ISMMM
INSTITUTO SUPERIOR MINERO
METALURGICO DE MOA
DR. ANTONIO NUÑEZ JIMENEZ

Ingeniería Informática
Facultad: Geología y Minas

Trabajo de Diploma

Para Optar por el Título de

Ingeniero Informático

Título: Aplicación para el apoyo a la toma de decisiones para la gestión del capital humano.

Autor (es): Taylor Leandro Ma Pérez.

Tutor (es): Ing. Eloy Rafael Jiménez Iglesias.

Dra.C. Yiezenia Rosario Ferrer.

Moa, 2014

"Año 56 de la Revolución""

DECLARACIÓN DE AUTORÍA

Declaro que soy el único autor de este trabajo y autorizo al Departamento de Informática del Instituto Superior Minero Metalúrgico de Moa para que hagan el uso que estimen pertinente con el mismo.

Para que así conste firmo la presente a los ___ días del mes de _____del 2014.

Taylor Leandro Ma Pérez

_____

Firma del Autor

Ing. Eloy Rafael Jiménez iglesias

_____

Firma del Tutor

Dra.C. Yiezenia Rosario Ferrer.

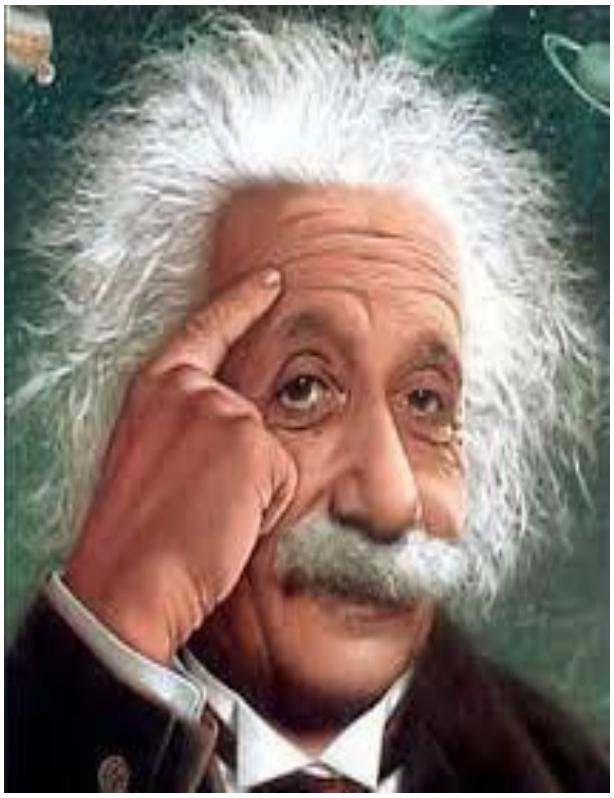
_____

Firma del Tutor

Pensamiento

Agradezco a los que me dijeron que no, porque gracias a ellos pude lograrlo.

Albert Einstein



Si buscas resultados distintos no siempre hagas lo mismo.

Albert Einstein

Dedicatoria

Dedico este a la mujer de mi vida, mi abuela Mama.

A mi mamá que es la mejor madre del mundo porque entre otras cosas siempre confió en mí y ha estado ahí para darme fuerzas y no rendirme.

A mi papá por sus consejos.

A mi papá Luis por su preocupación, interés y confianza en mí.

A mis hermanos Lori y el Negro.

A mi abuela Muñeca y a mi abuelo Coco.

A toda mi familia que siempre han esperado lo mejor de mí y me han inculcado los mejores sentimientos y principios para ser un hombre de bien.

"A todos ustedes dedico este trabajo de diploma".

Agradecimientos

Le agradezco a Dios por estar siempre a mi lado.

A mis padres.

A mis tutores por darme su tiempo, paciencia y ayuda.

A mi novia por su compañía en este difícil camino.

A mi suegra por estar siempre atenta de mis cosas.

A mi familia por estar siempre atentos y preocupados.

A Mainoldis por su apoyo para la realización de este trabajo.

A Adriannys, Yanelis y Roilber por su ayuda incondicional.

A los informáticos de FEMSA, Abelito y Pepe por brindarme su tiempo y recursos.

A mis amigos: Lisandra, Naivys, Kenia, Ernesto, Walter y Tiquito.

A mi grupo info 2009, compañeros de aula, en especial a: Quintana, Jose, Guerrero, Rafa, Roco, Yuris, Ivan y Jorge.

A mis vecinos por ser tan atentos conmigo.

"A todos ustedes, mil gracias".

Resumen

En la actualidad el proceso de selección de personal se ha convertido en uno de los asuntos de mayor importancia para todas las instituciones; incitando la búsqueda de la perfección del mismo. Moa, municipio minero, donde la mayor parte de la población está vinculada a la Industria del Níquel y sabiendo que este recurso es agotable, no se encuentra exento de un cierre minero, situación en la que quedarían miles de trabajadores desempleados. Esto traería consigo que el proceso de reordenamiento laboral del personal excedente fuera engorroso y lento pues se realizaría manualmente.

La presente investigación se trazó como objetivo elaborar una herramienta informática que permita obtener con mayor rapidez el personal adecuado para ocupar un cargo, favoreciendo así el proceso de selección del capital humano tras un reordenamiento laboral en la industria del Níquel.

Para esto se realizó una entrevista que permitiera conocer el proceso de selección de personal para la Industria del Níquel, así como una revisión bibliográfica en aras de determinar las tecnologías y herramientas más adecuadas para la informatización del proceso, permitiendo elaborar un sistema siguiendo las fases de la metodología de desarrollo de software XP, dándole cumplimiento al objetivo de este trabajo.

El principal logro de la investigación fue proporcionar una vía eficiente y cómoda para la realización del proceso de selección del personal, reduciendo el tiempo de realización del mismo y las deficiencias en cuanto al manejo de la información.

Abstract

At present, the staff selection process has become one of the most important issues for all institutions; this has motivated to look for the improvement of it. Moa, a mining municipality, where most of the population is linked with the Nickel Industry and knowing that this resource is exhausted, it is not exempt to a mining closing, situation in which thousands workers would be unemployed. This carries out that the process of labor reorganization of the surplus staff would be annoying and slow since it would be carried out manually.

In that way, the present research has as objective to elaborate a computer tool that allows obtaining with more speed the appropriate staff to occupy a position, favoring this way the selection process of the human capital after labor reorganization in the Nickel Industry.

To carry out the research an interview was made to know the staff selection process for the Nickel Industry, as well as a bibliographical revision to determine the technologies and more appropriate tools for the informatics process. It permitted to elaborate a system following the phases of the software development XP methodology, giving to achieve the proposed objective of this work.

The main result of the present research was an efficient and easy way for the realization of the staff selection process, reducing the realization time of this process and the deficiencies in handling of the staff information was provided.

Contenido

Índice de tablas e ilustraciones

Introducción

La toma de decisiones es un proceso común en las organizaciones, durante el cual se elige entre varias alternativas para dar solución a diferentes tipos de problemas. Este proceso es muy amplio e involucra, en la mayoría de los casos, varias áreas de conocimiento.

Al tomar una decisión es importante tener en cuenta la información del tema en cuestión. Las organizaciones generan grandes cantidades de información, pero el problema radica en tenerla organizada, resumida, que sea útil y fácil de interpretar.

El proceso de toma de decisiones está regido generalmente por la experiencia de los directivos y encargados involucrados en la planificación de dichas acciones, y la cantidad de factores subjetivos, a veces imprecisos (propio del comportamiento humano), que pueden convertirse en obstáculos ante la realización de una estrategia correcta.

El volumen y variedad de la información almacenada en bases de datos y otras fuentes, ha tenido un aumento notable en las últimas décadas, constituyendo gran parte de ella una reseña de sucesos ocurridas.

Asimismo, la influencia de la toma de decisiones como apoyo a la selección del capital humano es muy importante. Pues una decisión mal tomada al emplear a una persona podría llevar consigo ineficiencia y pérdida en la entidad. Pues no sería la más adecuada al desempeñar el cargo ocupado, al no tener la experiencia, el conocimiento y requisitos necesarios para llevar a cabo dicha labor.

La capacidad de mantener y desarrollar profesionales calificados se ha convertido en un diferenciador competitivo para las empresas de todos los sectores, especialmente las que operan en los mercados más dinámicos. La escasez de mano de obra calificada y los costos crecientes exigen una gestión más eficaz de este activo. En un mundo globalizado, el factor limitante para el crecimiento de muchas empresas no es el capital financiero, sino el capital humano.

Con la implementación de los lineamientos de la política económica y social del Partido y la Revolución se comienza el análisis del funcionamiento y estructura de instituciones y empresas a lo largo del país y de todos los sectores además de la búsqueda de nuevas fórmulas que garanticen eficiencia en el sistema económico cubano.

El municipio minero de Moa, no ha quedado exento a los cambios y restructuración en entidades donde ocurrían pérdidas en la economía del país. Se hace necesaria la reubicación según necesidades existentes y categoría científica y laboral para los trabajadores que quedaron disponibles o fuera de sus empleos.

El territorio no cuenta con una herramienta para manipular toda la información que permita la selección adecuada del talento humano para diferentes cargos de trabajo, en caso de un cierre de minas, tornándose extensa y poco práctica para ciertos tipos de consultas. Por lo que surge la necesidad de desarrollar un sistema capaz de procesar esta información, como ayuda a la toma de decisiones de los directivos responsables ante una reubicación laboral masiva.

La **situación problémica** existente se basa en la insuficiente gestión del capital humano de las empresas moenses ante un reordenamiento laboral tras un cierre de minas. Este proceso es lento porque es manual y por tanto es mucho el volumen de información por trabajadores para analizar. Se debe tener en cuenta que no se trata de un proceso sencillo, sino de una situación de cierre donde quedan miles de obreros disponibles. El análisis para determinar si una persona está capacitada para desempeñar un determinado cargo laboral se realiza de forma manual y con deficiencias, haciendo el proceso engorroso y confuso, al optar por una misma plaza, muchas personas.

De ahí que el presente trabajo de diploma para brindar solución a la situación descrita con anterioridad, posee el siguiente **problema científico:** ¿Cómo favorecer la toma de decisiones para el reordenamiento laboral tras un cierre minero en la comunidad de Moa?

En busca de una solución al problema antes planteado se propone como **objeto de estudio** la informatización de la toma de decisiones para el reordenamiento laboral tras un cierre minero en las empresas moenses.

El **campo de acción** de este trabajo de diploma lo conforman: los sistemas informáticos de gestión de información para la toma de decisiones.

La presente investigación tiene como **objetivo general:** desarrollar una aplicación informática como apoyo para la toma de decisiones ante un reordenamiento laboral después de un cierre de minas.

Los **objetivos específicos** de la investigación son:

- Elaborar el Marco Teórico de la investigación.
- Seleccionar la metodología, tecnologías y herramientas a utilizar para el desarrollo del sistema.
- Realizar el análisis y diseño de la aplicación a desarrollar que posibilite mostrar los principales parámetros a tener en cuenta para llevar a cabo la fase de desarrollo del sistema.
- Determinar la factibilidad de desarrollo del proyecto.

Como guía en esta investigación se plantea la siguiente **idea a defender:** si se desarrolla la aplicación informática para apoyar a la toma de decisiones, el proceso de reordenamiento del personal disponible tras un cierre minero será más eficiente, rápido y fiable.

Para cumplir los objetivos y resolver la situación problémica presentada, se ejecutaron las siguientes **tareas:**

- Realizar entrevistas.
- Búsqueda bibliográfica.
- Estudiar los antecedentes de la investigación.
- Investigar el proceso de selección de personal en la Industria del Níquel.
- Estudiar la metodología de desarrollo de software a utilizar.
- Estudiar las herramientas para la construcción del software.
- Realizar diseño e implementación de la herramienta.

- Conectar el sistema a la ontología de perfiles de cargos.
- Realizar pruebas al software para garantizar la calidad del mismo.
- Demostrar la factibilidad de la aplicación.

Para responder a las tareas propuestas anteriormente se utilizaron los siguientes métodos científicos en la investigación:

Métodos empíricos:

- Entrevista: necesaria en la recopilación de la información para el conocimiento del problema en general. En esta investigación se realizaron varias entrevistas con los expertos, con el fin de obtener información y requisitos necesarios para llevar a cabo el proyecto.
- Comparación: se utilizó en la búsqueda y solución de problemas, donde pudimos comparar todas las herramientas estudiadas y así definir cuál utilizar.
- Revisión de documentos utilizados para la recopilación de información: en el estudio de diferentes bibliografías para la selección de metodologías y herramientas, aportando elementos para la fundamentación de la solución.
- La observación: se empleó para percibir cómo se gestiona la información en las Empresas.

Métodos teóricos:

- Análisis y síntesis: empleado en la recopilación y el procesamiento de la información obtenida en los métodos empíricos y de esta forma arribar a las conclusiones.
- Método de Modelación: empleado en la construcción de modelos como el modelo físico de la Base de Datos y el modelo lógico de la misma.
- Inducción–deducción: empleado para la aplicación de la metodología de desarrollo y la interpretación de los resultados

El presenta trabajo se **estructura** en cuatro capítulos fundamentales:

Capítulo 1 Fundamentación del Marco Teórico: Se expone el estado del arte, donde se realiza la fundamentación teórica del tema. Al mismo tiempo se describe el objeto de estudio. Se explican conceptos y procesos para una mejor

comprensión de la investigación; se describe la metodología a seguir para la construcción de la aplicación; se realiza el estudio y selección de las herramientas, así como de los artefactos para su elaboración.

Capítulo 2 Planificación y Diseño: Se detallan las necesidades del cliente, se describen las funcionalidades que serán objeto de automatización mediante el empleo de las historias de usuarios (HU), se establece un plan de iteraciones necesarias sobre el sistema para su terminación. Además se presentan las tarjetas Clases, Responsabilidades y Colaboradores (CRC), que permitirán trabajar con una metodología basada en objetos.

Capítulo 3 Implementación y Pruebas: Se presenta el modelo de datos empleado para la aplicación concluyente, y se realiza el desarrollo de las iteraciones a partir del desglose de las historias de usuario en tareas de programación. Asimismo aparece una interfaz gráfica de usuario diseñadas para la aplicación final. Se describen además las pruebas realizadas y se indican las respuestas de la aplicación en el empleo de las diferentes funcionalidades.

Capítulo 4 Estudio de factibilidad: Se tienen en cuenta los costos a incurrir, deduciéndose si el proyecto realizado será factible o no llevarlo a cabo. Hay muchas formas de calcular el costo, pero para nuestro caso se utilizará la Metodología Costo Efectividad, la cual sugiere que la conveniencia de la ejecución de un proyecto se determina por la observación de ciertos factores en conjunto.

Capítulo 1: Fundamentación del marco teórico

Introducción

En este capítulo se hace referencia a un grupo de conceptos, de los cuales es muy importante tener dominio y conocimientos para una futura comprensión de términos que serán tratados en el desarrollo de este documento. También se verán los antecedentes con respecto a este tipo de trabajo, se realiza un estudio de la metodología, lenguajes y herramientas seleccionada para el desarrollo del trabajo.

1.1 Proceso de selección del Capital Humano.

El proceso de selección del capital humano en la industria del Níquel en Moa comienza cuando llega la solicitud de personal de una empresa a la EMPLENI (Empleadora del Níquel). Estando allí la solicitud oficial de una empresa el empleado responsable abre el software, que contiene su base de datos en Access, para hacer la búsqueda del personal por el criterio de cargo. Después de revisar el resultado mostrado por el software que sería una lista de candidatos a ser empleados, el especialista evalúa y selecciona los que considere que cumplen con los requisitos del cargo. Luego se convoca a una reunión para escoger de los candidatos seleccionados los más completos para ejercer en la plaza vacante. Elegidos los candidatos, se les realiza un test psicométrico para evaluar su coeficiente. En caso de ser aprobado, ya estarían listos para ejercer dicho trabajo.

1.2 Software libre

En el mundo ha habido un notorio crecimiento en el uso de los software libres hasta superar en ocasiones al mercado propietario debido a las ventajas especialmente económicas que brindan los mismos.

1.2.1 Ventajas del software libre

- Bajo costo de adquisición: Se trata de un software económico ya que permite un ahorro de grandes cantidades en la adquisición de las licencias.
- Innovación tecnológica: esto se debe a que cada usuario puede aportar sus conocimientos y su experiencia y así decidir de manera conjunta hacia donde se debe dirigir la evolución y el desarrollo del software. Este es un gran avance en la tecnología mundial.
- Independencia del proveedor: al disponer del código fuente, se garantiza una independencia del proveedor que hace que cada empresa o particular pueda seguir contribuyendo con el desarrollo y los servicios del software.
- Escrutinio público: esto hace que la corrección de errores y la mejora del producto se lleven a cabo de manera rápida y eficaz por cada uno de los usuarios que lleguen a utilizar el producto.
- Adaptación del software: esta cualidad resulta de gran utilidad para empresas e industrias específicas que necesitan un software personalizado para realizar un trabajo específico y con el software libre se puede realizar y con costes totales de operación (TCO) mucho más razonables.
- Lenguas: aunque el software se cree y salga al mercado en una sola lengua, el hecho de ser software libre facilita en gran medida su traducción y localización para que usuarios de diferentes partes del mundo puedan aprovechar estos beneficios.

Cuba por las condiciones del bloqueo económico tiene problemas para la comercialización de los sistemas informáticos, producto de las licencias. El software libre tiene licencias BSD (usada por PostgreSQL), GPL, AGPL, MPL y derivadas, las cuales nos proporcionan la posibilidad de poder comercializar el software, por tanto como parte de este proceso se decide para el desarrollo de la aplicación, la utilización de herramientas y tecnologías pertenecientes al software libre. Estas ventajas hacen que nuestro país siga una política de

migración hacia el software libre y también se pretende apoyar esta política con el desarrollo de esta aplicación.

Una vez que un producto de software libre ha empezado a circular, rápidamente está disponible a un costo muy bajo. Al mismo tiempo, su utilidad no decrece. El software, en general, podría ser considerado un bien de uso inagotable, tomando en cuenta que su costo marginal es pequeñísimo y que no es un bien sujeto a rivalidad (la posesión del bien por un agente económico no impide que otro lo posea).

El software libre permite el libre uso, modificación y redistribución y a menudo encuentra un hogar entre usuarios para los cuales el coste del software no libre es a veces prohibitivo, o como alternativa a la piratería. También es sencillo modificarlo localmente, lo que permite que sean posibles los esfuerzos de traducción a idiomas que no son necesariamente rentables para su comercialización.

1.3 Sistema informático

Un sistema informático puede ser definido como un sistema de información que basa la parte fundamental de su procesamiento en el empleo de la computación, como cualquier sistema es un conjunto de funciones interrelacionadas, hardware, software y de Recurso Humano. Un sistema informático normal emplea un sistema que usa dispositivos para programar y almacenar programas y datos. (EcuRed, 2013).

Si además de la información, es capaz de almacenar y difundir los conocimientos que se generan sobre cierta temática, tanto dentro como en el entorno de la entidad, entonces está en presencia de un sistema de gestión de información y conocimientos. Como utilizador final emplea esa información en dos actividades fundamentales: la toma de decisiones y el control.

1.3.1 Sistema informático para la toma de decisiones

Un sistema de información es un conjunto de elementos que interactúan entre sí con el fin de apoyar las actividades de una empresa o negocio. El equipo computacional: hardware necesario para que el sistema de información pueda

operar. El recurso humano que interactúa con el Sistema de Información está formado por las personas que utilizan el sistema; este realiza cuatro actividades básicas: entrada, almacenamiento, procesamiento y salida de información.

Entrada de Información: Es el proceso mediante el cual el Sistema de Información toma los datos que requiere para procesar la información. Las entradas pueden ser manuales o automáticas. Las manuales son aquellas que se proporcionan en forma directa por el usuario, mientras que las automáticas son datos o información que provienen o son tomados de otros sistemas o módulos.

Almacenamiento de información: El almacenamiento es una de las actividades o capacidades más importantes que tiene una computadora, ya que a través de esta propiedad el sistema puede recordar la información guardada en la sección o proceso anterior.

Procesamiento de Información: Es la capacidad del Sistema de Información para efectuar cálculos de acuerdo con una secuencia de operaciones preestablecida. Estos cálculos pueden efectuarse con datos introducidos recientemente en el sistema o bien con datos que están almacenados.

Salida de Información: La salida es la capacidad de un Sistema de Información para sacar la información procesada o bien datos de entrada al exterior. Es importante aclarar que la salida de un Sistema de Información puede constituir la entrada a otro Sistema de Información o módulo.

1.4 Antecedentes

- ARNOM. Sistema Integral de Recursos Humanos diseñado por profesionales en la materia que permite al usuario contar con una herramienta para la administración de su Capital Humano en las áreas de personal, Nómina, Control de Asistencia y Capacitación. (ARNOM, 2014)
- GOSEM. Software de Gestión Humana, es un software para la administración y gestión del recurso humano, desarrollado en tecnología

de punta .Net Web, basado en modelos modernos y especializados en los temas gestión por competencias organizacionales, aplicables para la región. Fue realizado con la asesoría y consultoría de la asociación colombiana de relaciones de trabajo (Ascort). Compuesto por seis módulos; talento, desarrollo humano, evaluaciones de gestión, remuneración, salud ocupacional / seguridad industrial. (GOSEM, 2014)

- SIGEIN. Es un sistema para la gestión integral de capital humano que automatiza y da mayor eficiencia a todas las tareas operativas de los procesos de evaluación y administración del talento, permitiendo que el equipo de desarrollo organizacional dedique mayor tiempo a la estrategia y menos a la operación, cuenta con una suite completa de módulos para la implementación de programas de desarrollo organizacional. (SIGEIN, 2014)
- IARA SAGREH. El sistema está diseñado para funcionar en un ambiente multiempresarial, con este producto se puede controlar toda la información relacionada con los Recursos Humanos, registra datos sobre el personal, estructura organizacional y relaciones laborales, Actualmente el sistema está instalado en 36 entidades, cinco del Grupo Empresarial Cubaníquel, en el Grupo Empresarial QUIMEFA (Unión Químico-Farmacéutica)y en la Unión Eléctrica, y el principal propósito es generalizar su aplicación en todo el Ministerio de Energía y Minas. (SAGREH, 2014)

Los sistemas para la gestión del capital humano mencionados anteriormente no cumplen con los parámetros requeridos para la selección del capital humano en la industria del níquel en Moa. Siendo necesario el desarrollo del sistema informático que conlleva la presente investigación, que cumple con los parámetros requeridos en dicha industria.

1.5 Metodología de Desarrollo de Sistemas Informáticos

Desarrollar un buen software depende de un sin números de actividades y etapas, donde el impacto de elegir la mejor metodología para un equipo en un determinado proyecto, es trascendental, para el éxito del producto.

Las metodologías de desarrollo de software son un conjunto de procedimientos, técnicas y ayudas a la documentación para el desarrollo de productos software. Pueden ser comparadas con un plan de contingencias en el que se va indicando paso a paso todas las actividades a realizar para lograr el producto informático deseado, indicando además quienes deben participar en el desarrollo de las actividades y qué papel deben de tener. Detallan además la información que se debe producir como resultado de una actividad y la información necesaria para comenzarla (Solís Álvarez, 2005).

1.5.1 Metodologías para el Desarrollo de Software analizadas

El desarrollo de software no es una tarea fácil. Prueba de ello es que existen numerosas propuestas metodológicas que inciden en distintas dimensiones del proceso de desarrollo. Por una parte tenemos aquellas propuestas más tradicionales que se centran especialmente en el control del proceso, estableciendo rigurosamente las actividades involucradas, los artefactos que se deben producir, las herramientas y notaciones que se usarán. Estas propuestas han demostrado ser efectivas y necesarias en un gran número de proyectos, pero también han presentado problemas en muchos otros. (FERRER, 2008)

Sin embargo, el resultado final sería un proceso de desarrollo más complejo que puede incluso limitar la propia habilidad del equipo para llevar a cabo el proyecto. Otra aproximación es centrarse en el factor humano o el producto software. A continuación se analizan varias metodologías de desarrollo de software escogiendo la más factible para el desarrollo de la herramienta a construir. (FERRER, 2008)

Proceso Unificado de Desarrollo (RUP)

Es una metodología para proyectos más largos, debido a la gran cantidad de diagramas que lleva consigo, además se documenta poco sobre el Sistema que se está llevando a cabo. (FERRER, 2008)

XP (Programación Extrema)

Metodología que adopta 12 prácticas que se pueden utilizar todas o no, eso lo deciden el programador y el cliente según las necesidades de este último o si la aplicación no requiere de todas. Se centra especialmente en documentar en forma de plantillas, tiene cuatro fases: Planeación, Diseño, Desarrollo o Implementación y Pruebas. En la primera fase se generan como artefactos los usuarios del negocio, las historias de usuarios, la lista de reserva del producto, el plan de iteraciones, entre otros. En la segunda se tiene el modelo de datos, tarjetas CRC. En tercera fase se desarrollaron las tareas de ingeniería y la cuarta fase son efectuadas las pruebas al software para verificar que el mismo cumpla con todos las funcionalidades acordadas, estas pruebas pueden ser aceptadas por el cliente o denegadas por el mismo. (FERRER, 2008)

SXP (Scrum + Programación. Extrema)

Esta Metodología fue elaborada en la Universidad de las Ciencias Informáticas, escogieron una parte de Scrum y otra parte de XP, para una mayor organización divide en carpetas todas sus plantillas. Es un híbrido de metodologías ágiles, que ofrece una estrategia tecnológica, a partir de la introducción de procedimientos que permiten actualizar los procesos de software para el mejoramiento de la actividad productiva fomentando el desarrollo de la creatividad, aumentando el nivel de preocupación y responsabilidad de los miembros del equipo, ayudando al líder del proyecto a tener un mejor control del mismo. Con la utilización de SCRUM para la gestión, se logra una planificación y organización inigualable; mientras que XP respalda con sus prácticas todo el proceso de desarrollo, obteniéndose de esta forma un proceso de software completo.

1.5.2 Metodología propuesta para el desarrollo de la aplicación

¿Por qué elegir XP?

La metodología escogida fue XP porque es el método ágil más documentado y no es necesario adoptarlo en forma completa, sino que pueden utilizarse varias de sus prácticas en forma independiente. Es una de las llamadas

metodologías ágiles de desarrollo de software más exitosas de estos tiempos. Está diseñada para entregar el software los clientes en el momento en que lo necesitan. Además alienta a los desarrolladores a responder a los requerimientos cambiantes de los clientes, aún en fases tardías del ciclo de vida del desarrollo. Uno de sus requerimientos es tener al cliente disponible durante todo el proyecto, formando parte del grupo de desarrollo.

La metodología XP o Extreme Programming es una de las variantes de las metodologías ágiles con más aceptación en la comunidad internacional de desarrollo. Esta metodología consiste en una programación rápida o extrema, cuya particularidad es tener como parte del equipo al usuario final, pues es uno de los requisitos para llegar al éxito del proyecto. XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico. (Universidad de Oriente)

Ventajas

Actualmente XP es la metodología ágil más documentada y extendida. Existe una gran comunidad de desarrolladores XP. Otra de las ventajas de XP es que no es necesario adoptarlo en forma completa, sino que pueden utilizarse varias de sus prácticas en forma independiente. Esto hace que el costo de su implementación sea mucho más accesible que el de otras metodologías. Algunas de las ventajas que tiene XP se exponen en los puntos siguientes:

- Puede ser implementado en forma parcial (elegir sólo algunas de las prácticas)
- Puede ser implementado en forma gradual.
- Puede adaptarse a las necesidades de cualquier equipo de desarrollo. De hecho, Kent Beck recomienda a los equipos que lo adapten a sus necesidades.
- Exige que se establezca una comunicación más fluida con el cliente y que este tenga mayor participación en el proceso de desarrollo. La consecuencia de esto es que el cliente se involucre más en el desarrollo del producto.
- Actualmente es la metodología ágil más extendida y documentada.

¿Qué propone XP?

- Empieza en pequeño y añade funcionalidad con retroalimentación continua.
- El manejo del cambio se convierte en parte sustantiva del proceso.
- El costo del cambio no depende de la fase o etapa.
- No introduce funcionalidades antes de que sean necesarias.
- El cliente o el usuario se convierte en miembro del equipo. (Beck, 1999)

Fases de la metodología XP

Fase I: Planificación

1- Se escriben historias de usuario, cuya idea principal es describir un caso de uso en dos o tres líneas con terminología del cliente (de hecho, se supone que deben ser escritos por él mismo), de tal manera que se creen test de aceptación para historias de usuarios (user store) y permita hacer una estimación de su tiempo de desarrollo. (Beck, 1999)

2- Se crea un plan de lanzamiento (release planning) que debe servir para crear un calendario que todos puedan cumplir y en cuyo desarrollo hayan participado todas las personas involucradas en el proyecto. Se usa como base las historias de usuario, participando el cliente en la elección de las que se desarrollarán, y según las estimaciones de tiempo de los mismos se crearán las iteraciones del proyecto. (Beck, 1999)

3- El desarrollo se divide en iteraciones, cada una de las cuales comienzan con un plan de iteración, para el que se eligen las historias de usuario a desarrollar y las tareas de desarrollo. (Beck, 1999)

4- Se cambia el proceso cuanto sea necesario, para adaptarlo al proyecto. (Beck, 1999)

Fase II: Diseño

1- Se eligen los diseños funcionales más simples.

2- Se elige una metáfora del sistema para que el nombrado de clases, siga una misma línea, facilitando la reutilización y la comprensión del código.

3- Se escriben tarjetas de clase-responsabilidades-colaboración (CRC) para cada objeto, que permitan abstraerse al pensamiento estructurado y que el equipo de desarrollo completo participe en el diseño.

Fase III: Implementación

1- El cliente está siempre disponible, de ser posible, cara a cara. La idea es que forme parte del equipo de desarrollo, y esté presente en todas las fases de XP. La idea es usar el tiempo del cliente para estas tareas en lugar de crear una detallada especificación de requisitos, y evitar la entrega de un producto insuficiente, que conlleve a la pérdida de tiempo.

2- El código se ajustará a unos estándares de codificación, asegurando la consistencia y facilitando la comprensión y refactorización del código.

3- Las pruebas unitarias se codifican antes que el código en sí, haciendo que la codificación de este último sea más rápida, y que cuando se afronte la misma se tenga más claro, qué objetivos tiene que cumplir lo que se va a codificar.

4- La programación del código se realiza en parejas, para aumentar la calidad del mismo. En cada momento, sólo habrá una pareja de programadores que integre el código.

5- Se integra código y se lanza dicha integración de manera frecuente, evitando divergencias en el desarrollo y permitiendo que todos trabajen con la última versión del desarrollo. De esta manera, se evitará pasar grandes períodos de tiempo integrando el código al final del desarrollo, ya que las incompatibilidades serán detectadas enseguida.

6- Se usa la propiedad colectiva del código, lo que se traduce en que cualquier programador puede cambiar la parte del código que desee. El objetivo es fomentar la contribución de ideas por parte de todo el equipo de desarrollo.

7- Se deja la optimización para el final.

8- No se hacen horas extra de trabajo.

Fase IV: Pruebas

1- Todo el código debe tener pruebas unitarias, y debe pasarlas antes de ser lanzado.

2- Cuando se encuentra un error de codificación o bug, se desarrollan pruebas para evitar caer en el mismo error.

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

1.6 Lenguaje de programación.

Un lenguaje de programación es un lenguaje que puede ser utilizado para controlar el comportamiento de una máquina, especialmente una computadora. Permite a los programadores especificar sobre qué datos la computadora debe operar, cómo estos deben ser almacenados, y qué acciones debe tomar ante cada circunstancia previamente definida. Al ser un estándar de escritura permite a más de un programador trabajar de forma colaborativa en la construcción de un programa (Gutiérrez, 2007).

En el transcurrir de los años y en la medida en que la tecnología ha ido avanzado, han venido surgiendo diferentes lenguajes de programación, cada uno con características y objetivos específicos distintos pero todos con la misma finalidad, la comunicación hombre-máquina a través de una estructura sintáctica similar al lenguaje común utilizado en la vida diaria.

1.6.1 Lenguajes de programación a utilizar para el desarrollo de la aplicación

Para la selección de los lenguajes de programación a utilizar fueron objeto de análisis los siguientes: C++ (Wikipedia, 2004), Delphi (Jiménez, 2010), además de los siguientes lenguajes que fueron los escogidos para el desarrollo del sistema.

SQL

El lenguaje estructurado de consultas (SQL) es un lenguaje de base de datos normalizado, utilizado por la gran mayoría de los servidores de bases de datos que manejan bases de datos relacionales u objeto-relacionales.

Es un lenguaje declarativo en el que las órdenes especifican cual debe ser el resultado y no la manera de conseguirlo (como ocurre en los lenguajes procedimentales). Al ser declarativo es muy sistemático, sencillo y con una curva de aprendizaje muy agradable ya que sus palabras clave permiten escribir las órdenes como si fueran frases en las que se especifica (en inglés) que es lo que queremos obtener.

Se ha convertido, debido a su eficiencia, en un estándar para las bases de datos relacionales, de hecho el gran éxito del modelo de base de datos relacional se debe en parte a la utilización de un lenguaje como SQL. A pesar de su tesórico carácter estándar, se han desarrollado, sobre una base común, diversas versiones ampliadas como las de Oracle o la de Microsoft SQL server.

HQL

Es el lenguaje que nos proporciona Hibernate para el manejo de consultas a la base de datos. Este lenguaje es similar a SQL y es utilizado para obtener objetos de la base de datos según las condiciones especificadas en el HQL. El uso de HQL nos permite usar un lenguaje intermedio que según la base de datos que usemos y el dialecto que especifiquemos, será traducido al SQL dependiente de cada base de datos de forma automática y transparente.

XML

XML proviene de eXtensibleMarkupLanguage (“Lenguaje de Marcas Extensible”). Se trata de un metalenguaje (un lenguaje que se utiliza para decir algo acerca de otro) extensible de etiquetas que fue desarrollado por el Word Wide Web Consortium (W3C), una sociedad mercantil internacional que elabora recomendaciones para la World Wide Web.

Las bases de datos, los documentos de texto, las hojas de cálculo y las páginas web son algunos de los campos de aplicación del XML. El

metalenguaje aparece como un estándar que estructura el intercambio de información entre las diferentes plataformas.

XML presenta una serie de ventajas muy atractivas para los desarrolladores, especialmente porque permite relacionar aplicaciones de diferentes lenguajes y plataformas; sin embargo, esto mismo puede ser visto como un arma de doble filo, dado que no incentiva la búsqueda de compatibilidad. La universalidad que persigue XML puede no llegar jamás si en lugar de aprovecharlo para resolver problemas, se generan nuevos sabiendo que tendrán una solución.

XML cumple un papel muy importante que es, sin lugar a dudas, su punto fuerte: le permite comunicarse con otras aplicaciones de diferentes plataformas y sin que importe el origen de la información en común. Se pueden tener, por ejemplo, un programa corriendo en Windows con una base de datos de SQL Server, y otro en Linux con Oracle, ambos compartiendo datos gracias a una estructura en XML.

Por último, XML es una de esas herramientas que a pesar de su poca complejidad esconden un gran potencial, gracias a ser fácil de usar e innegablemente útil.

Java

Java es un lenguaje de programación orientado a objetos, desarrollado por Sun Microsystems a principios de los años 90. El lenguaje en sí mismo toma mucha de su sintaxis de C y C++, pero tiene un modelo de objetos más simple y elimina herramientas de bajo nivel, que suelen inducir a muchos errores, como la manipulación directa de punteros o memoria. Con respecto a la memoria, su gestión no es un problema ya que esta es gestionada por el propio lenguaje y no por el programador.

Java es un lenguaje orientado a objetos, aunque no de los denominados puros; en Java todos los tipos, a excepción de los tipos fundamentales de variables (int, char, long...) son clases. Sin embargo, en los lenguajes orientados a objetos puros incluso estos tipos fundamentales son clases, por ejemplo en Smalltalk.

El código generado por el compilador Java es independiente de la arquitectura: podría ejecutarse en un entorno UNIX, Mac o Windows, es decir, es un lenguaje multiplataforma. El motivo de esto es que el que realmente ejecuta el código generado por el compilador no es el procesador del ordenador directamente, sino que este se ejecuta mediante una máquina virtual. Esto permite que los Applets de una web pueda ejecutarlos cualquier máquina que se conecte a ella independientemente de qué sistema operativo emplee (siempre y cuando el ordenador en cuestión tenga instalada una máquina virtual de Java).

1.7 Gestores de Bases de Datos.

Una Base de Datos (BD) es un conjunto de datos interrelacionados entre sí, almacenados con carácter más o menos permanente en la computadora, o sea, que puede considerarse una colección de datos variables en el tiempo. El software que permite la utilización y la actualización de los datos almacenados en una o varias bases de datos por uno o varios usuarios desde diferentes puntos de vista y a la vez, se denomina Sistema de Gestión de Bases de Datos (SGBD), cuyo objetivo fundamental consiste en suministrar al usuario las herramientas que le permitan manipular los datos en términos abstractos, de forma que no necesite conocer el modo de almacenamiento de los mismos en la computadora, ni el método de acceso empleado (Guerra, 2009).

1.7.1 Sistema gestor de Bases de Datos (SGBD): PostgreSQL 9.1

PostgreSQL es un sistema de base de datos objeto-relacional de código abierto bajo licencia GPL. Cuenta con más de 15 años de desarrollo activo y una arquitectura probada que se ha ganado una sólida reputación de fiabilidad, integridad y corrección de datos. Se ejecuta en todos los principales sistemas operativos, como son Linux, UNIX, Mac OS X y Windows. Tiene soporte completo para claves foráneas, uniones, vistas, disparadores y procedimientos almacenados. Incluye la mayoría de las sentencias SQL. También soporta almacenamiento de objetos binarios grandes, como imágenes, sonidos o

videos. Cuenta con interfaces nativas de programación para C/C++, Java, .Net, Python, Ruby, entre otros, y la documentación excepcional.

Ventajas

- Instalación ilimitada: Con PostgreSQL, nadie puede demandarlo por violar acuerdos de licencia, puesto que no hay costo asociado a la licencia del software.
- Soporte: Además de nuestras ofertas de soporte, tenemos una importante comunidad de profesionales y entusiastas de PostgreSQL de los que su compañía puede obtener beneficios y contribuir.
- Ahorros considerables en costos de operación: PostgreSQL ha sido diseñado y creado para tener un mantenimiento y ajuste mucho menor que otros productos, conservando todas las características, estabilidad y rendimiento.
- Estabilidad y confiabilidad: PostgreSQL nunca ha presentado caídas en varios años de operación de alta actividad. Ni una sola vez. Simplemente funciona.
- Extensible: El código fuente está disponible para todos sin costo. Si su equipo necesita extender o personalizar PostgreSQL de alguna manera, pueden hacerlo con un mínimo esfuerzo, sin costos adicionales. Esto es complementado por la comunidad de profesionales y entusiastas de PostgreSQL alrededor del mundo que también extienden PostgreSQL todos los días.
- Multiplataforma: PostgreSQL está disponible en casi cualquier Unix (34 plataformas en la última versión estable), y ahora en versión nativa para Windows.

¿Por qué utilizar PostgreSQL?

Se opta por usar PostgreSQL porque fue una tecnología requerida por el cliente. Además las mejoras de rendimiento, características y funciones de agregación de múltiples columnas, así como índices invertidos generalizados, que constituye una forma más escalable y programable de indexar datos semi-estructurados y texto. Es el sistema de gestión de bases de datos de código

abierto más avanzado del mundo y se ejecuta en casi todos los principales sistemas operativos.

1.8 Herramientas para el desarrollo de la aplicación.

1.8.1 Framework ORM: Hibernate 3.3

Se estima que en la construcción de aplicaciones que incluyen el trabajo con bases de datos alrededor del 35% del código está destinado a establecer la correspondencia Objeto-Relacional.

- Las bases de datos relacionales trabajan con tablas, los lenguajes OO trabajan con clases.
- Las tablas tienen columnas, las clases poseen atributos.
- Las instancias de las tablas son las filas, las instancias de las clases son los objetos.

Este singular problema ha sido resuelto con la inserción de los ORM (Capa de persistencia objeto/relacional). ¿Qué ventajas presenta la utilización de un ORM para crear nuestras soluciones?

- Permiten reducir susceptiblemente el código necesario para llevar a cabo las operaciones de persistencia y recuperación de objetos.
- Proporcionan interfaces más simples para el manejo de objetos a través de su propio lenguaje de consulta.
- Proveen al programador de configuraciones que le permiten optimizar los tiempos de respuesta en sus correspondientes aplicaciones.
- Solo tendremos que definir la forma en la que establecemos la correspondencia entre las clases y las tablas una sola vez (indicando qué propiedad se corresponde con qué columna, qué clase con qué tabla, etc.).
- Después de esto, podremos utilizar los objetos de nuestra aplicación y decirle a nuestra ORM que los haga persistentes, con una instrucción similar a: *orm.save(myObject).*

Hibernate es una herramienta para la plataforma Java que facilita el mapeo de atributos entre una base de datos relacional y el modelo de objetos de una

aplicación, mediante archivos declarativos (XML) que permiten establecer estas relaciones, siguiendo la DTD de mapeo de Hibernate. Desde estos podremos generar el código de nuestros objetos persistentes en clases Java y también crear bases de datos independientemente del entorno escogido. Su surgimiento está basado en el problema conocido como Impedancia Objeto-Relacional.

Hibernate es una capa de persistencia objeto/relacional y un generador de sentencias SQL. Permite diseñar objetos persistentes que podrán incluir polimorfismo, relaciones, colecciones, y un gran número de tipos de datos. De una manera muy rápida y optimizada podremos generar bases de datos en cualquiera de los entornos soportados: Oracle, PostgreSQL, MySql y otros. Además es una herramienta open source que se integra en cualquier tipo de aplicación justo por encima del contenedor de datos. Se destaca también la existencia de un lenguaje de consultas propio, el Hibernate Query Language (HQL). (Hibernate, 2014)

1.8.2 NetBeans 7.3

Es un software, en el cual se pueden crear programas en un lenguaje de programación determinado, de manera rápida y fácil. Es una herramienta libre y gratuita. Permite programar aplicaciones principalmente en Java, pero también admite otros lenguajes como PHP. Algo muy importante de NetBeans es que es compatible con diversos sistemas operativos, tal como lo es Windows, Mac, Linux o Solaris, además de tener una fácil instalación.

NetBeans posee múltiples ventajas, entre ellas se destacan las siguientes:

- Soporte a java script
- Intérprete de fondo (Background Parser) capaz de identificar errores sintácticos en tiempo de edición
- Completamiento de código
- Marcado sintáctico que presenta en diferentes estilos de letras, palabras claves, identificadores estándares y literales en general, facilitando la claridad del código
- Integración con subversión

- Soporte a documentación tanto para java script como para PHP

1.8.3 Administrador de Bases de Datos PostgreSQL: pgAdmin3 1.14

El pgAdmin 3 es una herramienta de código abierto para la administración de bases de datos PostgreSQL y derivados (EnterpriseDB Postgres Plus Advanced Servery Greenplum Database). Incluye:

- Interfaz administrativa gráfica
- Herramienta de consulta SQL(con un EXPLAIN gráfico)
- Editor de código procedural
- Agente de planificación SQL/shell/batch
- Administración de Slony-I

pgAdmin se diseña para responder a las necesidades de la mayoría de los usuarios, desde escribir simples consultas SQL hasta desarrollar bases de datos complejas.

La interfaz gráfica soporta todas las características de PostgreSQL y hace simple la administración. Está disponible en más de una docena de lenguajes y para varios sistemas operativos, incluyendo Microsoft Windows, Linux, FreeBSD, Mac OSX y Solaris.

pgAdmin III soporta versiones de servidores 7.3 y superiores. Versiones anteriores a 7.3 deben usar pgAdmin II.

Características

En pgAdmin3 se puede ver y trabajar con casi todos los objetos de la base de datos, examinar sus propiedades y realizar tareas administrativas.

- Agregados
- Casts
- Columnas
- Constraints
- Conversiones
- Bases de datos

- Dominios
- Funciones
- Grupos
- Índices
- Lenguajes (PLpgsql, PLpython, PLperl,otras)
- Clases de operadores
- Operadores
- Servidores PostgreSQL
- Reglas
- Esquemas
- Secuencias
- Tablas
- Triggers
- Tipos de datos
- Usuarios
- Vistas

Una característica interesante de pgAdmin3 es que, cada vez que se realiza alguna modificación en un objeto, escribe la(s) sentencia(s) SQL correspondiente(s), lo que hace que, además de una herramienta muy útil, sea a la vez didáctica. También incorpora funcionalidades para realizar consultas, examinar su ejecución (como el comando explain) y trabajar con los datos.

1.9 Herramientas CASE

Las Herramientas CASE (Computer Aided Systems Engineering – Ingenieria de Software Asistida por Ordenador) se pueden definir como un conjunto de programas y ayudas que dan asistencia a los analistas, ingenieros de software y desarrolladores, durante todos los pasos del ciclo de vida de desarrollo de un software. Existen herramientas CASE de trabajo visuales como el Rational Rose y el Visual Paradigm, entre otras que permiten realizar el modelado del desarrollo de los proyectos.

1.9.1 Visual Paradigm 6.4

Es una herramienta que interpreta UML profesional, que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Este software ayuda a una más rápida construcción de aplicaciones de calidad, mejores y a un menor costo. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. Una de las características más importantes del Visual Paradigm es que es multiplataforma (Guerra, 2009).

Propósito

Visual Paradigm ha sido concebida para soportar el ciclo de vida completo del proceso de desarrollo del software a través de la representación de todo tipo de diagramas. Constituye una herramienta de software libre de probada utilidad para el analista. Fue diseñado para una amplia gama de usuarios interesados en la construcción de sistemas de software de forma fiable, a través de la utilización de un enfoque Orientado a Objetos (Freedownloadmanager.org, 2004).

Características

- Software libre
- Disponibilidad en múltiples plataformas
- Diseño centrado en casos de uso y enfocado al negocio que generan un software de mayor calidad
- Modelo y código que permanece sincronizado en todo el ciclo de desarrollo
- Disponibilidad de múltiples versiones, para cada necesidad
- Licencia gratuita y comercial. Varios idiomas. Fácil de instalar y actualizar
- Compatibilidad entre ediciones

Ventajas

- Apoya todo lo básico en cuanto a artefactos generados en las etapas de definición de requerimientos y de especificación de componentes.
- Tiene apoyo adicional en cuanto a generación de artefactos automáticamente.
- Genera modelos VP-UML instantáneamente a partir de código binario .Net.
- Generación de documentación en formatos HTML y PDF.
- Disponibilidad en múltiples plataformas: Microsoft Windows (98, 2000, XP, o Vista), Linux, Mac OS X, Solaris o Java.
- Brinda la posibilidad de intercambiar información mediante la importación y exportación de ficheros con aplicaciones como Visio y Rational Rose.
- Generación de código e ingeniería inversa: brinda la posibilidad de generar código a partir de los diagramas, para las plataformas como .Net, Java y PHP, así como obtener los diagramas a partir del código.
- Generación de documentación: brinda la posibilidad de documentar todo el trabajo sin necesidad de utilizar herramientas externas.

1.10 Estilo Arquitectónico

Define las reglas generales de la organización en términos de un patrón y las restricciones en la forma y la estructura de un grupo numeroso.

1.10.1 Arquitectura en capas

Es donde se define cómo organizar el modelo de diseño a través de capas, que pueden estar físicamente distribuidas, lo que quiere decir que los componentes de una capa solo pueden hacer referencia a componentes en capas inmediatamente inferiores. Este patrón es importante porque simplifica la comprensión y la organización de sistemas complejos, reduciendo dependencias de forma que las capas más bajas no son consistentes de ningún detalle o interfaz de las superiores.

La programación por capas es un estilo de programación en el que el objetivo primordial es separar la lógica de negocios de la lógica de diseño, un ejemplo básico de esto es separar la capa de datos de la capa de presentación del usuario.

- Capa de presentación o interface: es la capa que le permite al usuario interactuar con el sistema, captura y le comunica la información al mismo, dando un mínimo de proceso (realiza un filtrado previo para asegurarse que no haya errores de formato). Esta capa se comunica únicamente con la de negocio.
- Capa de negocio: es donde residen los programas que se ejecutan, se reciben las peticiones del usuario y se envían las respuestas tras el proceso. Se denomina capa de negocio (e incluso lógica del negocio) pues es aquí donde se establecen todas las reglas que deben cumplirse. Esta capa se comunica con la de presentación, para recibir las solicitudes y presentar los resultados, y con la capa de datos, para solicitarle al gestor de bases de datos almacenar o recuperar datos de él.
- Capa de datos: es donde residen los datos y es la encargada de acceder ellos. Está formada por uno o más gestores de base s de datos que se encargan de realizar el almacenamiento de datos, reciben solicitudes de almacenamiento o recuperación de información desde la capa de negocio.

Todas estas capas pueden residir en un mismo ordenador aunque no es lo típico. Lo más usual es que haya una multitud de ordenadores donde reside la capa de interface (son los clientes de la arquitectura cliente/servidor). Las capas de negocios y de datos pueden residir en un mismo ordenador, y si el crecimiento de las necesidades lo aconseja, pueden dividirse en dos o más ordenadores. Así, si el tamaño o complejidad de las base de datos aumenta, pueden dividirse en varios ordenadores los cuales recibirán las peticiones del ordenador donde resida la capa de negocio. Si por el contrario, la complejidad fuese en la capa de negocio lo que obligase a la separación, esta lógica del negocio podría residir en uno o más ordenadores que realizarían las solicitudes a una única base de datos. (Fernández, 2013)

1.10.2 Justificación de la arquitectura a utilizar

Proponemos como estilo arquitectónico a utilizar en nuestro proyecto la arquitectura en tres capas debido a las ventajas que esta brinda:

Ventajas

- El desarrollo se puede llevar a cabo en varios niveles
- Desarrollos paralelos (en cada capa)
- Aplicaciones más robustas debido al encapsulamiento
- En caso de que sobrevenga algún cambio, sólo se ataca al nivel requerido sin tener que revisar entre código mezclado
- Mayor flexibilidad (se pueden añadir nuevos módulos para dotar al sistema de nueva funcionalidad)
- Alta escalabilidad. La principal ventaja de una aplicación distribuida bien diseñada es su buen escalado, es decir, que puede manejar muchas peticiones con el mismo rendimiento simplemente añadiendo más hardware. El crecimiento es casi lineal y no es necesario añadir más código para conseguir esta escalabilidad.
- Las llamadas de la interfaz del usuario en la estación de trabajo, al servidor de capa intermedia, son más flexibles que en el diseño de dos capas, ya que la estación solo necesita transferir parámetros a la capa intermedia.
- Con la arquitectura de tres capas, la interfaz del cliente no es requerida para comprender o comunicarse con el receptor de los datos. Por lo tanto, esa estructura de los datos puede ser modificada sin cambiar la interfaz del usuario en la PC.
- El código de la capa intermedia puede ser reutilizado por múltiples aplicaciones si está diseñado en formato modular. La separación de roles en tres capas, hace más fácil reemplazar o modificar una capa sin afectar a los módulos restantes.

Conclusiones del capítulo

En este capítulo se abordaron elementos necesarios para la compresión y fundamentación de la solución propuesta. Se hizo una valoración de los lenguajes de programación, el sistema gestor de bases de datos que se pretenden utilizar en la futura implementación, y la metodología de desarrollo que se utilizó. Una vez conocidas las herramientas y conceptos a usar se puede proseguir con la propuesta de solución.

Capítulo 2: Planificación y Diseño

Introducción

En este capítulo, se introduce la fase de planificación y diseño, donde se detallan las necesidades del cliente, se describen las funcionalidades que serán objeto de informatización mediante el empleo de las historias de usuarios (HU), se realiza una estimación del esfuerzo necesario para las mismas y se establece un plan de iteraciones necesarias sobre el sistema para su terminación. Además se presentan las tarjetas Clases, Responsabilidades y Colaboradores (CRC), que permitirán trabajar con una metodología basada en objetos.

2.1 Propuesta de solución

En la presente investigación se propone una herramienta informática, que sirva como apoyo a la toma de decisiones ante un reordenamiento laboral, posibilitando, que la selección del personal se pueda hacer de manera rápida y más eficiente. En la actualidad, cuando se necesita emplear una persona, se elabora un perfil de competencias para el cargo y se evalúa cuáles son las que cumplen con los requisitos del mismo. Este procedimiento se realiza de manera manual. Con el sistema propuesto se pretende informatizar todo el proceso, buscando eficiencia, rapidez y minimizar los errores que puede ocurrir en el mismo.

El sistema a implementar deberá ser capaz de realizar la gestión y autenticación de los usuarios, permitir ubicar la ruta del archivo XML generado por la Ontología de perfiles de cargo listar todos los perfiles y seleccionar por cuál se va a iniciar la búsqueda de trabajadores que cumplen con dicho perfil. Luego se visualizarán los resultados de la búsqueda en los cuales se pueden seleccionar los trabajadores a emplear, de las personas empleadas se deberán introducir datos como: fecha de empleo y centro donde fue ubicado. Deberá mostrar los trabajadores empleados y desempleados dependiendo de la búsqueda requerida. Así como emitir los siguientes reportes: graficar comportamiento del proceso de empleo, mostrar cantidad de desempleados

por especialidad y detalles de trabajadores que fueron resultados de la búsqueda.

2.2 Personas relacionadas con el sistema

Tabla 2.2 Personas relacionadas con el sistema.

Personas relacionadas con el sistema	Justificación
Administrador	Es la persona responsable de administrar y controlar la aplicación, otorgándoles a los usuarios sus respectivos privilegios.
Usuario	Esta persona es la que tiene cierto conocimiento sobre la selección y capacitación del capital humano.

2.3 Funcionalidades y característica del sistema

Las funcionalidades y características del sistema son el primer artefacto generado en esta fase. Consiste en dejar explícita las funcionalidades que tendrá el producto. Tiene como objetivo asegurar que el producto definido al terminar la lista es el más correcto, útil y competitivo posible. Esta lista puede crecer y modificarse a medida que se desarrolle el producto.

Tabla 2.3 Funcionalidades y características del sistema.

Código	Funcionalidades del sistema	Prioridad
F1	Crear usuario	Media
F2	Modificar usuario	Media
F3	Eliminar usuario.	Media
F4	Autenticar usuario.	Media
F5	Búsqueda de perfil de cargo	Alta
F6	Conexión con la Ontología	Alta
F7	Cruzamiento de información	Alta
F8	Listar trabajadores buscados	Baja
F9	Selección de trabajadores a emplear	Baja
F10	Introducir datos del trabajador empleado	Media
F11	Buscar trabajadores empleados	Media
F12	Listar trabajadores empleados	Media
F13	Buscar trabajadores desempleados	Media
F14	Listar trabajadores desempleados	Media
F15	Graficar comportamiento del proceso de empleo	Alta
F16	Mostrar cantidad de desempleados por especialidad	Alta
F17	Detalles de trabajadores que fueron resultados de la búsqueda	Alta
Características del sistema		
	Usabilidad	

1	Facilidad de uso por parte de los usuarios: el sistema debe presentar una interfaz amigable que permita la fácil interacción con el mismo y llegar de manera rápida y efectiva a la información buscada. Debe, además, ser una interfaz de manejo cómodo que posibilite a los usuarios sin experiencia una rápida adaptación.
2	Especificación de la terminología utilizada: el sistema debe adaptarse al lenguaje y términos utilizados por los clientes en la rama abordada con vista a una mayor comprensión por parte del cliente de la herramienta de trabajo.
3	Emplear perfiles de usuario: diferenciar las interfaces y opciones para los usuarios que accedan al sistema según los diferentes roles que estos tengan dentro del sistema.
4	Menús: el sistema debe presentar una serie de menús tanto laterales como en barra de íconos flotantes que permitan el acceso rápido a la información por parte de los usuarios, aprovechando así las potencialidades de estas estructuras.
5	Interfaces intuitivas: Potencialidades de capacitación orientadas a interfaces intuitivas, lo que enaltece la posibilidad de que el usuario aprenda mediante el uso y explotación de la herramienta.
	Fiabilidad
6	Seguridad de las bases de datos: la seguridad de la base de datos está a nivel de roles, con el fin de mantener la integridad de los datos en función del acceso de cada uno de ellos, trayendo consigo además la protección de la información.
7	Políticas de seguridad por usuario y rol: el sistema debe contar con un grupo de políticas de accesibilidad a las diferentes funcionalidades del mismo en dependencia del nivel de autorización que presente un usuario determinado.
	Eficiencia
8	Los algoritmos implementados deben ser los más óptimos posibles, para garantizar una respuesta rápida en el procesamiento de los datos. Pues se podría manejar un gran volumen de información.
	Restricciones de diseño
9	Servidor de base de datos con PostgreSQL 9.1 o superior bajo el sistema operativo Windows 7 o superior.
	Componentes Comprados

10	Este acápite no procede, para el desarrollo del sistema no fue necesario comprar ningún componente.
	Interfaz
11	Interfaz: la interfaz deberá ser sencilla con colores suaves a la vista y sin cúmulo de imágenes u objetos que distraigan al cliente del objetivo de su empleo.
	Interfaces de Comunicación
12	Este acápite no procede, el sistema no presenta comunicaciones a otros sistemas o dispositivos como las redes de área locales.
	Requisitos de Licencia
13	No hay ninguna restricción de uso para el sistema.

2.4 Historias de Usuario

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

Tabla 2.4.1 HU No.3 Búsqueda de perfiles de cargo.

Historia de Usuario	
Código: HU3.	**Nombre Historia de Usuario:** Búsqueda de perfiles de cargo.
Modificación de Historia de Usuario Número: Ninguna.	
Referencia: F5, F6.	
Programador: Taylor Leandro Ma Pérez.	**Iteración Asignada:** Primera
Prioridad: Alta	**Puntos Estimados:** 1.
Riesgo en Desarrollo: Alto	**Puntos Reales:** 1.
Descripción: La HU “Búsqueda de perfil de cargo” permite buscar el perfil de cargo deseado. Se busca la ruta del archivo XML generado por la ontología. Ya cargado el mismo se listan los perfiles de cargos incluidos en la ontología.	

Luego se escoge cual se va a aplicar en la búsqueda de trabajadores que cumplen con dicho perfil.

Luego se selecciona la opción aceptar.

Una vez aceptado el perfil de cargo se parsea la ontología y se transforma en entidades físicas del negocio (son instancias de objetos que después serán utilizados para emparejar la información cargada en memoria cuando está en BD).

Observaciones:

Prototipo de interface:

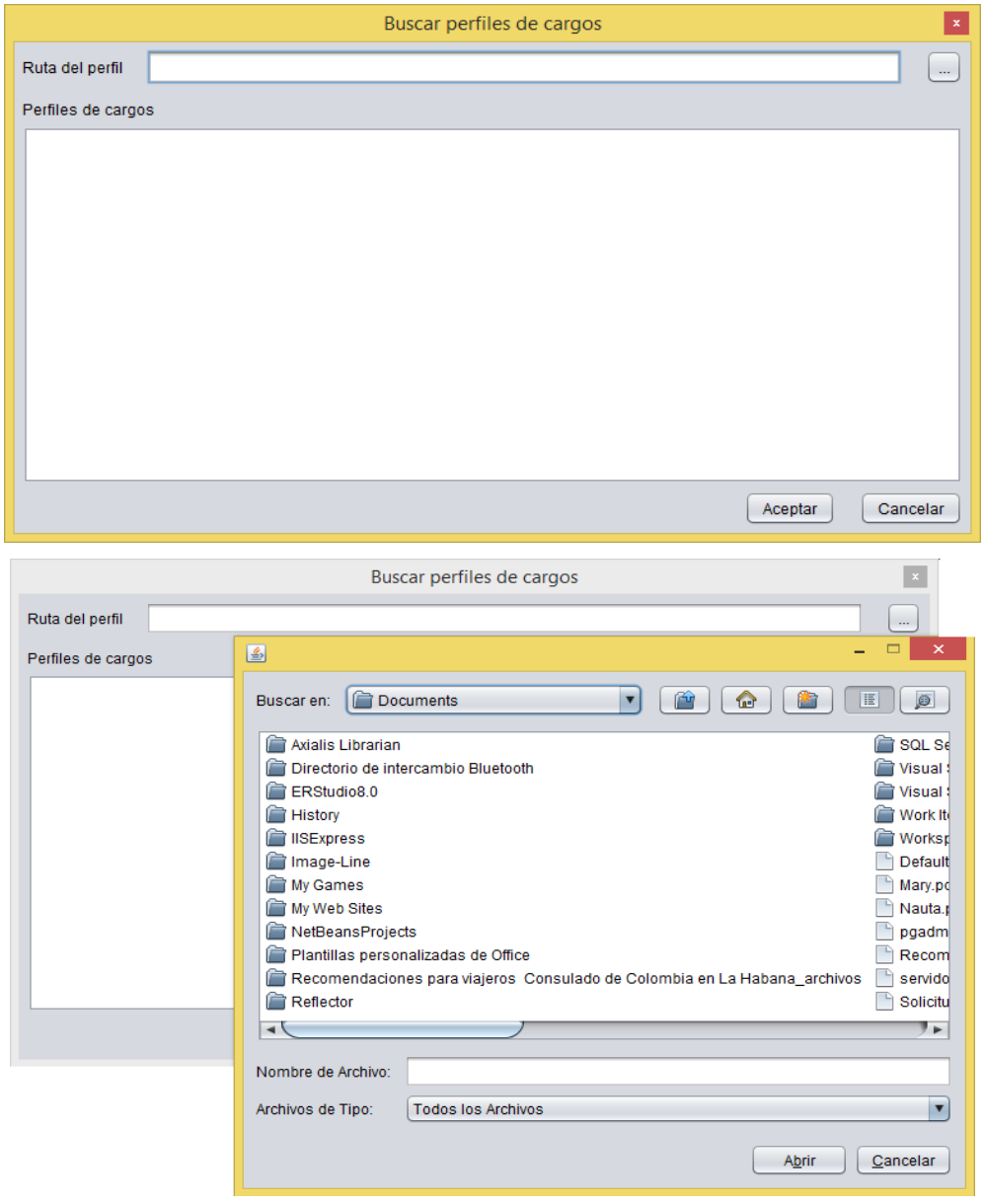


Tabla 2.4.2 HU No.4 Conexión con la Ontología.

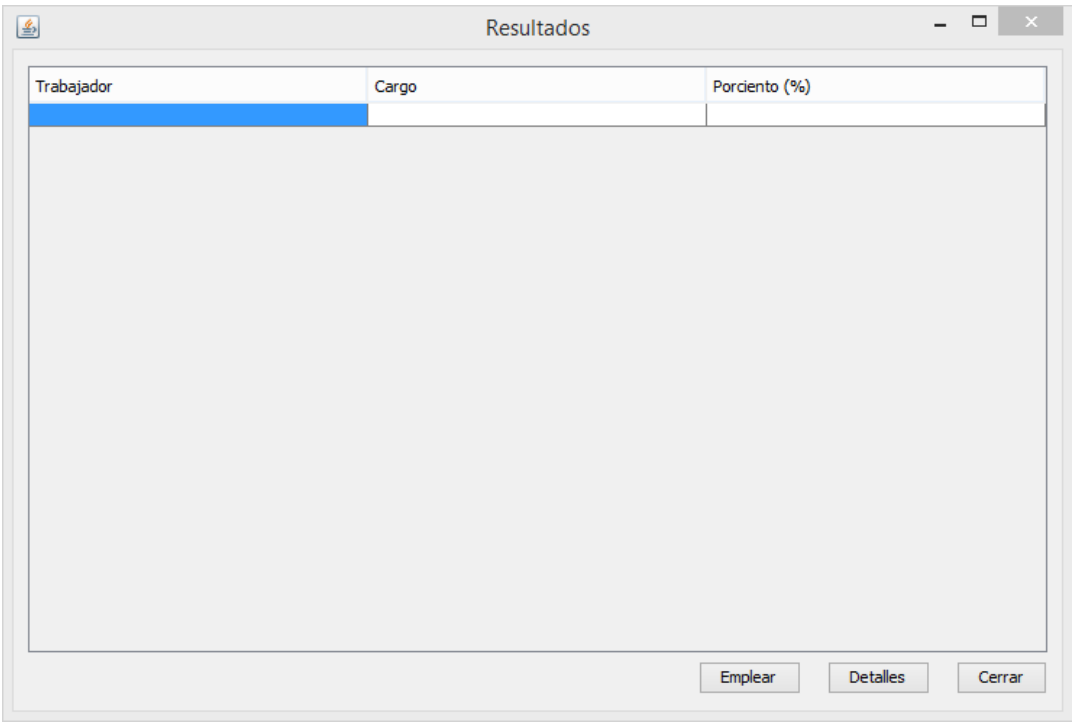
Historia de Usuario	
Código: HU4.	**Nombre Historia de Usuario:** Conexión con la Ontología.
Modificación de Historia de Usuario Número: Ninguna.	
Referencia: F6.	
Programador: Taylor Leandro Ma Pérez.	**Iteración Asignada:** Primera
Prioridad: Alta	**Puntos Estimados:** 1.
Riesgo en Desarrollo: Alto	**Puntos Reales:** 1.
Descripción: La HU “Conexión con la Ontología” permite conectarse a la Ontología. Una vez aceptado el perfil de cargo se parsea la ontología y se transforma en entidades físicas del negocio (son instancias de objetos que después serán utilizados para emparejar la información cargada en memoria cuando está en BD).	
Observaciones:	
Prototipo de interface:	

Tabla 2.4.3 HU No.5 Mostrar trabajadores buscados.

Historia de Usuario	
Código: HU5.	**Nombre Historia de Usuario:** Mostrar trabajadores buscados.
Modificación de Historia de Usuario Número: Ninguna.	
Referencia: F7, F8 y F9.	
Programador: Taylor Leandro Ma Pérez.	**Iteración Asignada:** Segunda.
Prioridad: Alta.	**Puntos Estimados:** 1.
Riesgo en Desarrollo: Alto.	**Puntos Reales:** 1.
Descripción: La HU “Mostrar trabajadores buscados” permite mostrar los trabajadores que fueron resultado de la aplicación de los requisitos del perfil seleccionado. Luego se selecciona el o los trabajadores que se van a emplear. Se selecciona la opción “Emplear”. Para visualizar los detalles de los trabajadores resultantes se selecciona la opción “Detalles”.	
Observaciones: Los trabajadores se mostraran según el % de coincidencia con los requisitos	

del cargo en una escala de mayor coincidencia a menor.

Prototipo de interface:



Ver Historias de Usuarios: Anexo 1.

2.5 Plan de entrega

En esta fase se establece la prioridad de cada HU, y a continuación, se realiza una estimación del esfuerzo necesario de cada una de ellas por parte de los programadores. Se toman acuerdos sobre el contenido de la primera entrega y se determina un cronograma en conjunto con el cliente. Una entrega debe obtenerse en no más de dos a tres meses.

Las estimaciones asociadas a la implementación de las historias se establecen empleando como medida el punto de estimación. Un punto de estimación equivale a una semana ideal de programación, donde los miembros de los equipos de desarrollo, trabajan el tiempo planeado sin ningún tipo de interrupción, este punto de estimación que se utiliza para representar la semana ideal, es de 5 días. Las historias generalmente tienen un valor entre 0.25 y 1 puntos.

2.5.1 Estimación del esfuerzo por historias de usuario

Para el buen desarrollo de la aplicación se realizó una estimación para cada una de las historias de usuario identificadas, y se obtienen los resultados que se muestran a continuación:

Tabla 2.5.1 Estimación de esfuerzo por HU.

Historias de usuarios	Puntos de estimación
Gestionar usuarios	0.50
Autenticar usuario	0.50
Búsqueda de perfil de cargo	1
Conexión con la Ontología	1
Mostrar trabajadores buscados	1
Introducir datos del trabajador empleado	0.50
Mostrar trabajadores	0.50
Graficar comportamiento del proceso de empleo	1
Mostrar cantidad desempleados por especialidad	1
Detalles de trabajadores que fueron resultados de la búsqueda	1

2.6 Plan de iteraciones

Después de realizar un análisis de las historias de usuarios y de priorizar cada una de estas se realizó el siguiente plan de iteración. El mismo tiene como entrada la relación de historias de usuario previamente definida.

Tabla 2.6 Plan de iteraciones.

Iteraciones	Descripción de las iteraciones	Orden de la HU a implementar	Duración de cada HU (días)	Duración total (días)
Primera	En esta iteración se van a implementar las HU 1, 2, 3 y	Gestionar usuario.	7 días.	28 días. (5 semanas y 3 días)
		Autenticar usuario.	7 días.	
		Búsqueda de perfil de cargo.	7 días.	

	4 que gestionan los usuarios del sistema, la HU búsqueda de perfil de cargo y la que permite la conexión con la Ontología.	Conexión con la Ontología.	7 días.	
Segunda	En esta iteración se van a realizar las HU 5, 6, 7, 8, 9 y 10 que permiten seleccionar los trabajadores a emplear y la implementación de los reportes.	Mostrar trabajadores buscados.	7 días.	42 días. (8 semanas y 2 días)
		Introducir datos del trabajador empleado.	7 días.	
		Mostrar trabajadores.	7 días.	
		Graficar comportamiento del proceso de empleo.	7 días	
		Mostrar cantidad desempleados por especialidad.	7 días	
		Detalles de trabajadores que fueron resultados de la búsqueda.	7 días	
Total			70 días	14 semanas

2.7 Tarjetas CRC

Las tarjetas CRC (clase, responsabilidad y colaboración) se realizan con el objetivo de generar jerarquías de generalización/especificación o jerarquías de agregación entre las clases, permiten identificar clases y sus responsabilidades y se hacen principalmente para realizar un diseño

simple y evitar que se implementen funcionalidades que no son necesarias en el producto que se desea obtener.

Tabla 2.7.1 Tarjeta CRC No.12 Trabajador.

Trabajador	
Descripción: Guarda información acerca de los trabajadores.	
Atributos:	
Nombre	**Descripción**
id estadoCivil nivelEscolaridad sexo nombre apellido1 apellido2 fechaNacimiento dni teléfono dirección nacionalidad hijos trabajadorExperienciaLaborals trabajadorEspecialidadeses requisitos trabajadorEmpresas	Identificador del trabajador
Responsabilidades:	
Nombre	**Colaboración**
Trabajador() getId() setId() getEstadoCivil() setEstadoCivil() getNivelEscolaridad() setNivelEscolaridad() getSexo() setSexo()	 EstadoCivil NivelEscolaridad Sexo

getNombre() setNombre() getApellido1() setApellido1() getApellido2() setApellido2() getFechaNacimiento() setFechaNacimiento() getDni() setDni() getTelefono() setTelefono() getDireccion() setDireccion() getNacionalidad() setNacionalidad() getHijos() setHijos() getTrabajadorExperienciaLaborals() setTrabajadorExperienciaLaborals() getTrabajadorEspecialidadeses() setTrabajadorEspecialidadeses() getRequisitos() setRequisitos() getTrabajadorEmpresas() setTrabajadorEmpresas()	TrabajadorExperienciaLaboral TrabajadorEspecialidades Requisitos TrabajadorEmpresas

Tabla 2.7.2 Tarjeta CRC No.27 TrabajadorDAO (Data Access Objects)

TrabajadorDAO	
Descripción: Se especializa en la persistencia de la entidad Trabajador en la BD.	
Atributos:	
Nombre	**Descripción**
Responsabilidades:	
Nombre	**Colaboración**
save()	Trabajador

create()	Trabajador
delete()	Trabajador
List<Cargos> findAll()	Trabajador
Cargos findById()	Trabajador

Ver Tarjetas CRC: Anexo 2.

Conclusiones del capitulo

En este capítulo se abordó la fase de planeación y diseño donde se delinearon las HU con la participación del cliente, se llevó a efecto la planificación de iteraciones de cada HU a partir de la estimación del esfuerzo necesario de las mismas. Presentando además, las principales clases mediante el empleo de las tarjetas CRC, culminando así esta fase y se determina que el equipo de trabajo está listo para pasar a la siguiente etapa de desarrollo.

Capítulo 3: Implementación y Pruebas.

Introducción

En este capítulo se inicia la fase de implementación y pruebas conforme a la metodología XP. Se presenta el modelo de datos empleado para la aplicación concluyente, y se realiza el desarrollo de las iteraciones a partir del desglose de las historias de usuario en tareas. Asimismo, aparecen las interfaces graficas de usuario diseñadas para la aplicación final. Se describen las pruebas realizadas y se indican las respuestas de la aplicación en el empleo de las diferentes funcionalidades, así como los posibles mensajes de error, información o aceptación que emiten las mismas cuando se utiliza una de estas funcionalidades.

3.1 Modelo de datos

En este epígrafe se muestra el modelo de datos empleado para la aplicación. Se crearon varias tablas en las que se van a almacenar los datos con los que va a trabajar la aplicación.

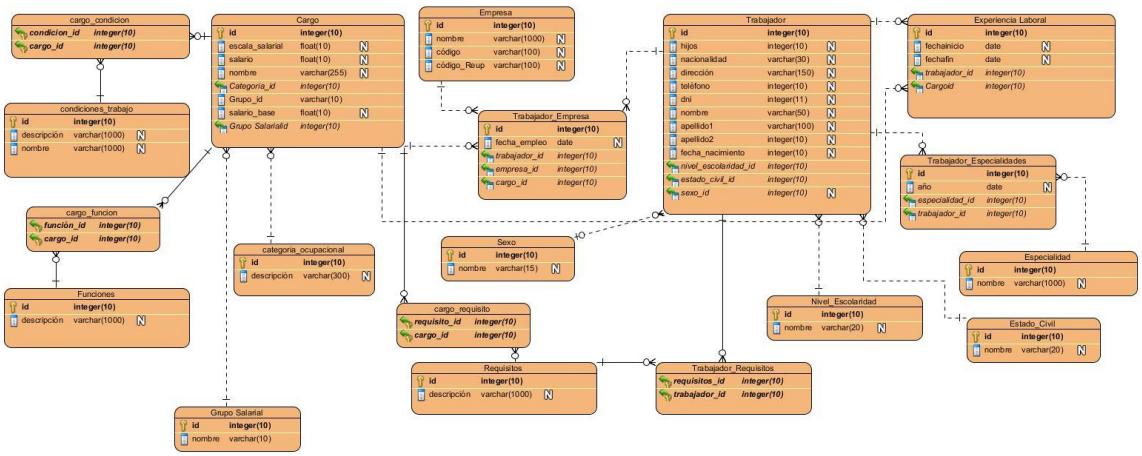


Figura 3.1 Modelo de Datos.

Ver figuras: Anexo 3.

3.2 Desarrollo de iteraciones

Durante la fase planificación y diseño fueron detalladas las historias de usuario correspondientes a cada una de las iteraciones a desarrollar, teniendo en

cuenta las prioridades y restricciones de tiempo, previstas por el cliente. Para darle cumplimiento a cada HU, primeramente se debe realizar una revisión del plan de iteraciones, y si es necesario, se le hacen modificaciones a este.

3.3 Tareas de programación por HU

Dentro del contenido de este plan, las HU se descomponen en tareas de programación o ingeniería, y a su vez, estas son asignadas al equipo de desarrollo para su implementación. Las tareas no tienen que ser entendidas necesariamente por el cliente, pues las mismas, sólo son utilizadas por los miembros del equipo de desarrollo, por lo que pueden ser escritas en lenguaje técnico. Las mismas se representan mediante las tarjetas de tareas.

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

Tabla 3.3.1 Tarea de programación No.3 Búsqueda de perfil de cargo.

Tareas de Historia de usuario	
Número tarea: P3	**Número historia:** HU3
Nombre tarea: Búsqueda de perfil de cargo.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Fecha inicio: 19 marzo 2014	**Fecha fin:** 25 marzo 2014
Responsable: Taylor Ma Pérez	
Descripción: En esta tarea se va a implementar la funcionalidad: ❖ Búsqueda de perfil de cargo.	

T**abla 3.3.2 Tarea de programación No.4 Conexión con la Ontología.**

Tareas de Historia de usuario	
Número tarea: P4	**Número historia:** HU4
Nombre tarea: Conexión con la Ontología.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Fecha inicio: 26 marzo 2014	**Fecha fin:** 2 abril 2014
Responsable: Taylor Ma Pérez	

Descripción:
En esta tarea se va a implementar la funcionalidad: ❖ Conexión con la Ontología.

Tabla 3.3.3 Tarea de programación No.5 Mostrar trabajadores buscados.

	Tareas de Historia de usuario
Número tarea: P5	**Número historia:** HU5
Nombre tarea: Mostrar trabajadores buscados.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Fecha inicio: 3 abril 2014	**Fecha fin:** 9 abril 2014
Responsable: Taylor Ma Pérez	
Descripción: En esta tarea se van a implementar las funcionalidades: ❖ Cruzamiento de la información. ❖ Listar trabajadores buscados. ❖ Selección del trabajador a emplear.	

Ver Tareas de programación: Anexo 4.

3.4 Interfaces del Software

En este epígrafe se muestra una interfaz de la aplicación referente a la visualización de trabajadores buscados.

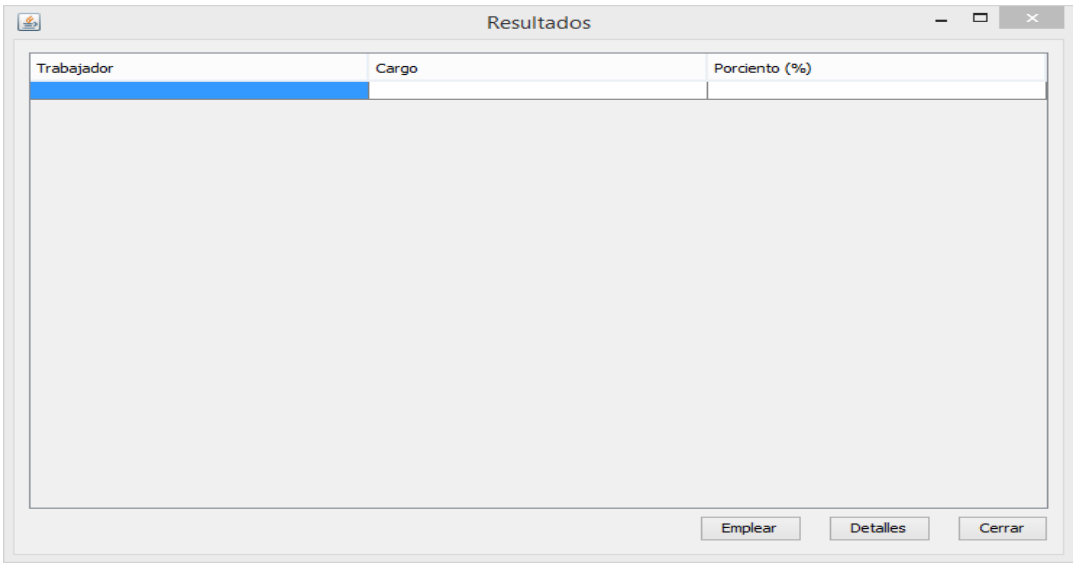


Figura 3.4.1 Mostrar trabajadores buscados.

3.5 Pruebas

En la Programación Extrema es esencial el desarrollo de las pruebas, permitiendo probar continuamente el código. Cada vez que se desea implementar las funcionalidades que tendrá el software, XP propone una redacción sencilla de prueba, para ser pasada por el código posteriormente. El proceso constante de las pruebas permite la obtención un producto con mayor calidad ofreciendo a los programadores una mayor certeza en el trabajo que desempeñan. En la metodología XP hay dos tipos de pruebas; las unitarias o desarrollo dirigido por pruebas (TDD test driven development), desarrolladas por los programadores verificando su código de forma automática, y las pruebas de aceptación, las cuáles son evaluadas luego de culminar una iteración verificando así que se cumplió la funcionalidad requerida por el cliente. Con estas normas se obtiene un código simple y funcional de manera bastante rápida y eficiente. Por esto es importante pasar las pruebas al 100%.

3.5.1 Alcance de las pruebas

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

3.5.2 Desarrollo dirigido por pruebas

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

3.5.3 Pruebas de aceptación

Las pruebas de aceptación en XP, se pueden asociar con las pruebas de caja negra que se aplican en otras metodologías de desarrollo, sólo que se crean a partir de las historias de usuario y no por un listado de requerimientos. Durante las iteraciones, las HU se traducen a pruebas de aceptación. En ellas se especifican desde la perspectiva del cliente los escenarios para probar que una historia de usuario ha sido implementada correctamente. La misma puede tener todas las pruebas de aceptación que necesite para asegurar su correcto funcionamiento. El objetivo que persiguen estas pruebas, es garantizar que las funcionalidades solicitadas por el cliente han sido realizadas. Una HU no se considera completa hasta que no ha transitado por sus pruebas de aceptación. Luego de ver los paradigmas anteriores empleados para la realización de las pruebas y reunirse con el cliente para su análisis, el mismo decidió que se lleve a cabo el proceso mediante las pruebas de aceptación.

Tabla 3.5.3.1 Prueba de aceptación #3 Búsqueda de perfil de cargo.

	Caso de prueba de aceptación
Código: (HU #3 _P3)	**Historia de usuario #3:** Búsqueda de perfil de cargo.
Nombre: Prueba para verificar la búsqueda de un perfil de cargo.	
Descripción: Esta prueba es para cargar un perfil de cargo y visualizar los datos del mismo.	
Condiciones de ejecución: El usuario debe de estar autenticado para acceder a esta sección.	
Entrada/Pasos de ejecución: El usuario da clic en el submenú “Procesar trabajadores por perfiles” del menú “Procesos” el manu principal. Da clic en el botón “…”, busca el archivo de la Ontología lo selecciona y carga.	
Resultado Esperado: Se carga el perfil de cargo requerido y se muestra el contenido del mismo.	

Tabla 3.5.3.2 Prueba de aceptación #5 Mostrar trabajadores buscados

	Caso de prueba de aceptación
Código: (HU #5 _P5)	**Historia de usuario #5:** Mostrar trabajadores buscados.
Nombre: Prueba para verificar que se muestren los trabajadores buscados	
Descripción: Se buscan los trabajadores por los criterios seleccionados.	
Condiciones de ejecución: El usuario debe estar autenticado en el sistema para acceder a esta sección.	
Entrada/Pasos de ejecución: Se da clic en el submenú ”Buscar trabajadores” de la opción “Procesos” del menú principal y se muestra la ventana de búsqueda de trabajadores. El usuario introduce los criterios por los que desea realizar la búsqueda. Luego da clic en el botón “Buscar”	
Resultado Esperado: Se muestran los datos de los trabajadores filtrados por los criterios seleccionados. **Se muestra el mensaje “No existe información a mostrar” en caso de que:** La búsqueda no arroje resultados.	

Ver Pruebas de aceptación: Anexo 5.

Conclusiones del capítulo

En este capítulo se llevó a cabo la fase de desarrollo y prueba donde se presenta el modelo de datos de la aplicación a construir, logrando una visión detallada de sus atributos y las relaciones entre sus clases. Se realizó el desarrollo de las iteraciones a partir de la distribución de tareas por HU, y se les practicó las pruebas de aceptación a las funcionalidades de mayor importancia.

Capítulo 4: Estudio de factibilidad.

4.1 Introducción

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

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

La técnica de Análisis de Costo - Beneficio, tiene como objetivo fundamental proporcionar una medida de los costos en que se incurren en la realización de un proyecto informático, y a su vez comparar dichos costos previstos con los beneficios esperados de la realización de dicho proyecto.

Esta parte es fundamental en la elaboración de cualquier proyecto pues haciendo un estudio correcto de factibilidad se puede ahorrar semanas, meses e incluso años de trabajo, hasta evitar poner en duda la reputación profesional si se realiza un sistema mal planificado desde una etapa temprana (Cortina, 2012).

4.2 Efectos económicos

Los efectos económicos pueden clasificarse como:

- Efectos directos

- Efectos indirectos
- Efectos externos
- Intangibles

4.2.1 Efectos directos

Positivos:

- La empresa que usaría el software ahorraría dinero. Pues los salarios destinados a las personas encargadas para la selección del personal en la reubicación laboral tras un cierre de minas, podrían reducirse, porque como el proceso de selección del personal estaría informatizado no serían necesarias tantas personas para llevarlo a cabo.
- Reduce el gasto de materiales de oficina utilizados en el proceso.

Negativos:

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

4.2.2 Efectos indirectos

- La organización encargada de hacer uso de este software no tendría que comprar un software propietario en el extranjero.

4.2.3 Efectos externos

Se contará con una herramienta disponible que facilitará la selección del personal, optimizando el tiempo y recursos.

4.2.4 Efectos intangibles

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

Situación sin proyecto

La selección del capital humano se realiza de forma manual, extendiendo el proceso a un período de tiempo excesivamente largo y a veces con ineficiencias.

Situación con proyecto

Con la implantación del sistema informático mejora el trabajo de los especialistas en Recursos Humanos debido a que esta herramienta informatiza la selección del capital humano y, por tanto, se reduce el tiempo de duración del proceso.

4.3 Beneficios y costos intangibles en el proyecto

- Costos: Resistencia al cambio.

Beneficios:

- Permite mayor rapidez y eficiencia en la búsqueda y análisis del personal calificado para un cargo determinado, ya que un proceso que manualmente duraría meses, incluso hasta años, se puede minimizar a semanas.

4.4 Ficha de costo

Para determinar el costo económico del proyecto se utilizará el procedimiento para elaborar una ficha de costo de un producto informático. Para la elaboración de la ficha se consideran los siguientes elementos de costo, desglosados en moneda libremente convertible y moneda nacional.

4.4.1 Costos en Moneda Libremente Convertible:

Costos Directos:

1. Compra de equipos de cómputo: No procede.

2. Alquiler de equipos de cómputo: No procede.

3. Compra de licencia de Software: No procede.

4. Depreciación de equipos: $ 64.94 CUC ($ 227.27 CUC por tres meses y medio de trabajo).

5. Materiales directos: No procede.

Total: **$ 227.27 CUC**

Costos indirectos:

1. Formación del personal que elabora el proyecto: No procede.

2. Gastos en llamadas telefónicas: No procede.

3. Gastos para el mantenimiento del centro: No procede.

4. Know How: No procede.

5. Gastos en representación: No procede.

Total: **$0.00.**

Gastos de distribución y venta:

1. Participación en ferias o exposiciones: No procede.

2. Gastos en transportación: No procede.

3. Compra de materiales de propagandas: No procede.

Total: **$0.00.**

Total general: **$ 227.27 CUC**

4.4.2 Costos en Moneda Nacional:

Costos Directos:

1. Salario del personal que laborará en el proyecto: $ 100.00 MN mensual ($350.00 por 3 meses y medio de trabajo. Nota: Este valor es el costo del esfuerzo hombre por horas de trabajo).

2. El 5% del total de gastos por salarios se dedica a la seguridad social: No procede.

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

4. Gasto por consumo de energía eléctrica: $ 58.23 MN

5. Gastos en llamadas telefónicas: No procede.

6. Gastos administrativos: No procede.

Total: $ 408.23 MN

Costos indirectos:

1. Know How: No procede.

Total: **$0.00**

Total general: **$ 408.23 MN.**

El valor del costo aproximado de una reinserción laboral en la empresa "Comandante Ernesto Che Guevara" para una muestra de 27 trabajadores es de $ 10862.58 CUC (Salazar, 2010). Por tanto, un cierre de minas afectaría gran parte del aparato económico del grupo empresarial CUBANIQUEL también afectando así diferentes instituciones del mismo, lo que conlleva a una reinserción aun mayor, y los costos de la misma serían muy elevados.

Se tomó como experimento el proceso de reordenamiento laboral llevado a cabo en nuestro instituto (ISMMM). El cual comenzó hace dos años determinándose una comisión de cinco expertos con sus respectivos sustitutos y aún no se ha terminado debido a la complejidad del mismo. En la Industria del Níquel, que posee una infraestructura mucho más grande que la de nuestra institución, el proceso sería aún más lento y engorroso, lo que traería consigo ineficiencias, antecedidas por errores humanos. Con la implantación del sistema se reduciría también el costo de este proceso, pues se reducirían los salarios destinados a las personas encargadas, y el tiempo de ejecución del proceso en cuestión. Con todo lo expuesto se demuestra la factibilidad de la presente investigación.

Como se hizo referencia anteriormente, la técnica seleccionada para evaluar la factibilidad del proyecto es la Metodología Costo-Efectividad. Dentro de esta metodología la técnica de punto de equilibrio aplicable a proyectos donde los beneficios tangibles no son evidentes, el análisis se basa exclusivamente en los costos. Para esta técnica es imprescindible definir una variable discreta que haga variar los costos. Tomaremos como costo el tiempo en minutos empleado

desde que se introducen los datos hasta el tiempo en que se muestran los resultados y el gráfico de comportamiento.

Valores de las variables (Solución manual)

1. Buscar toda la información guardada en la BD correspondiente a cada una de las personas que se perfilen para un cargo o puesto de trabajo (5 min).
2. Revisar la información recopilada para seleccionar el o los mejores candidatos que cumplen con los requisitos del cargo seleccionado. (60 min).

Valores de las variables (Solución con el sistema)

1. Realizar la búsqueda de la o las personas que cumplan con los requisitos del cargo seleccionado. (1 min).

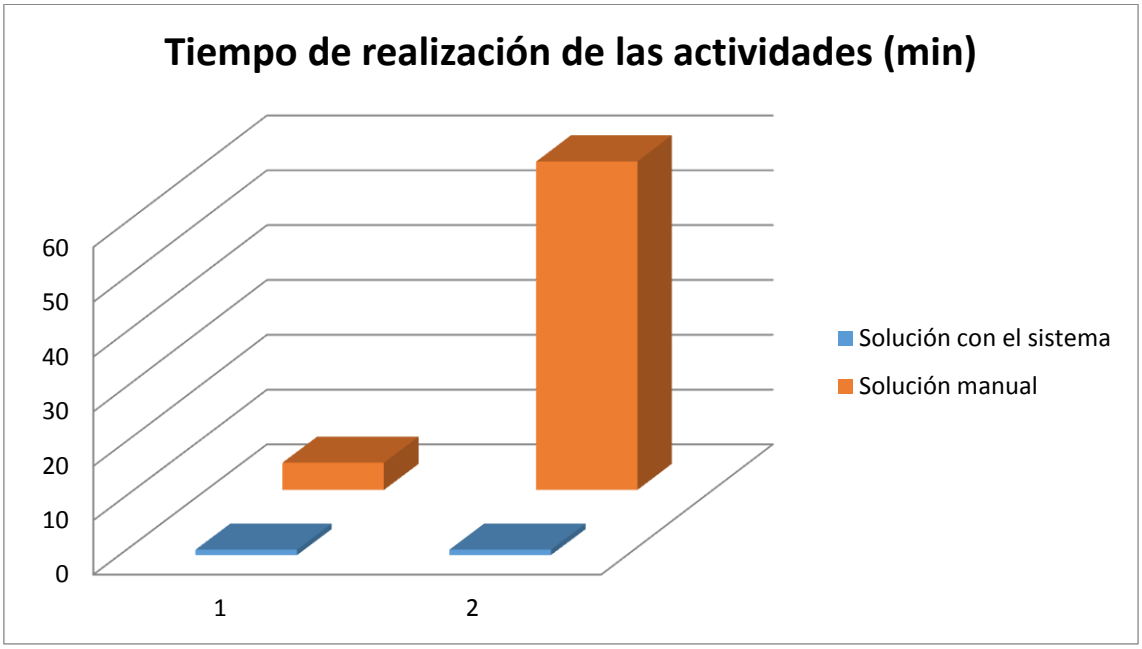


Figura 4.1 Grafica de la solución con el sistema y sin el sistema.

Teniendo en cuenta los resultados reflejados en las gráficas para cada uno de los casos queda demostrada la factibilidad del sistema evidenciada por la relación entre la complejidad del problema y el tiempo que demora la introducción y análisis de los datos de forma manual e informatizada.

Conclusiones del capítulo

A modo de conclusión podemos decir que se analizaron los efectos económicos, los beneficios y costos intangibles, y además se calculó el costo de ejecución del proyecto mediante la ficha de costo arrojando como resultado $ 227.27 CUC y $ 408.23 MN por lo que demostró ser un costo razonable pues en caso de haber usado un software propietario serían costos no rentables para el país.

Conclusiones generales

Con el desarrollo del proyecto se realizó el cumplimiento de los objetivos propuestos en esta investigación, arribándose a las siguientes conclusiones:

1. Se desarrolló el software en ayuda a la toma de decisiones para la selección del capital humano tras un reordenamiento laboral.
2. Durante el análisis de las metodologías y herramientas para el desarrollo de sistemas, se determinó la utilización de:
   - ✓ Una arquitectura de distribución: n capas
   - ✓ Metodología de desarrollo de software: XP
   - ✓ Y los lenguajes de programación: HQL, XML y JAVA
3. La documentación de sistema permitió dejar un registro que puede ser utilizado por otros desarrolladores que deseen mantener o extender el sistema. Esto se logró con la elaboración de los artefactos:
   - ✓ Funcionalidades y características del sistema
   - ✓ Historias de usuarios
   - ✓ Estimación de esfuerzo por Historias de usuarios
   - ✓ Plan de iteraciones
   - ✓ Tarjetas CRC
   - ✓ Tareas de programación
   - ✓ Casos de pruebas de aceptación
4. El estudio de factibilidad realizado siguiendo la metodología Costo-Efectividad arrojó como resultado los efectos económicos y beneficios, así como el costo de ejecución del proyecto, siendo este $ 227.27 CUC. y $ 408.23 MN demostrándose que es factible el proyecto.

Recomendaciones:

Como primeros pasos que den continuidad a este trabajo proponemos:

- Profundizar aún más en el análisis de los procesos de gestión de la información con Perfiles de cargos y competencias y Evaluaciones de competencias.
- Poner en funcionamiento del software en caso de un cierre de minero para explotar al máximo las posibilidades que brinda el software para la gestión de la información, permitiendo probar el sistema durante un período de tiempo significativo que garantice comprobar deforma práctica sus funcionalidades y obtener los datos necesarios para su perfeccionamiento.
- Agregarle nuevas funcionalidades al software.
- Mantener actualizadas la BD del sistema.

Referencias bibliográficas

- ARNOM. 2014. *Sistema Integral de Recursos Humanos*. [En línea]. [Consultado: 2014-03-05] Disponible en: http://www.internomina.com.mx/arnom
- Beck, Kent. 1999. *Extreme Programming Explained. Embrace Change. [trad.] Addison Wesley. s.l. :* Pearson Education.
- Cortina Perdomo, R. 2012. Jiménez Iglesias, E. R.; Pérez Toirac, N. *Sistema de gestión de informes de ensayos*. Tesis de Grado. Instituto Superior Minero Metalúrgico. 2010
- Ecured, 2013. *Sistema Informático*. [En línea]. [Consultado: 2014-04-22]. Disponible en: http://www.ecured.cu/index.php/Sistema_inform%C3%A1tico
- Fernández Zaldivar, Milton. *Museo Virtual de Geología del Instituto Superior Minero Metalúrgico de Moa.* Tesis de grado. Instituto Superior Minero Metalúrgico. 2013. 119h.
- FERRER, J. 2008. *Metodologías Ágiles.* [En línea] 2008. [Consultado: 2014-03-12]. Disponible en: http://libresoft.es/downloads/ferrer-20030312.pdf
- GOSEM. 2014. *Software de Gestión Humana*. [En línea]. [Consultado: 2014-03-05] Disponible en: http://www.sighsas.com/
- Guerra, Maité; Sosa Veranes y Vladimir Gonzales. 2009. *Sistema de Gestión de la Facultad 2*. Tesis de grado. 2009. págs. 38 y 50 - 54.
- Gutiérrez Saavedra, J. A. 2007. *El Mundo Informático. Software Libre.* [En línea]. [Consultado: 2014-03-03]. Disponible en: http://jorgesaavedra.wordpress.com/2007/05/05/lenguajes-de-programacion/.
- Hibernate. 2014. [En línea]. [Consultado: 2014-03-03]. Disponible en: http://www.hibernate.org/
- IARA, SAGREH. 2014. Sistema Automatizado para la Gestión de Recursos Humanos. [En línea]. [Consultado: 2014-03-05]. Disponible en: http://innovambiente.hazblog.com

- Jiménez Roche, Katiusca. 2010. *Obtención del Modelo geométrico para el diseño y explotación de canteras de materiales de construcción.* Tesis de grado. Instituto Superior Minero Metalúrgico. 2010. 78h.
- Monografías, 2010. Sistema de Información. [En línea]. [Consultado: 2014-04-22]. Disponible en: www.monografias.com/trabajos7/sisinf/sisinf.shtml
- Salazar Pérez, Yaniel. *Propuesta de una metodología para determinar los costos de reinserción laboral tras el cierre de mina en la empresa "Comandante Ernesto Che Guevara"*. Tesis de Maestría. UNIVERSIDAD DE HOLGUÍN "Oscar Lucero Moya". 2010. 87h.
- SIGEIN. 2014. *Sistema para la Gestión Integral de Capital Humano*. [En línea]. [Consultado: 2014-03-05]. Disponible en: http://www.sigein.com.mx
- Solís Álvarez, C. J. ; Figueroa Díaz, R. G. 2005. *Metodologías Tradicionales vs. Metodologías Ágiles*.
- Universidad de Oriente. *Metodología XP.* [En línea]. Disponible en: http://wikipedia.uo.edu.cu/es/articles/p/r/o/Programación_Extrema_3b63.html.
- Visual Paradigm International Ltd. 2004*. Visual Paradigm for UML.* [En línea]. [Consultado: 2014-03-03]. Disponible en: http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_%28M%C3%8D%29_14720_p..
- Wikipedia. 2004. Lenguaje de Programación C++. [En línea]. [Consultado: 2014-04-22]. Disponible en: http://wikipedia.uo.edu.cu/es/articles/c/+/+/C++.html.

Bibliografía

- Jacobson, I., Booch G., Rumbaugh J. *El Proceso Unificado de Desarrollo de Software.* Volumen II. Editorial Felix Varela, La Habana, 2004.
- Jacobson, I., Booch G., Rumbaugh J. *El Proceso Unificado de Desarrollo de Software.* Volumen I. Editorial Félix Varela, La Habana, 2004.
- Volumen I. Editorial Félix Varela, La Habana, 2004.
- Presuman R. *Ingeniería de Software. Un enfoque práctico.* Parte 1. Editorial Félix Varela, La Habana, 2005.
- Presuman R. *Ingeniería de Software. Un enfoque práctico.* Parte 2. Editorial Félix Varela, La Habana, 2005.
- CAVSI. 2004. ¿Qué es un Sistema Gestor de Bases de Datos o SGBD? http://www.cavsi.com/preguntasrespuestas/que-es-un-sistema-gestor-de-bases-de-datos-o-sgbd/.http://www.desarrolloweb.com/articulos/ sistemas-gestores-bases-datos. [En línea] 2004.
- Freedownloadmanager.org. 2004. Visual Paradigm for UML. http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_(M%C . [En línea] 2004.
- Gutiérrez, Jorge A. Saavedra. 2007. El Mundo Informático. Software Libre. http://jorgesaavedra.wordpress.com/2007/05/05/lenguajes-de-programacion/ . [En línea] 2007.

Glosario de términos

- **BD:** Una Base de Datos es un conjunto de datos relacionados entre sí, entendiéndose por dato los hechos conocidos, que pueden registrarse y tienen significado implícito.
- **Cliente:** Persona, organización o grupo de personas que solicita la construcción de un sistema, ya sea empezando desde cero, o mediante el refinamiento de versiones sucesivas.
- **Herramientas:** Son los ambientes de apoyo necesarios para automatizar las prácticas de Ingeniería de Software.
- **Interfaz:** Conjunto de representaciones de operaciones públicas.
- **ISMMM:** Instituto Superior Minero Metalúrgico de Moa.
- **Iteraciones:** En el contexto de un proyecto se refieren a la técnica de desarrollar y entregar componentes incrementales de funcionalidades de un negocio. Una iteración resulta en uno o más paquetes atómicos y completos del trabajo del proyecto que pueda realizar alguna función tangible del negocio. Múltiples iteraciones contribuyen con crear un producto completamente integrado.
- **Metodología Ágil:** Constituyen un nuevo enfoque en el desarrollo de software, mejor aceptado por los desarrolladores de proyectos que las metodologías convencionales debido a la simplicidad de sus reglas y prácticas, su orientación a equipos de desarrollo de pequeño tamaño, su flexibilidad ante los cambios y su ideología de colaboración.
- **Metodologías de Desarrollo:** Se define como un conjunto de filosofías, etapas, procedimientos, reglas, técnicas, herramientas, documentación y aspectos de formación para los desarrolladores de sistemas de información.
- **Metodologías tradicionales:** Metodologías basadas en procesos.
- **Multiplataforma:** Es un término usado para referirse a los programas, sistemas operativos, lenguajes de programación, u otra clase de software, que puedan funcionar en diversas plataformas.
- **Open Source:** Código abierto o código libre. Software que distribuye de forma libre su código fuente, de forma que los desarrolladores pueden

hacer variaciones, mejoras o reutilizarlo en otras aplicaciones. También conocido como free software.

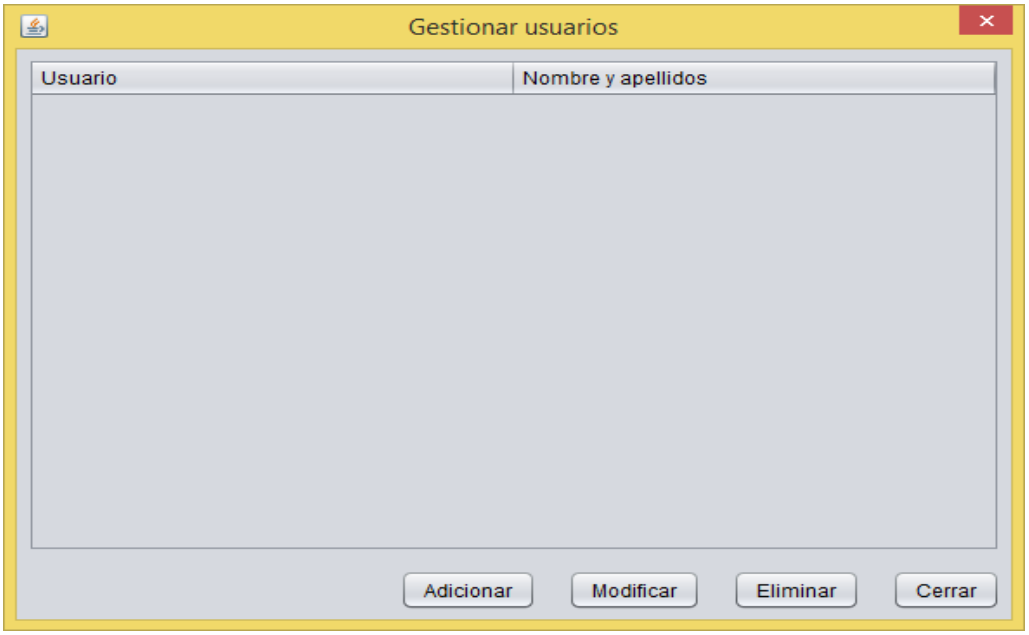
- **Programación Extrema(XP):** Es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo.
- **RUP:** El Proceso Unificado Racional o RUP (Rational Unified Process), es un proceso desarrollo de software y junto con el Lenguaje Unificado de Modelado UML, constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos. Se caracteriza por ser iterativo e incremental, estar centrado en la arquitectura y guiado por los casos de uso. Incluye artefactos y roles.
- **Software:** Es la suma total de los programas de cómputo, procedimientos, reglas, documentación y datos asociados que forman parte de las operaciones de un sistema de cómputo.
- **Software libre:** Es el software que, una vez obtenido, puede ser usado, copiado, estudiado, modificado y redistribuido libremente, aunque conserve su carácter de libre, puede ser vendido comercialmente.
- **Usuario:** Persona encargada de utilizar el sí sistema, obteniendo algún beneficio.

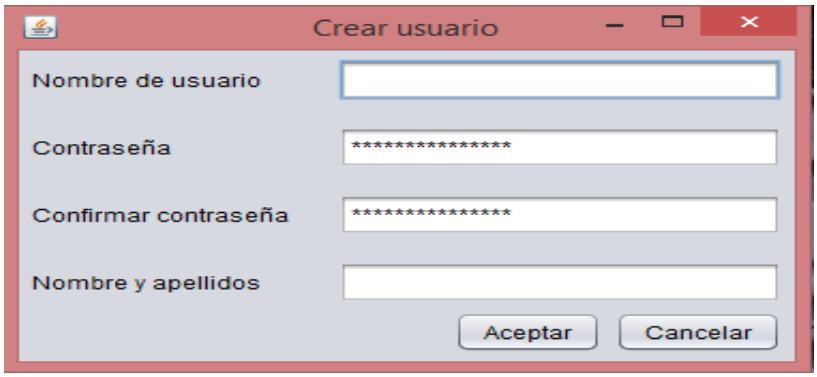
Anexos

Anexo 1: Historias de Usuario

Tabla 2.4.4 HU No.1 Gestionar usuario.

Historia de Usuario	
Código: HU1.	**Nombre Historia de Usuario:** Gestionar usuario
Modificación de Historia de Usuario Número: Ninguna.	
Referencia: F1, F2, F3.	
Programador: Taylor Leandro Ma Pérez.	**Iteración Asignada:** Primera
Prioridad: Media	**Puntos Estimados:** 0.50.
Riesgo en Desarrollo: Medio	**Puntos Reales:** 0.50.
Descripción: La HU “Gestionar usuario” permite gestionar usuarios un usuario. Se muestra un listado de los usuarios del sistema. Permitiendo realizar acciones sobre ellos como: adicionar, modificar y eliminar usuarios. Para crear un usuario se selecciona la opción “Adicionar”. Se muestra una interfaz “Crear usuario” en la cual se introducen datos como: nombre de usuario, contraseña, confirmación de contraseña, y nombre y apellidos. Para modificar un usuario se selecciona la opción “Modificar”. Se muestra una interfaz “Modificar usuario” en la cual se modifican datos como: nombre y apellidos y contraseña. Para eliminar un usuario se selecciona de la lista el usuario deseado. Se selecciona la opción “Eliminar”.	
Observaciones: 1. Para crear un usuario en caso de insertar la contraseña incorrecta muestra un mensaje “Las contraseñas deben coincidir.”	
Prototipo de interface:	





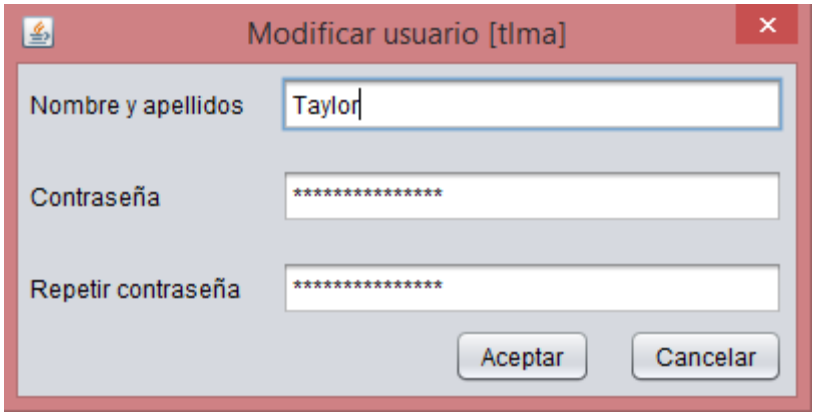


Tabla 2.4.5 HU No.2 Autenticar usuario.

	Historia de Usuario
Código: HU2.	**Nombre Historia de Usuario:** Autenticar usuario
Modificación de Historia de Usuario Número: Ninguna.	
Referencia: F4.	

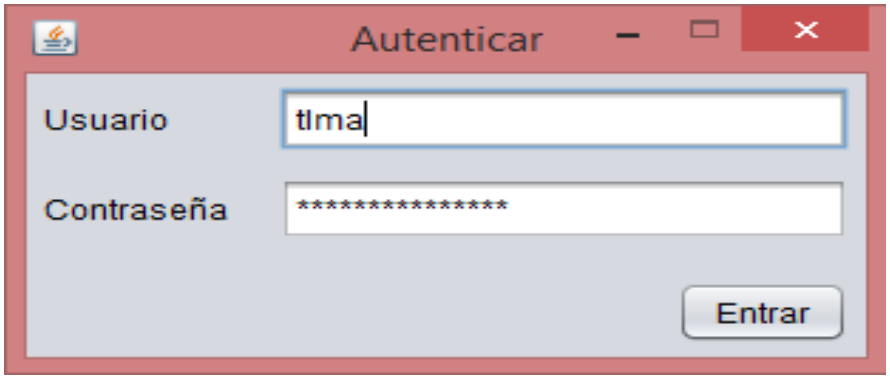
Programador: Taylor Leandro Ma Pérez.	**Iteración Asignada:** Primera
Prioridad: Media	**Puntos Estimados:** 0.50.
Riesgo en Desarrollo: Medio	**Puntos Reales:** 0.50.
Descripción: La HU “Autenticar usuario” permite autenticar un usuario.	
Observaciones: 1. Para autenticar un usuario en caso de introducir mal un campo, se muestra un mensaje “El usuario o contraseña es incorrecto”.	
Prototipo de interface: 	

Tabla 2.4.6 HU No.6 Introducir datos del trabajador empleado.

	Historia de Usuario
Código: HU6.	**Nombre Historia de Usuario:** Introducir datos del trabajador empleado.
Modificación de Historia de Usuario Número: Ninguna.	
Referencia: F10.	
Programador: Taylor Leandro Ma Pérez.	**Iteración Asignada:** Segunda
Prioridad: Media	**Puntos Estimados:** 0.50.
Riesgo en Desarrollo: Medio	**Puntos Reales:** 0.50.
Descripción: La HU “Introducir datos del trabajador empleado” permite introducir los datos del trabajador empleado. Una vez empleado el trabajador se introducen datos como: ubicación laboral y fecha de empleo. El nombre del trabajador empleado viene por defecto en el campo de texto “Trabajador”. Una vez introducido los datos se guardan los cambios y se muestra un mensaje de confirmación: “Los datos han sido guardados satisfactoriamente”.	

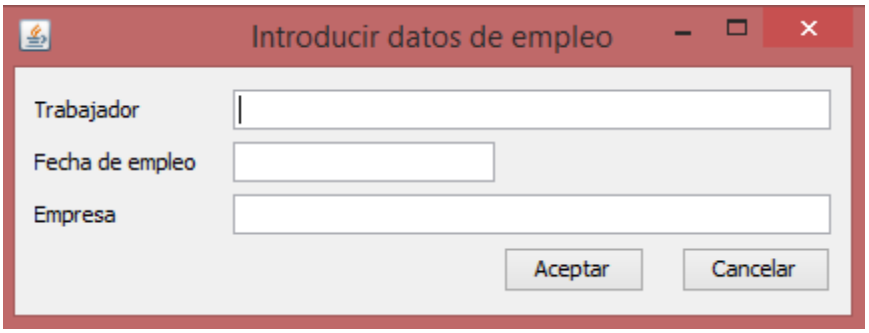
Observaciones: 1. En caso que se dejen campos obligatorios vacíos se muestra un mensaje de error “Este campo es requerido”.
Prototipo de interface: 

Tabla 2.4.7 HU No.7 Mostrar trabajadores.

Historia de Usuario	
Código: HU7.	**Nombre Historia de Usuario:** Mostrar trabajadores
Modificación de Historia de Usuario Número: Ninguna.	
Referencia: RF11, RF12, RF13 y RF14.	
Programador: Taylor Leandro Ma Pérez.	**Iteración Asignada:** Segunda.
Prioridad: Media	**Puntos Estimados:** 0.50.
Riesgo en Desarrollo: Medio	**Puntos Reales:** 0.50.
Descripción: La HU “Mostrar trabajadores” permite mostrar los trabajadores. Para mostrar los trabajadores empleados se marca la casilla empleados y se selecciona la opción buscar. Para mostrar los trabajadores desempleados solamente se selecciona la opción buscar. Esta búsqueda puede realizarse para un trabajador específico tanto empleado como desempleado introduciendo el nombre y apellidos del mismo. Esta búsqueda puede filtrarse por: nombre del trabajador, cargo y empresa	
Observaciones: En caso de no existir resultados en la búsqueda se muestra un mensaje “No existe información a mostrar”.	
Prototipo de interface:	

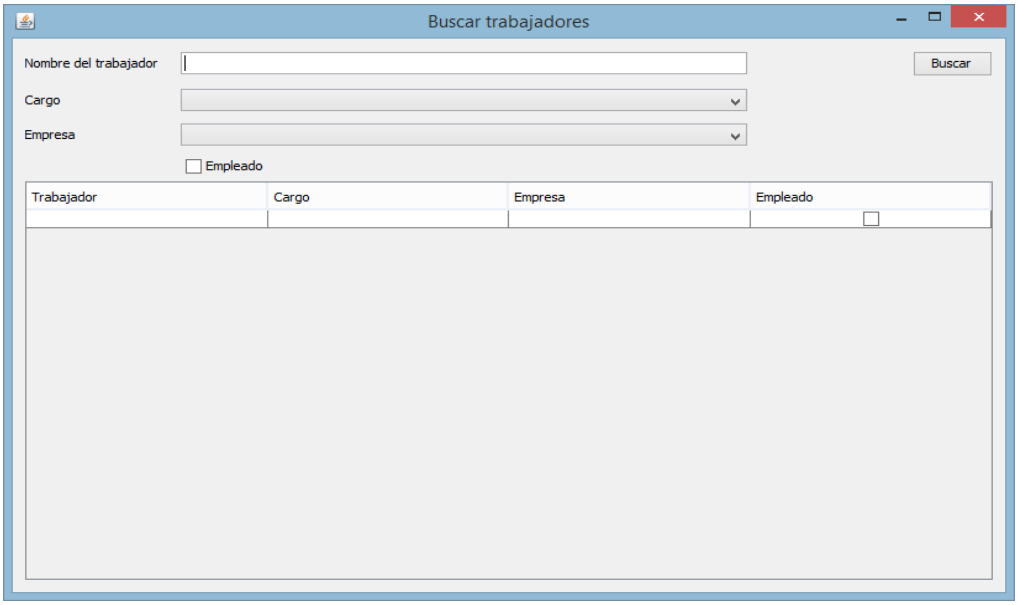


Tabla 2.4.8 HU No.8 Graficar comportamiento del proceso de empleo.

	Historia de Usuario
Código: HU8.	**Nombre Historia de Usuario:** Graficar comportamiento del proceso de empleo.
Modificación de Historia de Usuario Número: Ninguna.	
Referencia: RF15.	
Programador: Taylor Leandro Ma Pérez.	**Iteración Asignada:** Segunda
Prioridad: Alta.	**Puntos Estimados:** 1.
Riesgo en Desarrollo: Alto.	**Puntos Reales:** 1.
Descripción: La HU "Graficar comportamiento del proceso de empleo" permite mostrar el comportamiento del proceso de empleo por empresas. El cual se visualizará en una gráfica de pastel.	
Observaciones:	
Prototipo de interface:	

Tabla 2.4.9 HU No.9 Mostrar cantidad de desempleados por especialidad.

	Historia de Usuario
Código: HU9.	**Nombre Historia de Usuario:** Mostrar cantidad de

	desempleados por especialidad.
Modificación de Historia de Usuario Número: Ninguna.	
Referencia: RF16.	
Programador: Taylor Leandro Ma Pérez.	**Iteración Asignada:** Segunda
Prioridad: Alta.	**Puntos Estimados:** 1.
Riesgo en Desarrollo: Alto.	**Puntos Reales:** 1.
Descripción: La HU “Mostrar cantidad desempleados por especialidad” permite mostrar la cantidad de desempleados aún existentes en la Base de Datos por especialidad. Estos se visualizaran en una tabla.	
Observaciones:	
Prototipo de interface:	

Tabla 2.4.10 HU No.10 Detalles de trabajadores que fueron resultados de la búsqueda.

Historia de Usuario	
Código: HU10.	**Nombre Historia de Usuario:** Detalles de trabajadores que fueron resultados de la búsqueda.
Modificación de Historia de Usuario Número: Ninguna.	
Referencia: RF16.	
Programador: Taylor Leandro Ma Pérez.	**Iteración Asignada:** Segunda.
Prioridad: Alta.	**Puntos Estimados:** 1.
Riesgo en Desarrollo: Alto.	**Puntos Reales:** 1.
Descripción: La HU “Detalles de trabajadores que fueron resultados de la búsqueda” permite ver los detalles que van a ser la coincidencia de requisitos con el cargo y los que le faltan para coincidir cien por ciento con el mismo. Estos se visualizaran en una tabla.	
Observaciones:	
Prototipo de interface:	

Anexo 2: Tarjetas CRC

Tabla 2.7.4 Tarjeta CRC No.1 Especialidad.

Especialidad	
Descripción: Guarda información de la especialidad de cada trabajador.	
Atributos:	
Nombre	**Descripción**
id nombre trabajadorEspecialidades	Identificador de Especialidad
Responsabilidades:	
Nombre	**Colaboración**
getId() setId() getNombre() setNombre() getTrabajadorEspecialidadeses() setTrabajadorEspecialidadeses()	 TrabajadorEspecialidades

Tabla 2.7.5 Tarjeta CRC No.2 Nivel de Escolaridad.

Nivel de Escolaridad	
Descripción: Guarda información del nivel de escolaridad de cada trabajador.	
Atributos:	
Nombre	**Descripción**
id descripción	Identificador de Nivel de Escolaridad
Responsabilidades:	
Nombre	**Colaboración**
getId() setId() getNombre() setNombre() getTrabajadors() setTrabajadors()	 Trabajador

Tabla 2.7.6 Tarjeta CRC No.3 Empresa.

Empresa	
Descripción: Guarda información de las empresas.	
Atributos:	
Nombre	**Descripción**
id nombre código codigoReup trabajadorEmpresas	Identificador de Empresa
Responsabilidades:	
Nombre	**Colaboración**
getId() setId() getNombre() setNombre() getCodigo() setCodigo() getCodigoReup() setCodigoReup() getTrabajadorEmpresas() setTrabajadorEmpresas()	 TrabajadorEmpresas

Tabla 2.7.7 Tarjeta CRC No.4 Funciones.

Funciones	
Descripción: Guarda información de las funciones.	
Atributos:	
Nombre	**Descripción**
id descripción	Identificador de Funciones
Responsabilidades:	
Nombre	**Colaboración**
getId() setId() getDescripcion()	

setDescripcion()	

Tabla 2.7.8 Tarjeta CRC No. 5 Condiciones Trabajo.

Condiciones Trabajo	
Descripción: Guarda información de las condiciones de trabajo.	
Atributos:	
Nombre	**Descripción**
id descripción nombre cargo	Identificador de Condiciones de Trabajo
Responsabilidades:	
Nombre	**Colaboración**
getId() setId() getDescripcion() setDescripcion() getNombre() setNombre() getCargos() setCargos()	 Cargo

Tabla 2.7.9 Tarjeta CRC No.6 Cargos.

Cargos	
Descripción: Guarda información de los cargos.	
Atributos:	
Nombre	**Descripción**
id grupoSalarial categoriaOcupacional escalaSalarial salarioBase nombre trabajadorEmpresas	Identificador de Cargos

cargosesForCargoId condicionesTrabajos cargosesForFuncionId requisitos	
Responsabilidades:	
Nombre	**Colaboración**
getId() setId() getGrupoSalarial() setGrupoSalarial() getCategoriaOcupacional() setCategoriaOcupacional() getEscalaSalarial() setEscalaSalarial() getSalarioBase() setSalarioBase() getNombre() setNombre() getTrabajadorEmpresas() setTrabajadorEmpresas getCargosesForCargoId() setCargosesForCargoId() getCondicionesTrabajos() setCondicionesTrabajos() getCargosesForFuncionId() setCargosesForFuncionId() getRequisitos() setRequisitos()	 GrupoSalarial CategoriaOcupacional TrabajadorEmpresas CondicionesTrabajos Función Requisitos

Tabla 2.7.10 Tarjeta CRC No.7 Categoría Ocupacional.

Categoría Ocupacional	
Descripción: Guarda información de las categorías ocupacionales.	
Atributos:	
Nombre	**Descripción**
id	Identificador de Categoría Ocupacional

descripción cargo	
Responsabilidades:	
Nombre	**Colaboración**
getId() setId() getDescripcion() setDescripcion() getCargos() setCargos()	 Cargos

Tabla 2.7.11 Tarjeta CRC No.8 Estado Civil.

Estado Civil	
Descripción: Guarda información acerca de los estados civiles.	
Atributos:	
Nombre	**Descripción**
id nombre trabajadors	Identificador de Estado Civil
Responsabilidades:	
Nombre	**Colaboración**
getId() setId() getNombre() setNombre() getTrabajadors() setTrabajadors()	 Trabajador

Tabla 2.7.12 Tarjeta CRC No.9 Grupo Salarial.

Grupo Salarial	
Descripción: Guarda información acerca de los grupos salariales.	
Atributos:	
Nombre	**Descripción**
id	Identificador de Grupo Salarial

nombre cargo	
Responsabilidades:	
Nombre	**Colaboración**
getId() setId() getNombre() setNombre() getCargoses() setCargoses()	 Cargo

Tabla 2.7.13 Tarjeta CRC No.10 Requisito.

Requisito	
Descripción: Guarda información acerca de los Requisitos.	
Atributos:	
Nombre	**Descripción**
id descripción cargoses trabajadors	Identificador de Requisito
Responsabilidades:	
Nombre	**Colaboración**
getId() setId() getDescripcion() setDescripcion() getCargoses() setCargoses() getTrabajadors() setTrabajadors()	 Cargo Trabajador

Tabla 2.7.14 Tarjeta CRC No.11 Sexo.

Sexo	
Descripción: Guarda información acerca del sexo.	
Atributos:	
Nombre	**Descripción**
id nombre trabajadors	Identificador del Sexo
Responsabilidades:	
Nombre	**Colaboración**
getId() setId() getNombre() setNombre() getTrabajadors() setTrabajadors()	 Trabajador

Tabla 2.7.15 Tarjeta CRC NO.13 TrabajadorEmpresa.

TrabajadorEmpresa	
Descripción: Guarda información acerca de los trabajadores que laboran en una empresa	
Atributos:	
Nombre	**Descripción**
id fechaEmpleo trabajador cargos empresa	Identificador de TrabajadorEmpresa
Responsabilidades:	
Nombre	**Colaboración**
getId() setId() getEmpresa() setEmpresa() getTrabajador()	 Empresa

setTrabajador() getCargos() setCargos() getFechaEmpleo() setFechaEmpleo()	Trabajador Cargo

Tabla 2.7.16 Tarjeta CRC No.14 TrabajadorEspecialidades.

TrabajadorEspecialidades	
Descripción: Guarda información acerca de los títulos del trabajador.	
Atributos:	
Nombre	**Descripción**
id especialidad trabajador anno	Identificador delTrabajadorEspecialidades
Responsabilidades:	
Nombre	**Colaboración**
getId() setId() getEspecialidad() setEspecialidad() getTrabajador() setTrabajador() getAnno() setAnno()	 Especialidad Trabajador

Tabla 2.7.17 Tarjeta CRC No.15 TrabajadorExperienciaLaboral.

TrabajadorExperienciaLaboral	
Descripción: Guarda información sobre los periodos de trabajos desempeñados en un cargo.	
Atributos:	
Nombre	**Descripción**
id trabajador	Identificador de TrabajadorExperienciaLaboral

cargoId fechaInicio fechaFin	
Responsabilidades:	
Nombre	**Colaboración**
getId() setId() getTrabajador() setTrabajador() getCargoId() setCargoId() getFechaInicio() setFechaInicio() getFechaFin() setFechaFin()	Trabajador

Tabla 2.7.18 Tarjeta CRC No.16 CargosDAO.

CargosDAO	
Descripción: Se especializa en la persistencia de la entidad Cargos en la BD.	
Atributos:	
Nombre	**Descripción**
Responsabilidades:	
Nombre	**Colaboración**
save()	Cargos
create()	Cargos
delete()	Cargos
List<Cargos> findAll()	Cargos
Cargos findById()	Cargos

Tabla 2.7.19 Tarjeta CRC No.17 EspecialidadDAO.

EspecialidadDAO	
Descripción: Se especializa en la persistencia de la entidad Especialidad en la BD.	
Atributos:	
Nombre	**Descripción**
Responsabilidades:	
Nombre	**Colaboración**
save()	Especialidad
create()	Especialidad
delete()	Especialidad
List<Cargos> findAll()	Especialidad
Cargos findById()	Especialidad

Tabla 2.7.20 Tarjeta CRC No.18 Nivel de Escolaridad.

Nivel de EscolaridadDAO	
Descripción: Se especializa en la persistencia de la entidad Nivel de Escolaridad en la BD.	
Atributos:	
Nombre	**Descripción**
Responsabilidades:	
Nombre	**Colaboración**
save()	Nivel de Escolaridad
create()	Nivel de Escolaridad
delete()	Nivel de Escolaridad
List<Cargos> findAll()	Nivel de Escolaridad
Cargos findById()	Nivel de Escolaridad

Tabla 2.7.21 Tarjeta CRC No.19 EmpresaDAO.

EmpresaDAO	
Descripción: Se especializa en la persistencia de la entidad Empresa en la BD.	
Atributos:	
Nombre	**Descripción**
Responsabilidades:	
Nombre	**Colaboración**

save()	Empresa
create()	Empresa
delete()	Empresa
List<Cargos> findAll()	Empresa
Cargos findById()	Empresa

Tabla 2.7.22 Tarjeta CRC No.20 FuncionesDAO.

FuncionesDAO	
Descripción: Se especializa en la persistencia de la entidad Funciones en la BD.	
Atributos:	
Nombre	**Descripción**
Responsabilidades:	
Nombre	**Colaboración**
save()	Funciones
create()	Funciones
delete()	Funciones
List<Cargos> findAll()	Funciones
Cargos findById()	Funciones

Tabla 2.7.23 Tarjeta CRC No.21 Condiciones TrabajoDAO.

Condiciones TrabajoDAO	
Descripción: Se especializa en la persistencia de la entidad Condiciones Trabajo en la BD.	
Atributos:	
Nombre	**Descripción**
Responsabilidades:	
Nombre	**Colaboración**
save()	Condiciones Trabajo
create()	Condiciones Trabajo
delete()	Condiciones Trabajo
List<Cargos> findAll()	Condiciones Trabajo
Cargos findById()	Condiciones Trabajo

Tabla 2.7.24 Tarjeta CRC No.22 Categoría OcupacionalDAO.

Categoría OcupacionalDAO	
Descripción: Se especializa en la persistencia de la entidad Categoría Ocupacional en la BD.	
Atributos:	
Nombre	**Descripción**
Responsabilidades:	
Nombre	**Colaboración**
save()	Categoría Ocupacional
create()	Categoría Ocupacional
delete()	Categoría Ocupacional
List<Cargos> findAll()	Categoría Ocupacional
Cargos findById()	Categoría Ocupacional

Tabla 2.7.25 Tarjeta CRC No.23 Estado CivilDAO.

Estado CivilDAO	
Descripción: Se especializa en la persistencia de la entidad Estado Civil de la BD.	
Atributos:	
Nombre	**Descripción**
Responsabilidades:	
Nombre	**Colaboración**
save()	Estado Civil
create()	Estado Civil
delete()	Estado Civil
List<Cargos> findAll()	Estado Civil
Cargos findById()	Estado Civil

Tabla 2.7.26 Tarjeta CRC No.24 Grupo SalarialDAO.

Grupo SalarialDAO	
Descripción: Se especializa en la persistencia de la entidad Grupo Salarial en la BD.	
Atributos:	
Nombre	**Descripción**
Responsabilidades:	

Nombre	Colaboración
save()	Grupo Salarial
create()	Grupo Salarial
delete()	Grupo Salarial
List<Cargos> findAll()	Grupo Salarial
Cargos findById()	Grupo Salarial

Tabla 2.7.27 Tarjeta CRC No.25 RequisitoDAO.

RequisitoDAO	
Descripción: Se especializa en la persistencia de la entidad Requisito en la BD.	
Atributos:	
Nombre	**Descripción**
Responsabilidades:	
Nombre	**Colaboración**
save()	Requisito
create()	Requisito
delete()	Requisito
List<Cargos> findAll()	Requisito
Cargos findById()	Requisito

Tabla 2.7.28 Tarjeta CRC No.26 SexoDAO.

SexoDAO	
Descripción: Se especializa en la persistencia de la entidad Sexo en la BD.	
Atributos:	
Nombre	**Descripción**
Responsabilidades:	
Nombre	**Colaboración**
save()	Sexo
create()	Sexo
delete()	Sexo
List<Cargos> findAll()	Sexo
Cargos findById()	Sexo

Tabla 2.7.29 Tarjeta CRC No.28 TrabajadorEmpresaDAO.

TrabajadorEmpresaDAO	
Descripción: Se especializa en la persistencia de la entidad TrabajadorEmpresa en la BD.	
Atributos:	
Nombre	**Descripción**
Responsabilidades:	
Nombre	**Colaboración**
save()	TrabajadorEmpresa
create()	TrabajadorEmpresa
delete()	TrabajadorEmpresa
List<Cargos> findAll()	TrabajadorEmpresa
Cargos findById()	TrabajadorEmpresa

Tabla 2.7.30 Tarjeta CRC No.29 TrabajadorEspecialidadesDAO.

TrabajadorEspecialidadesDAO	
Descripción: Se especializa en la persistencia de la entidad TrabajadorEspecialidades en la BD.	
Atributos:	
Nombre	**Descripción**
Responsabilidades:	
Nombre	**Colaboración**
save()	TrabajadorEspecialidades
create()	TrabajadorEspecialidades
delete()	TrabajadorEspecialidades
List<Cargos> findAll()	TrabajadorEspecialidades
Cargos findById()	TrabajadorEspecialidades

Tabla 2.7.31 Tarjeta CRC No.30 TrabajadorExperienciaLaboral DAO.

TrabajadorExperienciaLaboral DAO	
Descripción: Se especializa en la persistencia de la entidad TrabajadorExperienciaLaboral en la BD.	
Atributos:	
Nombre	**Descripción**

Responsabilidades:	
Nombre	**Colaboración**
save()	TrabajadorExperienciaLaboral
create()	TrabajadorExperienciaLaboral
delete()	TrabajadorExperienciaLaboral
List<Cargos> findAll()	TrabajadorExperienciaLaboral
Cargos findById()	TrabajadorExperienciaLaboral

Tabla 2.7.32 Tarjeta CRC No.31 InformesManager.

InformesManager	
Descripción: Invoca los reportes.	
Atributos:	
Nombre	**Descripción**
Responsabilidades:	
Nombre	**Colaboración**
mostrar()	

Tabla 2.7.33 Tarjeta CRC No.32 MainForm

MainForm	
Descripción: Ventana principal del sistema	
Atributos:	
Nombre	**Descripción**
mnuUsuarios mnuProcesos mnuInformes mnuGestionarUsuarios mnuSalir mnuBuscarTrabajadores mnuProcesarTrabajadores mnuInformesEstadoEmpleo mnuInformesDesempleadosEspecialidad	
Responsabilidades:	
Nombre	**Colaboración**
main()	

MainForm() initComponents() mnuGestionarUsuariosMouseClicked() mnuSalirMouseClicked() mnuBuscarTrabajadoresMouseClicked() mnuProcesarTrabajadoresMouseClicked() mnuInformesEstadoEmpleoMouseClicked() mnuInformesDesempleadosEspecialidadMouseClicked()	 GestionarUsuarios Trabajadores BuscarPerfilCargo

Tabla 2.7.34 Tarjeta CRC No.33 BuscarPerfilCargo

BuscarPerfilCargo	
Descripción: Ventana para seleccionar el perfil de cargo.	
Atributos:	
Nombre	**Descripción**
btnAceptar btnBuscarPerfil btnCancel lblPerfilesCargos lblRutaPerfil lstCargos txtRutaPerfil	
Responsabilidades:	
Nombre	**Colaboración**
BuscarPerfilCargo() initComponents() btnAceptarMouseClicked() btnBuscarPerfilMouseClicked() btnCancelMouseClicked()	 ResultadosOntologias CargarArchivo

Tabla 2.7.35 Tarjeta CRC No.34 ResultadosOntologias

ResultadosOntologias	
Descripción: Ventana para seleccionar el perfil de cargo.	
Atributos:	
Nombre	**Descripción**

btnEmplear btnDetalles btnCerrar tblResultados	
Responsabilidades:	
Nombre	**Colaboración**
ResultadosOntologias() initComponents() btnEmplearMouseClicked() btnDetallesMouseClicked() btnCerrarMouseClicked()	CompletarInformacionEmpleo

Tabla 2.7.36 Tarjeta CRC No.35 CompletarInformacionEmpleo

CompletarInformacionEmpleo	
Descripción: Ventana para introducir los daos el trabajador a emplear.	
Atributos:	
Nombre	**Descripción**
lblTrabajador lblFechaEmpleo lblEmpresa btnAceptar btnCancelar txtTrabajador txtFechaEmpleo cboEmpresa	
Responsabilidades:	
Nombre	**Colaboración**
CompletarInformacionEmpleo() initComponents() btnAceptarMouseClicked() btnCancelarMouseClicked()	TrabajadorEmpresaDAO

Tabla 2.7.37 Tarjeta CRC No.36 Trabajadores

Trabajadores	
Descripción: Ventana para buscar trabajadores.	
Atributos:	
Nombre	**Descripción**
lblNombreTrabajador lblCargo lblEmpressa txtNombreTrabajador cboCargo cboEmpresa chkEmpleado tblResultado btnBuscar	
Responsabilidades:	
Nombre	**Colaboración**
Trabajadores() initComponents() btnBuscarMouseClicked()	 TrabajadorEmpresaDAO, TrabajadorDAO

Tabla 2.7.38 Tarjeta CRC No.37 GestionarUsuarios

GestionarUsuarios	
Descripción: Ventana para buscar trabajadores.	
Atributos:	
Nombre	**Descripción**
tblUsuarios btnAdicionar btnModificar btnEliminar btnCerrar	
Responsabilidades:	
Nombre	**Colaboración**
GestionarUsuarios()	

initComponents() btnAdicionarMouseClicked() btnModificarMouseClicked() btnEliminarMouseClicked() btnCerrarMouseClicked()	ModificarUsuario UsuariosDAO

Tabla 2.7.39 Tarjeta CRC No.38 ModificarUsuario

ModificarUsuario	
Descripción: Ventana para captar los datos del usuario.	
Atributos:	
Nombre	**Descripción**
lblNombre lblContraseña lblRepetir txtNombre txtContraseña txtRepetir btnAceptar btnCancelar	
Responsabilidades:	
Nombre	**Colaboración**
ModificarUsuario() initComponents() btnAceptarMouseClicked() btnCancelarMouseClicked()	 UsuariosDAO

Tabla 2.7.40 Tarjeta CRC No.39 Login

Login	
Descripción: Ventana para autenticar un usuario.	
Atributos:	
Nombre	**Descripción**
lblUsername lblContraseña txtUsername	

txtContraseña btnEntrar btnCancelar	
Responsabilidades:	
Nombre	**Colaboración**
Login() initComponents() btnEntrarMouseClicked() btnCancelarMouseClicked()	UsuariosDAO

Anexo 3: Interfaces de usuarios

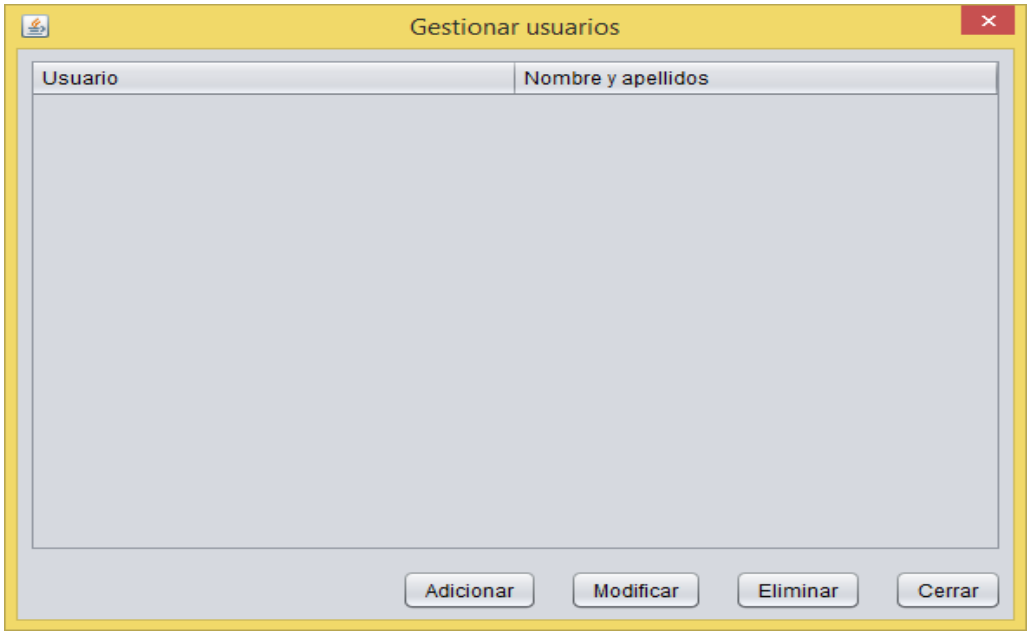


Figura 3.4.2 Interfaz de usuario. Gestionar usuarios

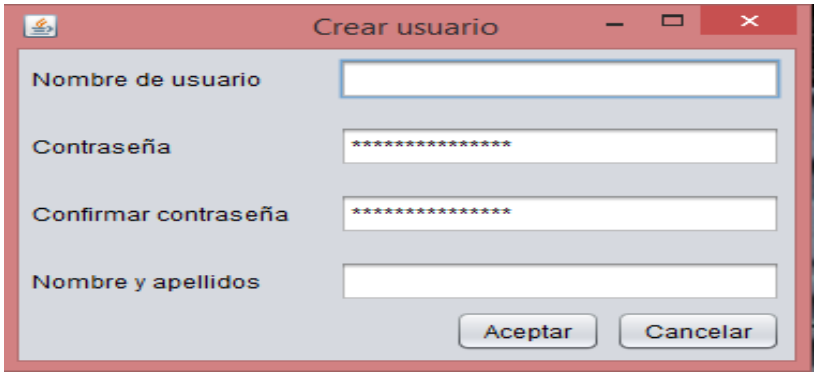


Figura 3.4.3 Interfaz de usuario: Crear usuario.

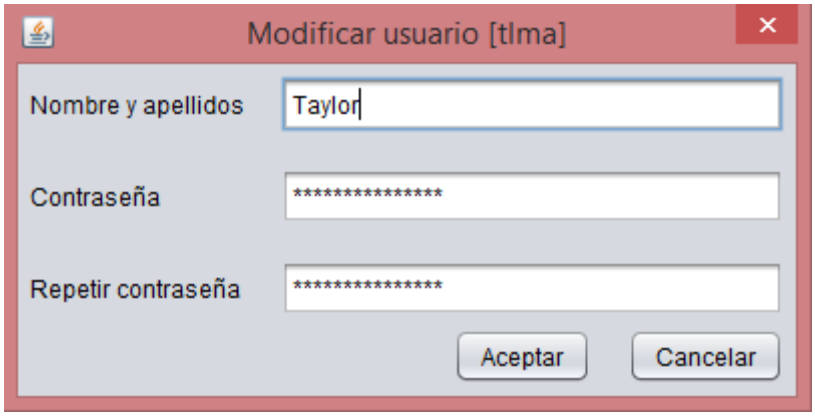


Figura 3.4.4 Interfaz de usuario: Modificar usuario.

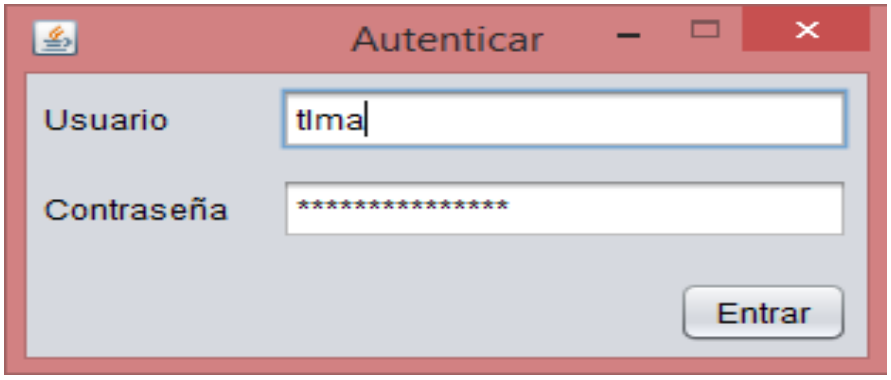


Figura 3.4.5 Interfaz de usuario: Autenticar usuario.

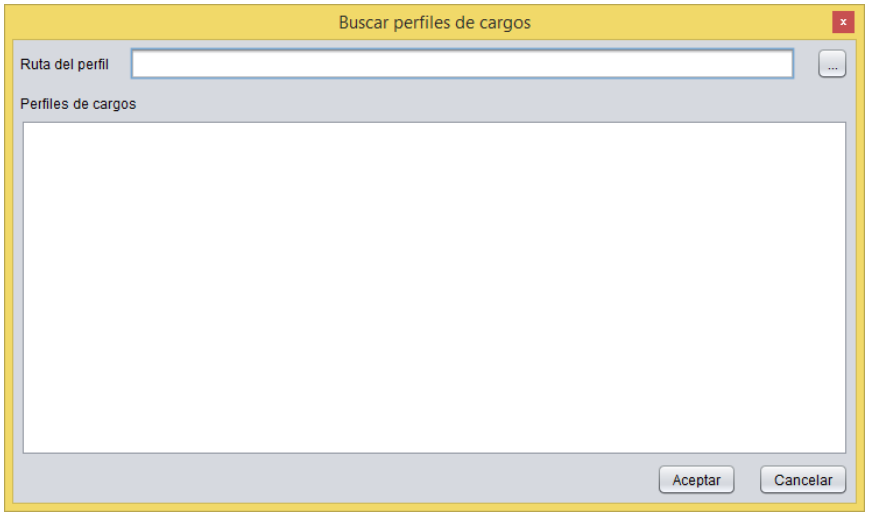


Figura 3.4.6 Interfaz de usuario: Búsqueda de perfil de cargo.

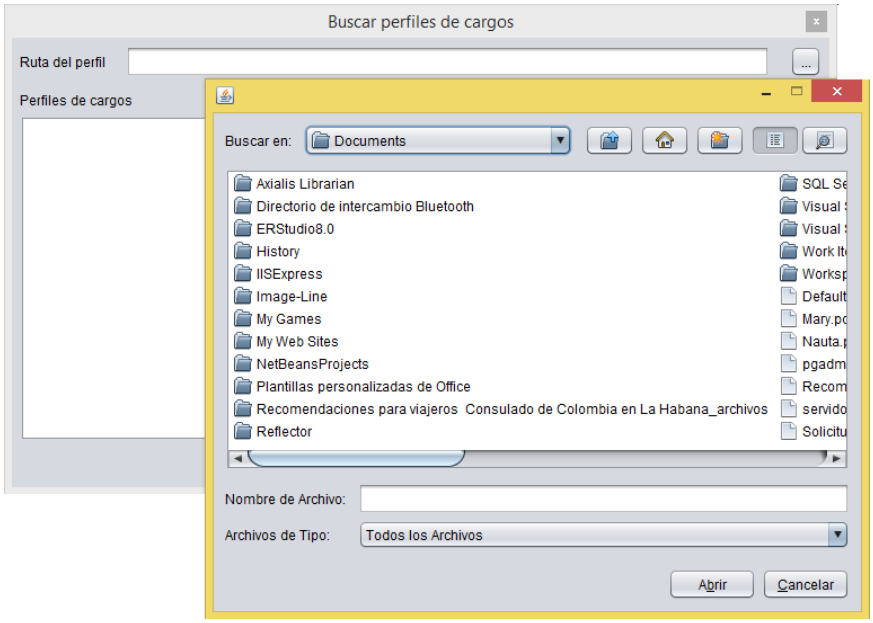


Figura 3.4.7 Interfaz de usuario: Búsqueda de perfil de cargo.

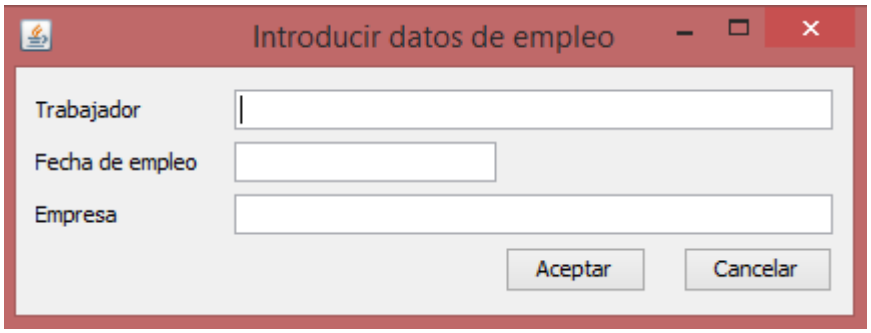


Figura 3.4.8 Interfaz de usuario: Introducir datos del trabajador empleado.

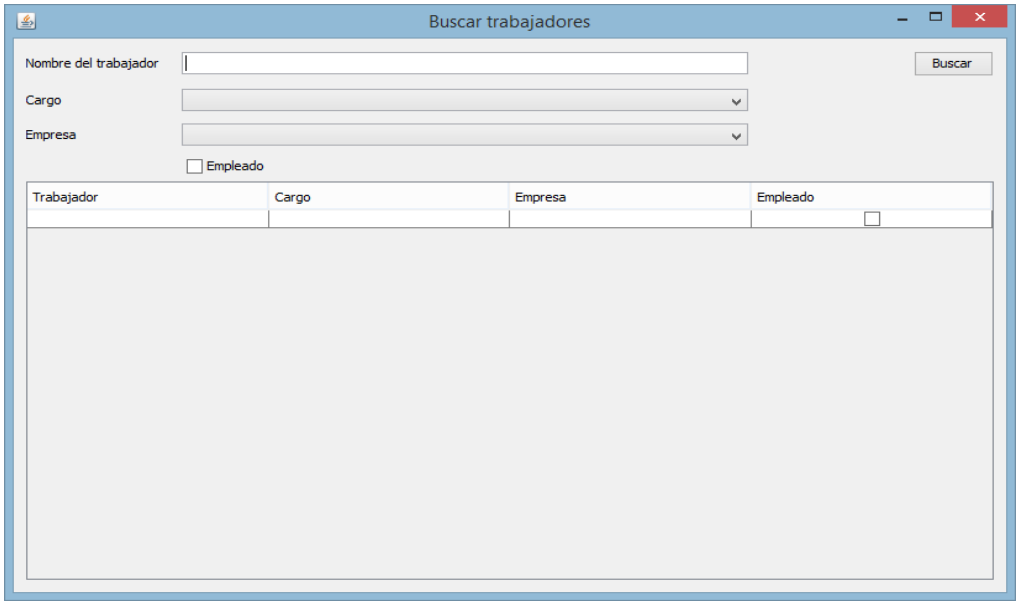


Figura 3.4.9 Interfaz de usuario: Mostrar trabajadores

Anexo 4: Tareas de programación

Tabla 3.3.4 Tarea de programación No.1 Gestionar usuario.

	Tareas de Historia de usuario
Número tarea: P1	**Número historia:** HU1
Nombre tarea: Gestionar usuario.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.50
Fecha inicio: 3 marzo 2014	**Fecha fin:** 11 marzo 2014
Responsable: Taylor Ma Pérez	
Descripción: En esta tarea se van a implementar las siguientes funcionalidades: ❖ Adicionar usuario ❖ Modificar usuario ❖ Eliminar usuario	

Tabla 3.3.5 Tarea de programación No.2 Autenticar usuario.

	Tareas de Historia de usuario
Número tarea: P2	**Número historia:** HU2
Nombre tarea: Autenticar usuario.	

Tipo de tarea: Desarrollo	**Puntos estimados:** 0.50
Fecha inicio: 12 marzo 2014	**Fecha fin:** 18 marzo 2014
Responsable: Taylor Ma Pérez	
Descripción: En esta tarea se va a implementar la funcionalidad: ❖ Autenticar usuario	

Tabla 3.3.6 Tarea de programación No.6 Introducir datos del trabajador empleado.

	Tareas de Historia de usuario
Número tarea: P6	**Número historia:** HU6
Nombre tarea: Introducir datos del trabajador empleado.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.25
Fecha inicio: 10 abril 2014	**Fecha fin:** 17 abril 2014
Responsable: Taylor Ma Pérez	
Descripción: La HU “Introducir datos del trabajador empleado” permite introducir los datos del trabajador empleado. Una vez empleado el trabajador se introducen datos como: ubicación laboral y fecha de empleo. Una vez introducido los datos se guardan los cambios y se muestra un mensaje de confirmación: “Los datos han sido guardados satisfactoriamente”.	

Tabla 3.3.7 Tarea de programación No.7 Mostrar trabajadores.

	Tareas de Historia de usuario
Número tarea: P7	**Número historia:** HU7
Nombre tarea: Mostrar trabajadores.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.50
Fecha inicio: 18 abril 2014	**Fecha fin:** 24 abril 2014
Responsable: Taylor Ma Pérez	
Descripción: En esta tarea se va a implementar la funcionalidad: ❖ Introducir datos del trabajador empleado.	

Tabla 3.3.8 Tarea de programación No.8 Graficar comportamiento del proceso de empleo.

	Tareas de Historia de usuario
Número tarea: P8	**Número historia:** HU8
Nombre tarea: Graficar comportamiento del proceso de empleo.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Fecha inicio: 25 abril 2014	**Fecha fin:** 1 mayo 2014
Responsable: Taylor Ma Pérez	
Descripción: En esta tarea se va a implementar la funcionalidad: ❖ Graficar el comportamiento del proceso de empleo.	

Tabla 3.3.9 Tarea de programación No.9 Mostar cantidad de desempleados por especialidad.

	Tareas de Historia de usuario
Número tarea: P9	**Número historia:** HU9
Nombre tarea: Mostrar cantidad de desempleados por especialidad.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Fecha inicio: 2 mayo 2014	**Fecha fin:** 15 mayo 2014
Responsable: Taylor Ma Pérez	
Descripción: En esta tarea se va a implementar la funcionalidad: ❖ Mostrar cantidad de desempleado por especialidad.	

Tabla 3.3.10 Tarea de programación No. 10 Detalles de trabajadores que fueron resultados de la búsqueda.

	Tareas de Historia de usuario
Número tarea: P10	**Número historia:** HU10
Nombre tarea: Detalles de trabajadores que fueron resultados de la búsqueda.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Fecha inicio: 16 mayo 2014	**Fecha fin:** 22 mayo 2014
Responsable: Taylor Ma Pérez	

Descripción: En esta tarea se va a implementar la funcionalidad: ❖ Detalles de trabajadores que fueron resultados de la búsqueda.

Anexo 5: Pruebas

Tabla 3.5.3.3 Prueba de aceptación No.1 Gestionar usuarios.

	Caso de prueba de aceptación
Código: (HU #1 _P1)	**Historia de usuario #1:** Gestionar usuarios.
Nombre: Prueba para verificar la gestión de usuario.	
Descripción: Gestionar de Usuarios.	
Condiciones de ejecución: El usuario “Administrador” debe estar autenticado en el sistema.	
Entrada/Pasos de ejecución: Se accede al submenú “Gestionar usuarios” de la opción “Usuarios” del menú principal. Escribe el nombre de los usuarios con sus datos para insertar, luego presiona el botón insertar. Después de haber insertado los usuarios es que se pueden modificar sus datos y eliminarlo. Para modificar un usuario se accede al submenú “Gestionar usuarios” de la opción “Usuarios” del menú principal. Se selecciona el usuario el cual se va a modificar y se modifican los datos del trabajador. Para eliminar un usuario se accede al submenú “Gestionar usuarios” de la opción “Usuarios” del menú principal. Se selecciona el usuario deseado y se da clic en el botón “Eliminar”	
Resultado Esperado: Si se insertaron los datos correctamente se podrán ver en un listado de usuarios registrados. Cuando se modifiquen datos se guardan los datos modificados. Cuando se elimina un usuario, se borra del listado de usuarios **Se muestra el mensaje de error “El usuario ya existe” en caso de que:** En caso de que se cree un usuario ya existente en la BD. **Se muestra el mensaje de error “Las contraseñas deben coincidir” en caso de que:**	

Al modificar o crear usuarios se escriban contraseñas distintas a la hora de confirmarlas.

Tabla 3.5.3.4 Prueba de aceptación No.2 Autenticar usuarios.

Caso de prueba de aceptación	
Código: (HU #2 _P2)	**Historia de usuario #2:** Autenticar usuarios.
Nombre: Prueba para verificar la autenticación de usuario.	
Descripción: Validación de entrada de los datos de los usuarios.	
Condiciones de ejecución: El usuario debe introducir su nombre de usuario y contraseña.	
Entrada/Pasos de ejecución: El usuario escribe su nombre de usuario y contraseña y luego da clic en el botón Entrar.	
Resultado Esperado: Si el usuario tiene acceso para entrar a la aplicación e inserta sus datos correctamente entrará sin problemas al Sistema. **Se muestra el mensaje de error "Este campo es requerido" en caso de que:** Se dejen campos obligatorios vacíos. **Se muestra el mensaje de error "El usuario o contraseña es incorrecto" en caso de que:** No exista el usuario a autenticar en la BD.	

Tabla 3.5.3.5 Prueba de aceptación No.6 Introducir datos del trabajador empleado.

Caso de prueba de aceptación	
Código: (HU #6 _P6)	**Historia de usuario #6:** Introducir datos del trabajador empleado.
Nombre: Prueba para verificar que se introduzcan los datos de un trabajador empleado correctamente.	
Descripción: Validación de entrada de datos de los trabajadores empleados.	
Condiciones de ejecución: El usuario debe estar autenticado para acceder a esta sección.	

Se debe de haber empleado un trabajador.
Entrada/Pasos de ejecución: El usuario entra datos como: fecha de empleo y empresa.
Resultado Esperado: Se modifican los datos del trabajador empleado. **Se muestra el mensaje de error "Este campo es requerido" en caso de que:** Se dejen campos obligatorios vacíos.

Tabla 3.5.3.6 Prueba de aceptación No.7 Mostrar trabajadores.

Caso de prueba de aceptación	
Código: (HU #7 _P7)	**Historia de usuario #7:** Mostrar trabajadores.
Nombre: Prueba para verificar que se muestren los trabajadores.	
Descripción: Validación de mostrar trabajadores.	
Condiciones de ejecución: El usuario debe estar autenticado para acceder a esta sección.	
Entrada/Pasos de ejecución: El usuario introduce el nombre del trabajador que desea buscar. Si se marca la opción empleado se muestran todos los trabajadores empleados en el sistema. Si no se marca se muestran todos los desempleados. La búsqueda puede filtrarse por nombre del trabajador, cargo y empresa	
Resultado Esperado: Mostrar trabajador(es) que son buscados. **Se produce un error en caso de que:** No se encuentre en la BD.	

Tabla 3.5.3.7 Prueba de aceptación No.8 Graficar comportamiento del proceso de empleo.

Caso de prueba de aceptación	
Código: (HU #8 _P8)	**Historia de usuario #8:** Graficar comportamiento del proceso de empleo.
Nombre: Prueba para verificar el comportamiento del proceso de empleo	

Descripción: Validación de graficar comportamiento de empleo.
Condiciones de ejecución: El usuario debe estar autenticado para acceder a esta sección. Deben de existir trabajadores empleados en la BD.
Entrada/Pasos de ejecución: Se accede al submenú “Informes” del menú principal y se le da clic a la opción “Estado de empleo”
Resultado Esperado: Se muestra la gráfica del comportamiento del proceso de empleo. **Se muestra el mensaje de error “No se puede mostrar el informe debido a un error” en caso de que:** Ocurra un error y no se pueda visualizar un informe.

Tabla 3.5.3.8 Prueba de aceptación #9 Mostrar cantidad de desempleados por especialidad.

	Caso de prueba de aceptación
Código: (HU #9 _P9)	**Historia de usuario #9:** Mostrar cantidad de desempleados por especialidad.
Nombre: Prueba para verificar la visualización de trabajadores desempleados por especialidad.	
Descripción: Validación de mostrar cantidad desempleados por especialidad.	
Condiciones de ejecución: Deben existir trabajadores desempleados en la BD.	
Entrada/Pasos de ejecución: Se accede al submenú “Informes” del menú principal y se le da clic a la opción “Trabajadores desempleados por especialidad”	
Resultado Esperado: Se muestra la cantidad de desempleados por especialidad. **Se muestra el mensaje de error “No se puede mostrar el informe debido a un error” en caso de que:** Ocurra un error y no se pueda visualizar un informe.	

Tabla 3.5.3.9 Prueba de aceptación #10 Detalles de trabajadores que fueron resultados de la búsqueda.

Caso de prueba de aceptación	
Código: (HU #10 _P10)	**Historia de usuario #10:** Detalles de trabajadores que fueron resultados de la búsqueda.
Nombre: Prueba para verificar los detalles de los trabajadores resultantes de la búsqueda.	
Descripción: Validación de detalles de trabajadores que fueron resultados de la búsqueda.	
Condiciones de ejecución: El usuario debe estar autenticado para acceder a esta sección. Se deben haber encontrado trabajadores que su perfil de competencia coincida con el perfil de cargo requerido.	
Entrada/Pasos de ejecución: Se le da clic al botón “Detalles” de la interfaz “Resultados”	
Resultado Esperado: Se muestran detales de los trabajadores encontrados como: ❖ Los requisitos que le faltaron para cumplir cien por ciento con el cargo. ❖ Los requisitos de coincidencia que tiene con el cargo. **Se muestra el mensaje de error “No se puede mostrar el informe debido a un error” en caso de que:** Ocurra un error y no se pueda visualizar un informe.	