**:

UCP = UUCP * TCF * EF

UCP = 133 * 1.03* 0.755

UCP = 103.43

PASO 3. Estimación de esfuerzo a través de los puntos de casos de uso.

El esfuerzo en horas-hombre está dado por:

E = UCP *CF

Para obtener el factor de conversión (CF) se contabilizan cuántos valores de los que afectan el factor ambiente (E1...E6) están por debajo de la media (3) y los que están por arriba de la media para los restantes (E7, E8).

- Si el total es 2 o menos se utiliza el factor de conversión 20 Horas-Hombre / Punto de Casos de uso.
- Si el total es 3 ó 4 se utiliza el factor de conversión 28 Horas-Hombre / Punto de Casos de uso.
- Si el total es mayor o igual que 5 se recomienda efectuar cambios en el proyecto ya que se considera que el riesgo de fracaso del mismo es demasiado alto.

Total EF = Cantidad EF < 3 (entre E1 –E6) + Cantidad EF > 3 (entre E7, E8) Total EF = 2 + 0 Total EF = 2

En este caso se puede decir que:

CF = 20 Horas-Hombre / Punto de Casos de uso

Por tanto:

E = UCP *CF

E = 103.43* 20

E = 2069 Horas-Hombre

Paso 4. Calcular esfuerzo de todo el proyecto.

Para una estimación más completa de la duración total del proyecto, hay que añadir a la estimación del esfuerzo obtenida por los Puntos de Casos de Uso, las estimaciones de esfuerzo de las demás actividades relacionadas con el desarrollo de software. Para ello se puede tener en cuenta el siguiente criterio, que estadísticamente se considera aceptable. El criterio plantea la distribución del esfuerzo entre las diferentes actividades de un proyecto, según la siguiente aproximación:

Actividad	Porcentaje	Horas- Hombres
Análisis	10 %	206.9
Diseño	20 %	413.8
Programación	40 %	827.6
Pruebas	15 %	310.35
Sobrecarga (otras actividades)	15 %	310.35
Total	100 %	2069

Tabla #14 Esfuerzo del proyecto.

Suponiendo que una persona trabaje 4 horas al día, y un mes tiene como promedio 30 días y que los domingos sean días no laborables; en 1 mes se trabajarán 26 días que equivalen a una cantidad de 104 horas.

En la Tabla #154 se hace una distribución de las diferentes actividades realizadas en el proyecto con las horas de duración que posee cada una. En esta investigación teniendo en cuenta que solo se llega hasta el análisis y además se le adiciona las actividades de sobrecarga donde hay un total de 517.25 horas-hombre, se concluyó que la duración de las actividades realizadas por el rol de analista expuestas anteriormente tienen una duración de 5 meses.

3.2. TÉCNICAS DE VALIDACIÓN DE REQUISITOS

3.2.1. TÉCNICAS DE VALIDACIÓN

Los requisitos una vez definidos necesitan ser validados. Este es un proceso de vital importancia que se centra en demostrar que la definición de los requisitos del sistema son realmente las necesidades del cliente. Existen diferentes Técnicas de validación para los requisitos entre los que se encuentran:(Escalona, María José ; Koch, Nora;, 2002)

» **Listas de chequeo**: Esta técnica es muy fácil de utilizar. En general es una lista de preguntas que el analista debe usar para evaluar cada requerimiento, verificando y marcando los puntos de la

lista, mientras se lee el documento de requisitos. Cuando se descubren problemas potenciales deben ser anotados, ya sea en los márgenes del documento o en una lista de análisis. Las listas brindan un recordatorio de lo que se debe buscar y reducen la posibilidad de obviar alguna verificación importante.

» **Generación de Casos de Prueba:** El propósito en esta técnica es comprobar que el producto cumple todos los requisitos que pueden ser verificados antes de continuar con el desarrollo de las disciplinas subsecuentes del proceso de desarrollo del software. Basada en la premisa de que cada requerimiento debe ser posible de probar, la técnica propone el diseño de Casos de Prueba donde se evalúen todos los requisitos. Un Caso de Prueba para requisitos funcionales generalmente consiste en definir las entradas y las salidas del sistema, además de acciones del usuario que posibilitan completar lo que expresa el requerimiento.

» **Comentarios o tutoriales (*Reviews o walk-throughs*):** Consiste en la lectura y corrección de la completa documentación o modelado de la definición de requisitos. Con ello solamente se puede validar la correcta interpretación de la información transmitida.

» **Auditorías:** La revisión de la documentación con esta técnica consiste en un chequeo de los resultados contra una lista de chequeos predefinida o definida a comienzos del proceso, es decir que sólo una muestra es revisada.

» **Matrices de Trazabilidad:** Consiste en marcar los objetivos del sistema y chequearlos contra los requisitos del mismo. Es necesario ir viendo qué objetivos cubre cada requisito, de esta forma se podrán detectar inconsistencias u objetivos no cubiertos.

» **Prototipos:** Algunas propuestas se basan en obtener de la definición de requisitos prototipos que, sin tener la totalidad de la funcionalidad del sistema, permitan al usuario hacerse una idea de la estructura de la interfaz del sistema con el usuario. Esta técnica tiene el problema de que el usuario debe entender que lo que está viendo es un prototipo y no el sistema final. Sin embargo, posibilita una mayor efectividad en la comunicación de los diseñadores y reduce la necesidad y el costo de rehacer un sistema ya implementado cuando los problemas se identifican en etapas avanzadas del desarrollo. La construcción de prototipos es conveniente cuando se pretende

involucrar a usuarios en el proceso de desarrollo. En estos casos las especificaciones técnicas y los modelos abstractos no suelen ser una buena vía de comunicación.

3.2.2. APLICACIÓN DE LA TÉCNICA SELECCIONADA

En el presente trabajo se utilizan como técnicas de validación la Matriz de Trazabilidad y los Prototipos. Con la aplicación de la técnica Matriz de trazabilidad se pudo apreciar que no existieron inconsistencias en el proceso de incluir los requisitos en los Casos de Uso. Esta matriz describe los requisitos funcionales del sistema de forma vertical y los Casos de Uso del sistema de forma horizontal, de manera que puedan hacerse corresponder entre ellos. Después de ser aplicada se puede apreciar como resultado que todos los requisitos están asociados a un Caso de Uso para cubrir una determinada funcionalidad del sistema. (Ver Tabla #9).

La técnica de Prototipos fue de gran ayuda para que los clientes y/o usuarios pudieran expresar sus necesidades lo que ayudó a los analistas a cubrir esas necesidades de forma precisa. Lo antes mencionado permitió verificar que los requisitos identificados se correspondieran con las demandas reales. Estos prototipos fueron una vía efectiva debido a que posibilitaron comunicar, discutir y definir ideas entre el cliente y los desarrolladores por lo que ayudaron a clarificar requisitos de usuarios y definir alternativas. Los prototipos generados proporcionaron una primera visión del producto, además posibilitaron comprender mejor el problema y su solución además de revelar errores u omisiones en los requisitos propuestos.

3.3. CONCLUSIONES PARCIALES

En este capítulo se realizó un análisis de la factibilidad del sistema donde se estimó el esfuerzo que requiere el mismo. Teniendo en cuenta que en el desarrollo del módulo núcleo, centrándose en las actividades del análisis donde labora 1 persona, se estimó un tiempo de duración de 5 meses aproximadamente. Para lograr una mejor calidad en los artefactos construidos se aplicaron diferentes técnicas de Ingeniería de requisitos que permitieron obtener, definir y validar las necesidades expresadas por el cliente; las cuales fueron definidas mediante requisitos funcionales. La Matriz de trazabilidad y los prototipos ayudaron a detectar y corregir los errores cometidos en la elaboración de los artefactos.

CONCLUSIONES

Al finalizar el presente trabajo de diploma se arribó a las siguientes conclusiones: el estudio de los fundamentos teóricos de los conceptos asociados al dominio y el objeto de estudio ayudaron a un mayor entendimiento del problema. Se realizó un estudio comparativo de las tecnologías a utilizar que permitió obtener los artefactos necesarios para la modelación de la solución informática, contribuyendo a la clarificación y comprensión de los modelos elaborados. La interacción con los clientes del sistema propició que se obtuvieran resultados satisfactorios en la identificación de las condiciones y capacidades que debe cumplir el módulo núcleo del proyecto Minería. El análisis de los resultados obtenidos, es decir, de la especificación de requisitos y del modelo de sistema, permitió medir el grado de factibilidad de estos en función de la calidad, funcionalidad, ambigüedad y consistencia, lo cual demostró que cuentan con la calidad requerida. El análisis de factibilidad permitió predecir el éxito o fracaso de un producto de software. Se realizó la validación de los resultados obtenidos, mediante la utilización de métricas como la matriz de trazabilidad y los prototipos que mejoraron la calidad de los artefactos generados.

RECOMENDACIONES

Al finalizar este trabajo quedan algunas recomendaciones que pueden servir de punto de partida para mejorar aún más el análisis obtenido:

- Realizar el diseño e implementación del módulo Núcleo a partir del modelado de sistema presentado.
- Continuar perfeccionando el modelado de sistema mediante la actualización de los cambios que sean necesarios durante las etapas de diseño, implementación y pruebas.
- Continuar con las investigaciones para añadir nuevas funcionalidades al sistema y obtener mejoras en futuras versiones, logrando adecuarlo cada vez más a las necesidades de los usuarios.

BIBLIOGRAFÍA CITADA

ABC. 2008. Definición ABC. [En línea] 27 de octubre de 2008. [Citado el: 4 de noviembre de 2010.] http://www.definicionabc.com/general/mineral.php.

Academic. 2000-2010. Academic. [En línea] 2000-2010. [Citado el: 5 de diciembre de 2010.] http://www.esacademic.com/dic.nsf/eswiki/973053.

Animación. 2000. Animación. [En línea] 2000. [Citado el: 1 de diciembre de 2010.] http://personal.redestb.es/amaro/animacion.htm.

Argudo Vásconez, Joaquín Andrés y Astudillo Quituisaca, William Hermel. 2010. METODOLOGÍA DE DESARROLLO APLICADA AL SOFTWARE LIBRE. 2010, CAPITULO VI.

babylon. 1997. babylon. [En línea] 1997. [Citado el: 29 de noviembre de 2010.] http://www.babylon.com/definition/n%C3%BAcleo_%28inform%C3%A1tica%29/.

Chilena, Minería. 2010.*Minería Chilena.* CHile : GRUPO EDITORIAL EDITEC, 2010.

código, Planeta. 2008. Planeta código. [En línea] 18 de junio de 2008. [Citado el: 15 de febrero de 2011.] http://www.planetacodigo.com/wiki/glosario:matriz_de_trazabilidad.

Edgardo Tejada, Carlos. noviembre-diciembre 2001.*Ideas Orientadoras sobre el sistema logístico del componente ejército del teatro de operaciones.* s.l. : Military Review, noviembre-diciembre 2001.

Escalona, María José ; Koch, Nora;. 2002. e-amanecer. [En línea] diciembre de 2002. [Citado el: 9 de marzo de 2011.] http://e-amanecer.com/licenciatura/docu/LSI-2002-4-1.pdf.

Estudio de Técnicas basadas en Puntos de Función para la Estimación. Portillo, Jose Antonio Pow-Sang. 2005. Perú : Pontificia Universidad Cato lica del Peru, Seccio n Ing. Informatica, 2005.

Eva. 2005. Entorno virtual de aprendizaje. *Eva.* [En línea] Ingeniería de software, 5 de junio de 2005. [Citado el: 5 de diciembre de 2010.] http://eva.uci.cu/mod/resource/view.php?inpopup=true&id=22434.

Flores, Mirian Milagros Díaz. 2011. USMP. [En línea] 3 de 5 de 2011. [Citado el: 3 de 5 de 2011.] http://www.usmp.edu.pe/publicaciones/boletin/fia/info49/articulos/RUP%20vs.%20XP.pdf.

Hooping. 2009. Hooping.net blog. [En línea] 20 de octubre de 2009. [Citado el: 1 de 12 de 2010.] http://blog.hooping.net/?tag=internet.

IBM. 2010. IBM. [En línea] 2010. [Citado el: 19 de febrero de 2010.] http://www-142.ibm.com/software/products/es/es/reqpro/.

IH. 2007. Informática Hoy, Información para estar actualizado y disfrutar de la informática y la tecnología. [En línea] 2007. [Citado el: 3 de 12 de 2010.] http://www.informatica-hoy.com.ar/aprender-informatica/Que-es-un-sistema-informatico.php.

Larman, Craig. 2000. UML Y Patrones Una introducción al análisis y diseño orientado a objetos y al proceso unificado. 2da Edición. 2000, pág. 18.

mineria, Informática aplicada a la. 2000. Informática aplicada a la mineria. [En línea] Ayuda del DATAMINE , 2000. [Citado el: 5 de diciembre de 2010.] http://informaticaminera.blogspot.com/.

Mis respuestas. 2005. Mis respuestas.com. [En línea] 2005. [Citado el: 4 de noviembre de 2010.] http://www.misrespuestas.com/que-es-la-geologia.html.

Ponjuán Dante, G. 2004.*Aspectos introductorios acerca de los sistemas en: Sistemas de información, Principios y aplicaciones.* La Habana : Félix Varela, 2004.

Pressman, Roger S. 2007.*INGENIEÍA DEL SOFTWARE Un enfoque práctico.* s.l. : 5ta edición, 2007.

Pressman, Roger S. 2007. INGENIEÍA DEL SOFTWARE Un enfoque práctico. *INGENIEÍA DEL SOFTWARE Un enfoque práctico.* Sexta edición. EEUU : Hill, Mc Graw, 2007, Capítulo 11, pág. 316.

Prof. Juan Carlos Gutiérrez Lázaro. 2008-2009. UML Diagramas de clases y casos de uso. [En línea] 2008-2009. [Citado el: 15 de febrero de 2011.] http://www.fdi.ucm.es/profesor/jcgutierrez/Tema%202/02UML-1.pdf.

RAE. 2001. Real Academia Española. [En línea] 2001. [Citado el: 4 de diciembre de 2010.] http://www.rae.es/rae.html.

Ramírez, MsC. Livia M. Reyes. 2000.*Consideraciones teóricas sobre los sistemas de información, los sistemas de información para la prensa y los sistemas integrados de información.* 2000.

Rumbauhg, , J y Jacobson, I. 2000.*El Lenguaje Unificado de Modelado. Manual de referencia.* Madrid : Pearson Educación, 2000.

Sitio de descargas de software. 2007. Sitio de descargas de software. [En línea] 5 de Marzo de 2007. [Citado el: 1 de 12 de 2010.] http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_(M%C3%8D)_14720_p/.

Sommerville, Ian. 2005. INGENIERÍA DEL SOFTWARE. Séptima edición . Madrid : PEARSON EDUCACIÓN.S.A, 2005, pág. 5.

Soto, prof Lauro. 2003. Definición de sistema. [En línea] 2003. [Citado el: 4 de noviembre de 2010.] http://www.mitecnologico.com/Main/DefinicionDeSistema.

SPIP. 2008. SPIP. [En línea] Agosto de 2008. [Citado el: 1 de 12 de 2010.] http://www.spip.net/es_article3460.html.

Szyperski, Clemens. 1998.*"Component Software Component Software Beyond Object – Oriented Programming".* 1998.

Tierra, Asociación Nacional de Maestros de Ciencias de la. 2000. Ventanas al universo. [En línea] 2000. [Citado el: 3 de Diciembre de 2010.] http://www.windows2universe.org/earth/geology/min_intro.html&lang=sp.

Villajos Carmona, Santiago, y otros. 2004. Mineralogía. [En línea] 2004. [Citado el: 4 de noviembre de 2010.] http://www.educared.net/concurso/276/mineralog%C3%ADa.html#sectA.

BIBLIOGRAFÍA CONSULTADA

ABC. 2008. Definición ABC. [En línea] 27 de octubre de 2008. [Citado el: 4 de noviembre de 2010.] http://www.definicionabc.com/general/mineral.php.

Academic. 2000-2010. Academic. [En línea] 2000-2010. [Citado el: 5 de diciembre de 2010.] http://www.esacademic.com/dic.nsf/eswiki/973053.

Agüera, Ivan Obeso. 2000. Estudio de herramientas de análisis y gestión de requisitos. 2000.

Animación. 2000. Animación. [En línea] 2000. [Citado el: 1 de diciembre de 2010.] http://personal.redestb.es/amaro/animacion.htm.

Argudo Vásconez, Joaquín Andrés y Astudillo Quituisaca, William Hermel. 2010. METODOLOGÍA DE DESARROLLO APLICADA AL SOFTWARE LIBRE. 2010, CAPITULO VI.

Asociación nacional de maestros de ciencias de la tierra. 2008. ¿Qué es un Mineral? Ventana al universo. [En línea] 2008. [Citado el: 3 de diciembre de 2010.] http://www.windows2universe.org/earth/geology/min_intro.html&lang=sp.

babylon. 1997. babylon. [En línea] 1997. [Citado el: 29 de noviembre de 2010.] http://www.babylon.com/definition/n%C3%BAcleo_%28inform%C3%A1tica%29/.

bnamericas. 1996. Mineria. [En línea] 1996. [Citado el: 7 de diciembre de 2010.] http://www.bnamericas.com/company-profile/mineria/Mintec_Inc,-Mintec.

Cecilia. 2008. blog ipcorp. [En línea] 11 de diciembre de 2008. [Citado el: 3 de diciembre de 2010.] http://www.ipcorp.com.ar/blog/2008/12/11/osrmt-open-source-requirements-management-tool/.

Chilena, Minería. 2010.*Minería Chilena.* CHile : GRUPO EDITORIAL EDITEC, 2010.

código, Planeta. 2008. Planeta código. [En línea] 18 de junio de 2008. [Citado el: 15 de febrero de 2011.] http://www.planetacodigo.com/wiki/glosario:matriz_de_trazabilidad.

Definición. 2000. definición.org. [En línea] 2000. [Citado el: 4 de diciembre de de 2010.] http://www.definicion.org/sistema.

Definicion.de. 2001. Definición de informática - Qué es, Significado y Concepto.Definición. [En línea] 2001. [Citado el: 3 de diciembre de 2010.] http://definicion.de/informatica/.

Edgardo Tejada, Carlos. noviembre-diciembre 2001.*Ideas Orientadoras sobre el sistema logístico del componente ejército del teatro de operaciones.* s.l. : Military Review, noviembre-diciembre 2001.

Escalona, María José ; Koch, Nora;. 2002. e-amanecer. [En línea] diciembre de 2002. [Citado el: 9 de marzo de 2011.] http://e-amanecer.com/licenciatura/docu/LSI-2002-4-1.pdf.

Estudio de Técnicas basadas en Puntos de Función para la Estimación. Portillo, Jose Antonio Pow-Sang. 2005. Perú : Pontificia Universidad Cato lica del Peru, Seccio n Ing. Informatica, 2005.

Eva. 2005. Entorno virtual de aprendizaje. *Eva.* [En línea] Ingeniería de software, 5 de junio de 2005. [Citado el: 5 de diciembre de 2010.] http://eva.uci.cu/mod/resource/view.php?inpopup=true&id=22434.

Flores, Mirian Milagros Díaz. 2011. USMP. [En línea] 3 de 5 de 2011. [Citado el: 3 de 5 de 2011.] http://www.usmp.edu.pe/publicaciones/boletin/fia/info49/articulos/RUP%20vs.%20XP.pdf.

Geovirtual. 2000. Geología General. [En línea] 2000. [Citado el: 3 de diciembre de 2010.] http://www.geovirtual.cl/geologiageneral/ggcap02e.html.

Griem, Dr. Wolfgang y Griem, Susanne Klee. 1999.Los minerales más importantes. Los Feldespatos. [En línea] 1999. [Citado el: 3 de diciembre de 2010.] http://www.geovirtual.cl/geologiageneral/ggcap02e.html.

Hernández, Enrique Orallo. 2002. El Lenguaje Unificado de Modelado (UML). 2002.

Hill, Mc Graw. 1999.*INGENIERÍA DEL SOFTWARE Un enfoque práctico. .* EEUU : Sexta edición, Capítulo 1, pág. 5., 1999.

Hooping. 2009. Hooping.net blog. [En línea] 20 de octubre de 2009. [Citado el: 1 de 12 de 2010.] http://blog.hooping.net/?tag=internet.

Hooping. 1995. Hooping.net Negocios web. [En línea] 1995. [Citado el: 1 de diciembre de 2010.] http://www.hooping.net/glossary/plugin-94.aspx.

IBM. 2010. IBM. [En línea] 2010. [Citado el: 19 de febrero de 2010.] http://www-142.ibm.com/software/products/es/es/reqpro/.

IH. 2007. Informática Hoy, Información para estar actualizado y disfrutar de la informática y la tecnología. [En línea] 2007. [Citado el: 3 de 12 de 2010.] http://www.informatica-hoy.com.ar/aprender-informatica/Que-es-un-sistema-informatico.php.

Larman, Craig. 2000. UML Y Patrones Una introducción al análisis y diseño orientado a objetos y al proceso unificado. 2da Edición. 2000, pág. 18.

Masadelante. 2002. Definición de software y hardware -¿Qué es software y qué es hardware? [En línea] 2002. [Citado el: 4 de diciembre de 2010.] http://www.masadelante.com/faqs/software-hardware..

Mastermagazine. 2005. Mastermagazine.com. [En línea] 10 de febrero de 2005. [Citado el: 4 de diciembre de 2010.] http://www.mastermagazine.info/termino/4182.php.

mineria, Informática aplicada a la. 2000. Informática aplicada a la mineria. [En línea] Ayuda del DATAMINE , 2000. [Citado el: 5 de diciembre de 2010.] http://informaticaminera.blogspot.com/.

mininginnovation. 2006. Mining innovation in Latin America. [En línea] 2006. [Citado el: 6 de diciembre de 2010.] http://www.mininginnovation.cl/empresas/mintec.htm.

Mis respuestas. 2005. Mis respuestas.com. [En línea] 2005. [Citado el: 4 de noviembre de 2010.] http://www.misrespuestas.com/que-es-la-geologia.html.

Obeso, Ivan Agüera. 1999.*Estudio de herramientas de análisis y gestión de requisitos.* 1999.

Ponjuán Dante, G. 2004.*Aspectos introductorios acerca de los sistemas en: Sistemas de información, Principios y aplicaciones.* La Habana : Félix Varela, 2004.

Pressman, Roger S. 2007.*INGENIEÍA DEL SOFTWARE Un enfoque práctico.* s.l. : 5ta edición, 2007.

Pressman, Roger S. 2004. UML Y Patrones Una introducción al análisis y diseño orientado a objetos y al proceso unificado. s.l. : 2da Edición. pág. 18., 2004.

Pressman, Roger S. 2007. INGENIEÍA DEL SOFTWARE Un enfoque práctico. *INGENIEÍA DEL SOFTWARE Un enfoque práctico.* Sexta edición. EEUU : Hill, Mc Graw, 2007, Capítulo 11, pág. 316.

Prof. Juan Carlos Gutiérrez Lázaro. 2008-2009. UML Diagramas de clases y casos de uso. [En línea] 2008-2009. [Citado el: 15 de febrero de 2011.] http://www.fdi.ucm.es/profesor/jcgutierrez/Tema%202/02UML-1.pdf.

RAE. 2001. Real Academia Española. [En línea] 2001. [Citado el: 4 de diciembre de 2010.] http://www.rae.es/rae.html..

Ramírez, MsC. Livia M. Reyes. 2000.*Consideraciones teóricas sobre los sistemas de información, los sistemas de información para la prensa y los sistemas integrados de información.* 2000.

Rumbauhg, , J y Jacobson, I. 2000.*El Lenguaje Unificado de Modelado. Manual de referencia.* Madrid : Pearson Educación, 2000.

Sitio de descargas de software. 2007. Sitio de descargas de software. [En línea] 5 de Marzo de 2007. [Citado el: 1 de 12 de 2010.] http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_(M%C3%8D)_14720_p/.

Sommerville, Ian. 2005. INGENIERÍA DEL SOFTWARE. Séptima edición . Madrid : PEARSON EDUCACIÓN.S.A, 2005, pág. 5.

Soto, prof Lauro. 2003. Definición de sistema. [En línea] 2003. [Citado el: 4 de noviembre de 2010.] http://www.mitecnologico.com/Main/DefinicionDeSistema.

SPIP. 2008. SPIP. [En línea] Agosto de 2008. [Citado el: 1 de 12 de 2010.] http://www.spip.net/es_article3460.html.

Szyperski, Clemens. 1998.*"Component Software Component Software Beyond Object – Oriented Programming".* 1998.

Tierra, Asociación Nacional de Maestros de Ciencias de la. 2000. Ventanas al universo. [En línea] 2000. [Citado el: 3 de Diciembre de 2010.] http://www.windows2universe.org/earth/geology/min_intro.html&lang=sp.

Villajos Carmona, Santiago, y otros. 2004. Mineralogía. [En línea] 2004. [Citado el: 4 de noviembre de 2010.] http://www.educared.net/concurso/276/mineralog%C3%ADa.html#sectA.

Visual Paradigm. 2007. Visual Paradigm.com. [En línea] 2007. [Citado el: 5 de diciembre de 2010.] http://www.visualparadigm.com/product/vpuml/.

wordreference. 2005. Diccionario de la lengua española. [En línea] 2005. [Citado el: 3 de diciembre de 2010.] http://www.wordreference.com/definicion/software.

worldlingo. 2008. Feldespato. [En línea] 2008. [Citado el: 3 de diciembre de 2010.] http://www.worldlingo.com/ma/enwiki/es/Feldspar.