



INSTITUTO SUPERIOR MINERO METALÚRGICO

“Dr. Antonio Núñez Jiménez”

Facultad de Metalurgia - Electromecánica

Moa, Holguín

**Trabajo de Diploma en opción por el Título de Ingeniero
Informático**

SISTEMA INFORMÁTICO PARA LA SOLUCIÓN DE PROBLEMAS DE ÁRBOL DE EXPANSIÓN MÍNIMA

Autor: Alexeis Legrá Quintero

Tutor: Lic. Isidro Manuel Corría Ramírez

Consultante: Ing. Yoel Orlando Perdomo Fernández

Moa, Holguín

Julio del 2009

DECLARACIÓN DE AUTORÍA

Declaro que soy el único autor de este trabajo y autorizo al Centro ISMM para que hagan el uso que estimen pertinente con este trabajo.

Para que así conste firmo la presente a los _____ días del mes de _____ del 2009.

Alexeis Legrá Quintero

Nombre completo del primer autor

Lic. Isidro Manuel Corría Ramírez

Nombre completo del primer tutor

OPINIÓN DEL USUARIO DEL TRABAJO DE DIPLOMA

El Trabajo de Diploma, titulado **Sistema informático para la solución de problemas de árbol de expansión mínima**, fue realizado en nuestra entidad **ISMM**. Se considera que, en correspondencia con los objetivos trazados, el trabajo realizado le satisface:

- ☐ Totalmente
- ☐ Parcialmente en un _____ %

Los resultados de este Trabajo de Diploma le reportan a esta entidad los beneficios siguientes (cuantificar):

Como resultado de la implantación de este trabajo se reporta un efecto económico que asciende a <valor> MN y/o <valor> CUC. (Este valor debe ser REAL, no indica lo que se reportará, sino lo que reporta a la entidad. Puede desglosarse por conceptos, tales como: cuanto cuesta un software análogo en el mercado internacional, valor de los materiales que se ahorran por la existencia del software, valor anual del (de los) salario(s) equivalente al tiempo que se ahorra por la existencia del software).

Y para que así conste, se firma la presente a los _____ días del mes de _____ del año _____

Nombre del representante de la entidad

Cargo

Firma

Cuño

OPINIÓN DEL TUTOR DEL TRABAJO DE DIPLOMA

Título: Sistema informático para la solución de problemas de árbol de expansión mínima

Autor: Alexeis Legrá Quintero

El tutor del presente Trabajo de Diploma considera que durante su ejecución el estudiante mostró las cualidades que a continuación se detallan.

<Aquí el tutor debe expresar cualitativamente su opinión y medir (usando la escala: muy alta, alta, adecuada) entre otras las cualidades siguientes:

- Independencia
- Originalidad
- Creatividad
- Laboriosidad
- Responsabilidad>

<Además, debe evaluar la calidad científico-técnica del trabajo realizado (resultados y documento) y expresar su opinión sobre el valor de los resultados obtenidos (aplicación y beneficios) >

Por todo lo anteriormente expresado considero que el estudiante está apto para ejercer como Ingeniero Informático; y propongo que se le otorgue al Trabajo de Diploma la calificación de <nota 2-Desaprobado, 3-Aprobado, 4-Bien, 5-Excelente>. <Además, si considera que los resultados poseen valor para ser publicados, debe expresarlo también>

(Si procede)

Nombre completo del primer tutor

<Grado científico, Categoría docente y/o
investigativa>

Nombre completo del segundo tutor

<Grado científico, Categoría docente y/o
investigativa>

Fecha: _____

Agradecimientos

A mi familia, y muy especial a mi abuela, por tanto amor y confianza que han depositado en mí.

A mi novia, por todo su amor y estar a mi lado en los momentos buenos y malos.

A mis amigos y compañeros de aula: en especial a Lisandro M, Ezequiel, Roberto, Yoel, Paucides, Osmanis, Scull, Yisel, María, por darme su apoyo incondicional.

A mi tutor, Lic. Isidro Manuel Corría Ramírez, por su ayuda y ser guía en todo momento, a mis profesores del departamento.

A todos aquellos que de una forma u otra han contribuido con la realización de este trabajo y en mi formación como futuro profesional.

A todos, gracias...

Alexeis Legrá Quintero.

Dedicatoria

Dedico este trabajo a todas aquellas personas portadoras de los más nobles sentimientos: el amor, la amistad, la confianza, que me mostraron siempre el camino correcto por el cual he de transitar.

En especial:

A mi abuela, símbolos de inspiración de todos mis proyectos.

A mis padres, por darme la oportunidad de haber de vivir en este mundo.

A mis tíos, faro de mis pasos, y oportuno consejo.

A mi hermano, por inspirarme a ser un ejemplo y guía para él.

A mi novia, por alentarme y estar a mi lado todo el tiempo.

A mis suegros, por acogerme como un hijo más en la familia.

En general a todas aquellas personas que no menciono, no por ser menos importantes sino porque no alcanzarían estas pocas líneas para mencionarlos.

*If people do not believe that mathematics is simple, it is only
because they do not realize how complicate life is.*

John Von Neuman

Resumen

A pesar de las favorables condiciones que nos brindan las nuevas tecnologías existen elementos que dificultan un mejor desempeño del profesional a la hora de la toma de decisiones mediante la solución de problemas de árbol de expansión mínima (PAEM).

Con el desarrollo de esta investigación se propone elaborar una herramienta informática que contribuyera a elevar la efectividad de solucionar problemas de árbol de expansión mínima en el Instituto Superior Minero Metalúrgico “Dr. Antonio Núñez Jiménez” (ISMM), para una mejor toma de decisiones.

Para la realización de la investigación se realizó una revisión bibliográfica sobre aplicaciones y herramientas para su construcción. En el presente documento se recoge un resumen del estudio bibliográfico realizado, se presenta la metodología de Ingeniería de Software que se siguió para la construcción del software la cuál se propone como solución de la problemática encontrada. En el trabajo se realiza además un estudio de factibilidad y una valoración de sostenibilidad del producto informático que se obtiene como resultado de la investigación.

Abstract

In spite of the favorable conditions that the new technologies offer us exist elements that they make difficult a professional's best performance to the hour of the decision making through the solution of problems of tree of minimal expansion (PAEM).

With the development of this research it's proposed elaborating an information-technology tool that it contribute to elevating the effectiveness to solve problems of tree of minimal expansion in the Instituto Superior Minero Metalúrgico Dr. Antonio Núñez Jiménez (ISMM) for a best take of decisions.

A bibliographic revision on applications and tools for his construction was accomplished for the realization of research. A summary of bibliographic study accomplished is picked up in the present document; Software Engineering methodology that was kept for the software's construction is shown, which it's proposed like solution of problems found. Besides, feasibility's study and sustainability's evaluation of the information-technology product that is obtained as a result of the research are realized.

Índice

INTRODUCCIÓN	1
CAPÍTULO 1 FUNDAMENTACIÓN	8
1.1 INTRODUCCIÓN	8
1.2 ESTADO DEL ARTE.....	8
1.2.1 <i>Conceptos Fundamentales</i>	8
1.3 SISTEMAS AUTOMATIZADOS EXISTENTES VINCULADOS AL CAMPO DE ACCIÓN	16
1.4 TENDENCIAS Y TECNOLOGÍAS ACTUALES	17
1.4.1 <i>Lenguajes de programación para el desarrollo de aplicaciones</i>	17
1.4.2 <i>Fundamentación de la selección de lenguaje a utilizar</i>	22
1.4.3 <i>Metodologías para el desarrollo de Sistemas Informáticos</i>	23
1.5 CONCLUSIONES	28
CAPÍTULO 2 MODELO DEL DOMINIO	29
2.1 INTRODUCCIÓN	29
2.2 DEFINICIÓN DE LAS ENTIDADES Y LOS CONCEPTOS PRINCIPALES	29
2.3 REPRESENTACIÓN DEL MODELO DEL DOMINIO.....	30
2.4 REQUERIMIENTOS FUNCIONALES Y NO FUNCIONALES DEL SISTEMA.	30
2.4.1 <i>Requisitos Funcionales</i>	30
2.4.2 <i>Requisitos no Funcionales</i>	31
2.5 CONCLUSIONES.	32
CAPÍTULO 3 DISEÑO E IMPLEMENTACIÓN DEL SISTEMA	33
3.1 INTRODUCCIÓN	33
3.2 ACTORES DEL SISTEMA A AUTOMATIZAR.....	33
3.3 DIAGRAMA DE CASOS DE USO DEL SISTEMA A AUTOMATIZAR	33
3.4 DESCRIPCIÓN DE LOS CASOS DE USO.....	34
3.5 DIAGRAMA DE CLASES DEL DISEÑO	35
3.5.1 <i>Diagrama de Clases General</i>	35
3.5.2 <i>Descripción General de las Clases</i>	37
3.6 PRINCIPIOS DE DISEÑO	38
3.6.1 <i>Interfaz de usuario</i>	38
3.6.2 <i>Formato de salida de los reportes</i>	43
3.6.3 <i>Ayuda</i>	43
3.7 DIAGRAMA DE SECUENCIA	44
3.8 DIAGRAMA DE DESPLIEGUE	48

3.9	DIAGRAMA DE COMPONENTES	48
3.10	CONCLUSIONES	49
CAPÍTULO 4 ESTUDIO DE FACTIBILIDAD.....		50
4.1	INTRODUCCIÓN.....	50
4.2	EFFECTOS ECONÓMICOS.....	50
4.3	BENEFICIOS Y COSTOS INTANGIBLES EN EL PROYECTO	52
4.4	FICHA DE COSTO	53
4.5	VALORACIÓN DE SOSTENIBILIDAD DEL PRODUCTO SEGÚN SU IMPACTO ADMINISTRATIVO, SOCIO- HUMANISTA, AMBIENTAL Y TECNOLÓGICO.	56
4.6	CONCLUSIONES	59
CONCLUSIONES		60
RECOMENDACIONES.....		61
REFERENCIAS BIBLIOGRÁFICAS.....		62
BIBLIOGRAFÍA		63
GLOSARIO DE TÉRMINOS.....		65
ANEXO 1 DESCRIPCIÓN DE LOS CASOS DE USO DEL SISTEMA.....		I
ANEXO 2 DESCRIPCIÓN GENERAL DE LAS CLASES.....		VI

Índice de tablas

TABLA 1. ENTIDADES Y CONCEPTOS DEL DOMINIO.....	29
TABLA 2. DEFINICIÓN DE ACTORES DEL SISTEMA A AUTOMATIZAR.....	33
TABLA 3. DESCRIPCIÓN DEL CASO DE USO (CONSTRUIR AEM).....	34
TABLA 4. DESCRIPCIÓN DE LA CLASE “GRAPH”	37
TABLA 5. DESCRIPCIÓN DEL CASO DE USO “INSERTAR NODO”.....	I
TABLA 6. DESCRIPCIÓN DEL CASO DE USO “EDITAR NODO”.....	I
TABLA 7. DESCRIPCIÓN DEL CASO DE USO “ELIMINAR NODO”	II
TABLA 8. DESCRIPCIÓN DEL CASO DE USO “INSERTAR ARCO”.....	II
TABLA 9. DESCRIPCIÓN DEL CASO DE USO “EDITAR ARCO”.....	II
TABLA 10. DESCRIPCIÓN DEL CASO DE USO “CAMBIAR PESO”	III
TABLA 11. DESCRIPCIÓN DEL CASO DE USO “CREAR GRAFO”	IV
TABLA 12. DESCRIPCIÓN DEL CASO DE USO “MOVER NODO”	IV
TABLA 13. DESCRIPCIÓN DEL CASO DE USO “MOSTRAR SOLUCIÓN”.....	IV
TABLA 14. DESCRIPCIÓN DEL CASO DE USO “GUARDAR SOLUCIÓN”.....	V
TABLA 15. DESCRIPCIÓN DE LA CLASE “DRAW PANEL”	VI
TABLA 16. DESCRIPCIÓN DE LA CLASE “LABEL”	VI
TABLA 17. DESCRIPCIÓN DE LA CLASE “MAINWINDOW”	VII
TABLA 18. DESCRIPCIÓN DE LA CLASE “VISUALGRAPH”	VIII

Índice de Figura

FIGURA 1 DIAGRAMA DEL MODELO DEL DOMINIO.....	30
FIGURA 2. DIAGRAMA DE CASOS DE USO DEL SISTEMA.	33
FIGURA 1. DIAGRAMA DE CLASES DEL SISTEMA.....	35
FIGURA 4. INTERFAZ DE USUARIO “INSERTAR NODO”	38
FIGURA 5. INTERFAZ DE USUARIO “EDITAR NODO”	38
FIGURA 6. INTERFAZ DE USUARIO “ELIMINAR NODO”	39
FIGURA 7. INTERFAZ DE USUARIO “INSERTAR ARCO”	39
FIGURA 8. INTERFAZ DE USUARIO “EDITAR ARCO”	40
FIGURA 9. INTERFAZ DE USUARIO “CAMBIAR PESO”.....	40
FIGURA 10. INTERFAZ DE USUARIO “CREAR GRAFO”.....	41
FIGURA 11. INTERFAZ DE USUARIO “MOVER NODO”	41
FIGURA 12. INTERFAZ DE USUARIO “CONSTRUIR AEM”	42
FIGURA 13. INTERFAZ DE USUARIO “GUARDAR SOLUCIÓN”	42
FIGURA 14. INTERFAZ DE USUARIO “MOSTRAR AEM”.....	43
FIGURA 15. DIAGRAMA DE SECUENCIA “INSERTAR & EDITAR NODO”.	44
FIGURA 17. DIAGRAMA DE SECUENCIA “INSERTAR & EDITAR ARCO”.	45
FIGURA 18. DIAGRAMA DE SECUENCIA “CAMBIAR PESO”.	45
FIGURA 19. DIAGRAMA DE SECUENCIA “CREAR GRAFO”.....	46
FIGURA 20. DIAGRAMA DE SECUENCIA “MOVER NODO”.....	46
FIGURA 21. DIAGRAMA DE SECUENCIA “CONSTRUIR AEM & MOSTRAR SOLUCIÓN”. ..	47
FIGURA 22. DIAGRAMA DE SECUENCIA “GUARDAR SOLUCIÓN”.....	47
FIGURA 23. DIAGRAMA DE DESPLIEGUE.....	48
FIGURA 24. DIAGRAMA DE COMPONENTE.....	48
FIGURA 25. PUNTO DE EQUILIBRIO.....	56



Introducción

Desde los orígenes de la humanidad el hombre en su afán por la supervivencia se ha visto inmerso en la necesidad de recurrir a la toma de decisiones con el fin de adaptarse a las diferentes condiciones en que se ha desarrollado.

La situación económica y social que caracteriza a la sociedad moderna genera profundos cambios en las organizaciones, las cuales se preparan para ser más flexibles y establecen estrategias con el objetivo de adaptarse al entorno altamente turbulento en el que desarrollan sus acciones. Ante ambientes tan poco estables y la imposibilidad de actuar a ciegas, los miembros de las organizaciones y, en particular, sus altas gerencias necesitan manipular grandes volúmenes de información para cumplir con sus funciones esenciales. Deben implementarse entonces, prácticas administrativas dirigidas a garantizar el éxito organizacional y entre ellas, la toma de decisiones, soportada en el análisis de información. Para lo que ha venido desarrollando mecanismos que avalen o justifiquen dichas decisiones.

Con respecto al concepto "toma de decisiones", Schein plantea: "la toma de decisiones es el proceso de identificación de un problema u oportunidad y la selección de una alternativa de acción entre varias existentes, es una actividad diligente clave en todo tipo de organización."[1].

La teoría de decisiones plantea que cuando se está en la tarea de seleccionar la mejor alternativa se caerá en una de cuatro categorías generales, dependiendo de las habilidades para predecir la consecuencia de cada alternativa; derivándose de estas diferentes formas de toma de decisión como son:

Bajo certidumbre: si se puede predecir con certeza las consecuencias de cada alternativa.

Bajo riesgo: esta categoría incluye aquellas decisiones para las que las consecuencias de una acción dada dependen de algún evento probabilista.

Bajo incertidumbre: es parecido a la toma de decisiones bajo riesgos con la diferencia de que no se tiene conocimientos de las probabilidades de los eventos futuros, es decir, de cuán posible sean las diferentes consecuencias.



Bajo conflicto: aquí se tiene aquellos casos de toma de decisiones bajo incertidumbre en los que hay un oponente, las probabilidades de los eventos no solo se conocen; están influenciadas por un oponente cuya meta es vencer.

Para dar solución a este problema surge la Investigación Operativa o Investigación de Operaciones (IO), una moderna disciplina científica que se caracteriza por la aplicación de teoría, métodos y técnicas especiales, para buscar la solución de problemas de administración, organización y control que se producen en los diversos sistemas que existen en la naturaleza. Actualmente incluye gran cantidad de ramas como la Programación Lineal, Programación No Lineal, Programación Dinámica, Simulación, Teoría de Colas, Teoría de Inventarios, Teoría de Grafos, etc.

Aunque su nacimiento como ciencia se establece durante la Segunda Guerra Mundial y debe su nombre a las operaciones militares, los verdaderos orígenes de la Investigación Operativa se remontan mucho más atrás en el tiempo, hasta el siglo XVII. Sin embargo su auge es debido, en su mayor parte, al gran desarrollo de la computación, gracias a la cual es posible resolver problemas en la práctica y obtener soluciones que de otra forma conllevarían un enorme tiempo de cálculo. Debido al gran éxito de la Investigación Operativa en el campo militar, esta se extendió a otros campos tales como la industria, física, informática, economía, estadística y probabilidad, ecología, educación, servicio social, etc., siendo hoy en día utilizada prácticamente en todas las áreas.

El objetivo más importante de la aplicación de la Investigación Operativa es apoyar en la “toma óptima de decisiones” en los sistemas y en la planificación de sus actividades.

El enfoque fundamental de la Investigación Operativa es el enfoque de sistemas, por el cual, a diferencia del enfoque tradicional, se estudia el comportamiento de todo un conjunto de partes o sub-sistemas que interaccionan entre sí, se identifica el problema y se analizan sus repercusiones, buscándose soluciones integrales que benefician al sistema como un todo.

Una de las clase de problemas que a los cuales la IO da solución, son los problemas de (redes) y dentro de estos el problema del árbol de expansión



mínima (PAEM), problema común de optimización combinatoria. Formulado inicialmente por Boruvka en 1926, quien se dice tuvo que aprender de éste durante la electrificación del sur de Moravia donde él proporcionó una solución para hallar la distribución más económica a través de una red de una línea de energía. Desde entonces su formulación ha sido aplicada en numerosos problemas combinatorios tales como: problemas de transporte, diseño de redes de telecomunicaciones, sistemas distribuidos y otros. Al mismo tiempo algunos algoritmos de tiempo polinomial fueron desarrollados para su resolución por Prim, Kruskal, Dijkstra y Sollin.

Podemos afirmar que la utilización de sistemas informáticos de análisis de datos ha supuesto una revolución en la toma de decisiones por varias razones:

1. Se simplifica enormemente los procedimientos matemáticos y como consecuencia se emplean técnicas que antes eran muy difícil aplicar.
2. Se pasa del cálculo manual tradicional al cálculo por ordenador lo que implica una mayor rapidez en la obtención de los resultados, mayor fiabilidad, mejor manejo de estos procedimientos y una gran facilidad de aplicación.
3. Propicia una nueva forma de enseñanza del análisis de datos sin necesidad de una gran formación matemática para efectuar operaciones que en ocasiones son muy complejas.

A pesar de las favorables condiciones que nos brindan las nuevas tecnologías existen elementos que dificultan un mejor desempeño del estudiantado y el profesional en el Instituto Superior Minero Metalúrgico Dr. Antonio Núñez Jiménez (ISMM) ante el hecho de la toma de decisiones mediante la solución de problemas de árbol de expansión mínima (PAEM).

Por ejemplo situaciones como estas se evidencian en los estudiantes que reciben la asignatura Investigación de Operaciones II en las especialidades de Informática y Contabilidad, los cuales cuentan con softwares que facilitan el proceso de la toma de decisiones como el Matemática, Microsoft Office Excel entre otros, pero a medida que estos se desarrollan aumentan la demanda de



hardware: memoria(RAM), procesador y capacidad de almacenamiento , necesitando computadoras con condiciones mas exigentes.

Debido a que son de propósito general y no exactamente para resolver PAEM, carecen de integridad respecto a los conocimientos para el eficiente análisis e interpretación de los resultados, obliga al usuario a un profundo estudio de los diferentes manuales de usuarios de los mismo, esto es tiempo de estudio que el estudiante deja de emplear en el contenido de la asignatura y tiempo que pierde en su corto periodo de tiempo de maquina, el cual cada vez se hace más limitado por la difícil situación que atraviesa el ISMM donde el por ciento de máquinas por estudiantes es muy bajo, para emplearlo en el funcionamiento de los softwares antes mencionados.

Esta misma situación la comparten los administrativos que se enfrenten a solucionar problemas de este tipo, como el centro de red del ISMM a la hora de diseñar la conectividad de una red de computadoras optimizando el cableado, a los cuales el uso de estos software se les torna más difícil debido a que estos no son especialistas en la materia ,y consumirían mucho tiempo en aprender a trabajar con los mismos ,además estas herramientas están ligadas al sistema operativo propietario Microsoft Windows, no siendo compatibles con otros sistemas operativos que son usados en el centro de red del ISMM, lo cual genera problemas para su uso. Por estas razones el Ministerio de Educación Superior (MES) se a trazado una estrategia a mediano plazo para migrar a plataformas patentizadas baja licencias GNU.

El análisis de estas situaciones llevó a plantear el siguiente **problema científico**:

¿Cómo elevar la efectividad de la toma de decisiones mediante la solución de PAEM en el Instituto Superior Minero Metalúrgico “Dr. Antonio Núñez Jiménez (ISMM)”?

A partir del problema se define como **objeto de investigación**: la toma de decisiones mediante los problemas de árbol de expansión mínima.



Para dar solución al problema se trazó como **objetivo**: elaborar una herramienta informática que contribuya a elevar la efectividad de la toma de decisiones mediante la solución de PAEM en el ISMM.

Delimitándose el **campo de acción**: automatización del proceso de solución de PAEM en el ISMM.

Para guiar la investigación, se concibió la **hipótesis**: Una herramienta informática de apoyo al proceso de solución de problemas de árbol de expansión mínima, que utilice pocos recursos computacionales, sea multiplataforma, brinde al usuario seguridad y rapidez en los tiempos de respuesta, y libre de costo alguno, elevaría la efectividad en el proceso de toma de decisiones en el Instituto Superior Minero Metalúrgico Dr. Antonio Núñez Jiménez.

Para dar cumplimiento al objetivo se propuso cumplir con las siguientes **tareas de investigación**:

1. Elaborar los fundamentos teóricos sobre la toma de decisiones en los problemas de árbol de expansión mínima.
2. Efectuar una selección entre las principales técnicas existentes que solucionen el problema.
3. Diseñar e implementar una herramienta informática que contribuya a elevar la efectividad de la toma de decisiones mediante la solución de PAEM.
4. Establecer pruebas al software con estudiantes y profesionales.
5. Efectuar el estudio de Factibilidad.

Para cumplimentar estas tareas se han empleado **métodos teóricos y empíricos** de investigación científica.

1. Entre los **métodos empíricos** utilizados se encuentran la entrevista, el análisis de documentos y la encuesta.
 - La **entrevista** posibilitó conocer más a fondo las necesidades de los profesionales y estudiantes en la solución de problemas de árbol de expansión mínima además de permitir determinar los principales requerimientos del sistema.



- El **análisis de documentos** se empleó para la recopilación de la información y permitió conocer cómo funcionan actualmente los procesos de solución de problemas de árbol de expansión mínima.
2. Los **métodos teóricos** proporcionaron calidad en la investigación. Se usaron:
- El **análisis y síntesis** para la recopilación y el procesamiento de la información obtenida con los métodos empíricos y arribar a las conclusiones de la investigación.
 - El **hipotético deductivo** en la elaboración de la hipótesis y para su verificación.
 - La **modelación** para realizar una representación de la realidad, lograr detectar problemas en la forma actual de procesar la información y encontrar las funcionalidades que debe de tener el sistema que se propone, que lo harán más completo y le brindarán satisfacción al usuario con un producto de mayor calidad.
 - El **enfoque sistémico** para la modelación del software

El presente trabajo consta de introducción, cuatro capítulos, conclusiones, recomendaciones, referencias bibliográficas, bibliografía, glosario de términos y anexos:

En el Capítulo 1, Fundamentación, se ofrece una breve descripción del objeto de estudio, el flujo de los procesos en los que interviene el mismo y un análisis crítico al respecto. Se brinda además una panorámica de los sistemas automatizados existentes vinculados al campo de acción y las tendencias y tecnologías actuales así como un análisis crítico de las fuentes utilizadas. Se efectúa una selección entre las principales técnicas existentes que solucionen el problema.

En el Capítulo 2, Modelo del Dominio, se explica el bajo nivel de estructuración del negocio y la necesidad de utilizar un modelo de domino para mostrar la dinámica del sistema, se definen las entidades y conceptos principales y las reglas de negocio a considerar así como la representación del modelo de dominio.



En el Capítulo 3, Diseño e Implementación del Sistema, se describen en detalles los flujos de trabajos relacionados a estas etapas de diseño e implementación de la metodología utilizada, RUP.

En el Capítulo 4, Estudio de Factibilidad, se presenta el estudio de factibilidad de este proyecto, se utilizará la Metodología Costo Efectividad (Beneficio), la cual plantea la conveniencia de la ejecución del proyecto, además se presenta la Valoración de Sostenibilidad del producto según su impacto administrativo, socio – humanista, ambiental y tecnológico.

Para concluir se muestran las Conclusiones a las que se arribaron, las Recomendaciones que se proponen, las Referencias Bibliográficas, la Bibliografía utilizada, el Glosario de Términos y Anexos con información necesaria sobre el trabajo.



Capítulo 1 Fundamentación

1.1 Introducción

En el presente capítulo se brinda una visión general de los aspectos relacionados con la solución de problemas de árbol de expansión mínima.

Se abordan las pautas específicas que constituyen los fundamentos teóricos sobre los que se apoya nuestra propuesta, además se realiza un análisis de las herramientas más comunes para el desarrollo de aplicaciones y se fundamenta la elección del lenguaje seleccionado, así como la metodología a utilizar.

Se hace una valoración de los sistemas automatizados existentes vinculados al campo de acción.

1.2 Estado del Arte.

1.2.1 Conceptos Fundamentales

El problema de árbol de expansión mínima es uno de los temas que estudia la Investigación de Operaciones, el cual consta con muchas aplicaciones prácticas importantes como hallar soluciones en diversas áreas de (transporte, diseño de redes de telecomunicaciones, sistemas distribuidos y otros), es un valioso instrumento aplicable a cualquier problema de investigación económica, puesto que brinda al investigador o grupo de investigadores una determinada estructura, mediante la cual puede enmarcar su trabajo. Este problema de construir un árbol de expansión mínima (AEM) a partir de un grafo fue formulado en 1926 por Boruvka como se menciono anteriormente.

Un grafo $G = (V, E)$, esta formado por un grupo de vértices, V , y un conjunto de aristas, E . Cada arista es un par (v, w) , donde $v, w \in V$, y. En ocasiones los vértices se denominan nodos, y las aristas arcos. Si los pares que definen las aristas están ordenados, se dice que el grafo es dirigido. Los grafos dirigidos en ocasiones se le denominan diégrafos. En un diégrafo, el vértice w es adyacente al vértice v si y solo si $(v, w) \in E$. Algunas veces las aristas tienen una tercera componente, denominada peso o coste. Un camino en un grafo es un conjunto de vértices conectados entre si por aristas. La longitud de dicho camino es el número de aristas en el camino, es decir $N-1$. Esta longitud se le denomina



longitud del camino sin pesos. La longitud del camino con pesos es la suma de los costes de las aristas en el camino.

Un grafo con n nodos podría tener tantos arcos como $n! / [(n-2)! 2!] = n(n-1)/2$. Si están dirigidos, este número pudiese ser doble. Este enorme número de arcos posibles es una de las razones del porque existen soluciones de algoritmos especiales para problemas de redes particulares.

Un árbol de expansión mínima se construye a partir de un grafo no dirigido, siendo este el árbol con menor coste obtenido de dicho grafo, en el cual no necesariamente existe un solo AEM, pueden existir varios árboles con el mismo coste, esto sucede por los empates encontrados al construir un AEM, donde se puede tomar cualquiera de las vías que lleven hacia una solución óptima.

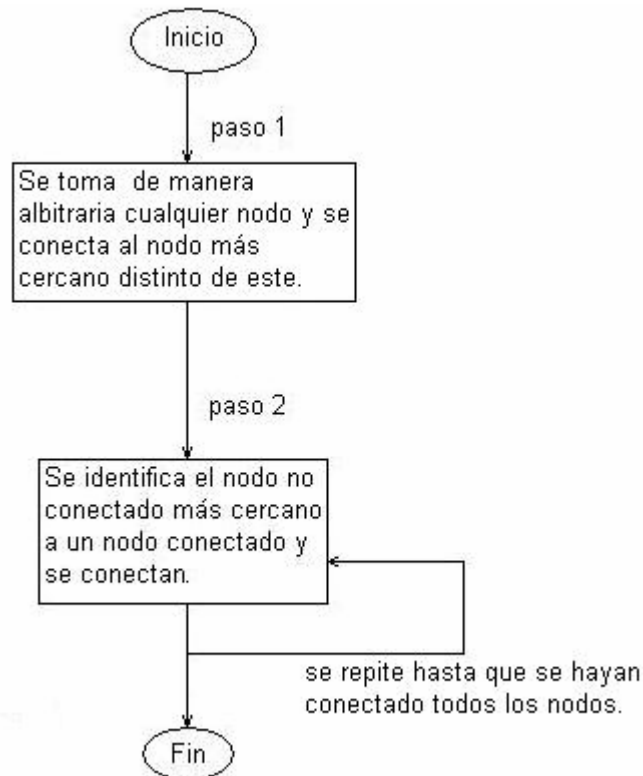
Un árbol de expansión mínima es una red no dirigida y conexa, con n nodos y $(n-1)$ ligaduras, en la que la información dada incluye alguna medida de longitud positiva (distancia, costo, tiempo, etc.) asociada con cada ligadura y cada nodo marca una posición determinada.

Los algoritmos mas conocidos desarrollados para dar solución al PAEM son Prim, Kruskal, Dijkstra y variantes de estos, a pesar del tiempo que ha transcurrido desde sus orígenes en la actualidad mantienen su vigencia. El costo computacional de estos algoritmos es polinomial.

Debido al costo computacional se han venido empleando Algoritmos Evolutivos para su solución. Desde varias décadas atrás, se han tratado de encontrar algoritmos eficientes para el PAEM, cuya finalidad es encontrar un árbol de expansión mínima que no viole ciertas restricciones de capacidad definidas.

El PAEM cae dentro del dominio de los problemas de optimización, en los cuales se trata de encontrar la mejor de varias soluciones posibles -punto óptimo- dentro del espacio de búsqueda, siendo necesario conocer como medir y saber cuando una solución es buena o mala.

Para la solución de problemas de árbol de expansión mínima se tiene en cuenta el siguiente algoritmo, que cuenta con las siguientes fases:



Empates: los empates para el nodo más cercano distinto (paso 1) o para el nodo no conectado mas cercano (paso 2), se puede romper en forma arbitraria y el algoritmo todavía debe llevar a una solución óptima .No obstante, estos empates son señal de que pueden existir (pero no necesariamente) soluciones óptimas múltiples .Todas estas soluciones se pueden identificar si se buscan las demás formas de romper los empates hasta el final.

Este algoritmo a pesar de ser un algoritmo ávido encuentra una solución óptima. Para el desarrollo del presente trabajo de los algoritmos antes mencionados selecciono el algoritmo de Prim, debido que Dijkstra y Kruscal fueron confeccionados específicamente para resolver el problema del camino mas corto y las modificaciones para que estos resuelvan el PAEM hace que tengan el mismo costo que Prim $O(n) = n^3$

Analizaremos un algoritmo para determinar un árbol de expansión mínima (AEM) conocido como algoritmo de Prim. Este construye un árbol agregando aristas de



manera iterativa hasta obtener un AEM. En cada iteración agregamos una arista de peso mínimo que no forme un ciclo en el árbol en cuestión.

Este algoritmo determina un AEM en una grafica conexa con pesos.

Entrada: Una grafica conexa con pesos, con vértices $1, \dots, n$ y vértice inicial s . Si (i, j) es una arista, $w(i, j)$ es igual al peso de (i, j) , si (i, j) no es una arista, entonces $w(i, j)$ es igual a ∞ (un valor mayor que cualquier peso real)

Salida: El conjunto de aristas E es un AEM.

Demostración de la validez del algoritmo

Análisis de la complejidad

Primera aproximación

En el paso 2, si S tiene k vértices hay $n-k$ vértices en $V-S$. Por tanto, necesitamos hallar la arista de mínimo peso entre $k(n-k)$ aristas. Como $k(n-k) < (n-1)^2$, el coste resulta $O(n^2)$. Pero el bucle del paso 2 se repite $n-1$ veces luego la complejidad es $O(n^3)$

Segunda aproximación

Una buena estructura de datos mejora la complejidad. Consideremos las listas S y $Z=V-S$. A cada vértice de Z le etiquetamos inicialmente así:

$$\begin{aligned} t(z) &= w(uz) && \text{si existe la arista } uz, \\ t(z) &= \infty && \text{si no existe} \end{aligned}$$

Ahora en el paso 2, se elige el vértice z de Z con etiqueta mínima, se halla $v \in S$ tal que $t(z) = w(vz)$, se añade z a S y se actualizan las etiquetas de los vértices de Z así:

$$t(x) := \min\{t(x), w(zx)\}$$

Ahora, en cada paso el mínimo se calcula entre $n-k$ etiquetas. Y así el coste total es

$$(n-1) + (n-2) + \dots + 1 = n(n-1)/2 = O(n^2)$$

En la implementación de una estructura de datos abstracta (EDA) Grafo, no se puede prescindir de las relaciones entre sus vértices, para lo cual se emplea una matriz de adyacencia; esta se puede representar de dos formas:



- Representación mediante un arreglo bidimensional (arreglo): Una forma sencilla de representar un grafo es utilizar una matriz bidimensional. Una matriz de adyacencia representa un grafo utilizando un espacio cuadrático. En este marco, la inicialización conllevaría un tiempo $O([V]^2)$. Aunque es posible evitar el tiempo cuadrático de inicialización el coste en espacio será aun $O([V]^2)$, que resulta razonable para grafos densos pero es totalmente inaceptable para grafos dispersos.
- Representación mediante listas: Para grafos dispersos una representación mejor es la suministrada por las listas de adyacencia. En las listas de adyacencia lo que haremos será guardar por cada nodo, además de la información que pueda contener el propio nodo, una lista dinámica con los nodos a los que se puede acceder desde él. La información de los nodos se puede guardar en un vector, al igual que antes, o en otra lista dinámica.
- Para evitar uno de los problemas que teníamos con las listas de adyacencia, que era la dificultad al recorrer la lista, podemos utilizar **matrices dispersas**, que contienen tanta información como las matrices de adyacencia, pero, en principio, no ocupan tanta memoria como estas matrices, ya que al igual que en las listas de adyacencia, sólo representaremos aquellos enlaces que existen en el grafo. La meta heurística implementada en la matriz dispersa para elevar la eficiencia de su recorrido consiste en, llegar a una posición (i, j) con el menor costo posible, si $(i > j)$ la posición se busca por las (j) , pero en ocasiones siendo $(i > j)$ el costo para acceder a esa posición es menor por las (i)

Matriz Dispersa

Una matriz dispersa o esparcida es una matriz de grandes dimensiones, en la que la mayoría de sus elementos son ceros o un valor determinado distinto de cero. Solo unos pocos son diferentes. Claramente, es un problema el almacenamiento de estos objetos.

Evidentemente almacenar muchos ceros en una matriz conlleva utilizar un espacio de almacenamiento que puede ser muy grande en proporción a la

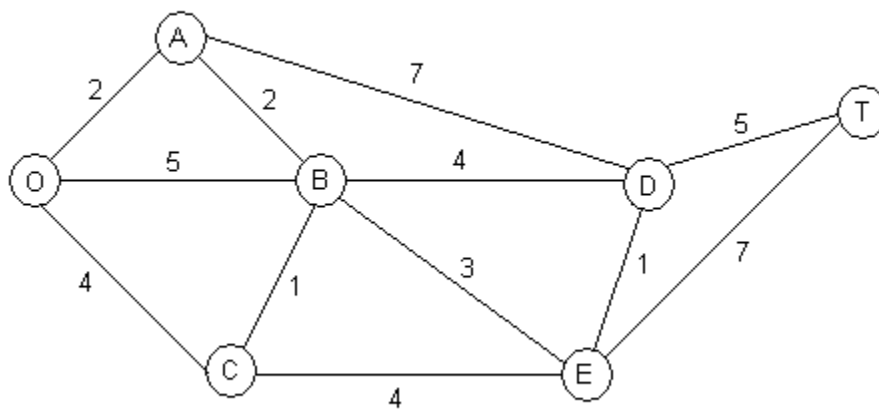
cantidad de información que realmente se necesita. Las técnicas específicas para manejo de matrices dispersas parten de estos principios para reducir el gasto de memoria y optimizar el rendimiento computacional (operaciones aritméticas).

Tendríamos como atributo de la clase el elemento disperso, que puede ser cualquier número, no solamente el cero. Este elemento es el que no se almacenaría en la matriz.

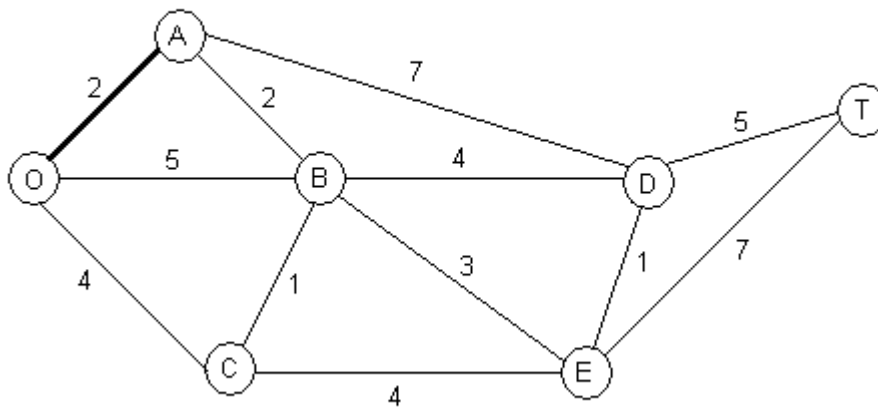
A continuación se mostrara un problema de la vida real que al que se le puede dar solución mediante la construcción de un árbol de expansión mínima:

La administración de Servada Park necesita determinar los caminos bajo los cuales se deben tener las líneas telefónicas para conectar todas las estaciones con una longitud total mínima de cable .Se describirá paso a paso la solución de este problema con base en los datos que se dan en la siguiente figura.

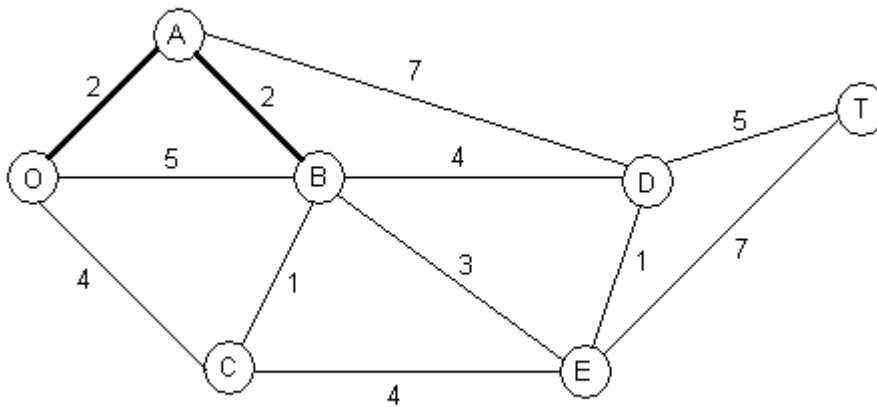
Los nodos y distancias para el problema se resumen enseguida, en donde las líneas delgadas ahora representan ligaduras potenciales.



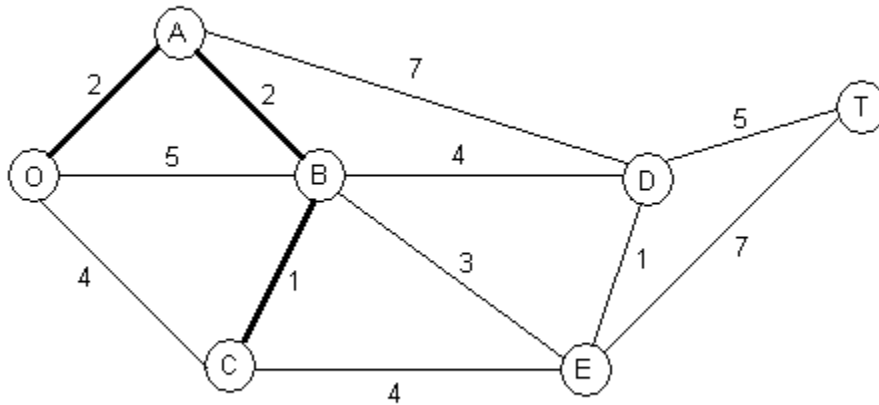
En forma arbitraria, se selecciona el nodo O para comenzar .El nodo no conectado más cercano a O es el nodo A. Se conecta el nodo A al nodo O.



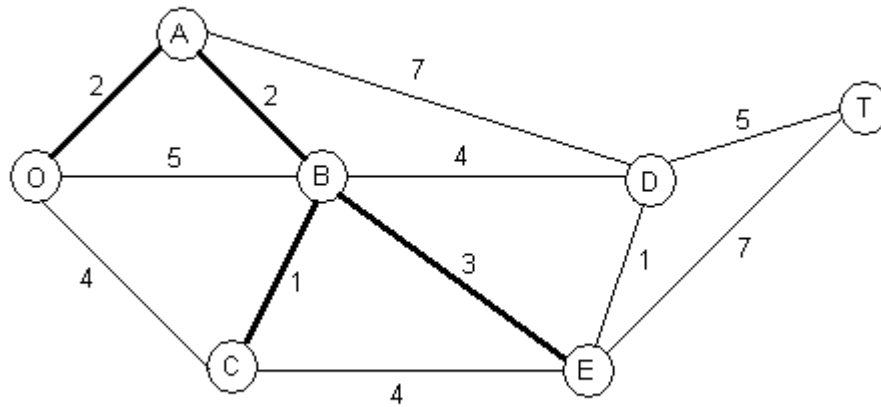
El nodo no conectado más cercano a cualesquiera de los nodos O o A es el nodo B (más cercano a A). Se conecta el nodo B al nodo A.



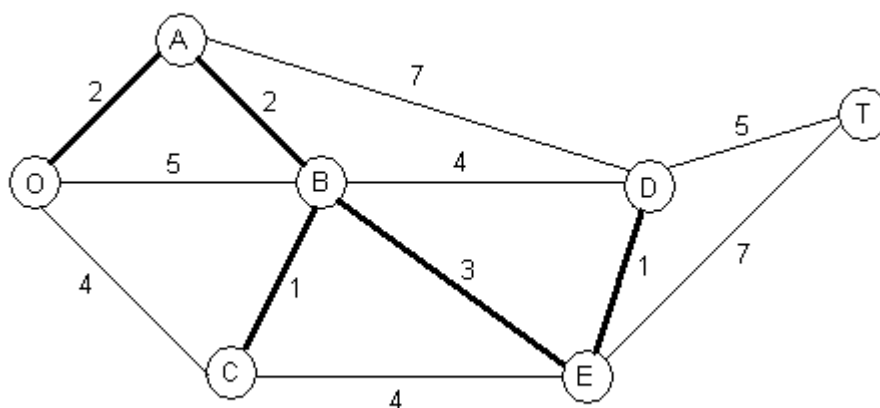
El nodo no conectado más cercano a O, A o B es el nodo C (más cercano a B). Se conecta el nodo C al nodo B.



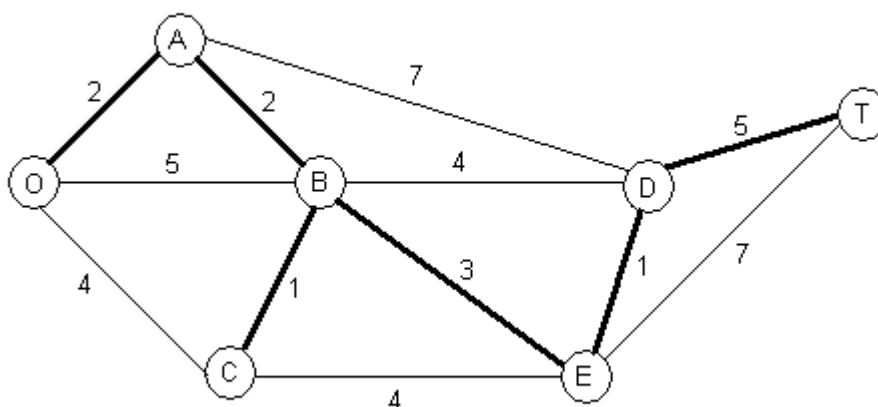
El nodo no conectado más cercano a O, A, B o C es el nodo E (más cercano B).
Se conecta el nodo E al nodo B.



El nodo no conectado más cercano a O, A, B, C o E es el nodo D (más cercano a E). Se conecta el nodo D al nodo E.



El único nodo no conectado es el nodo T. Esta más cerca del nodo D. Se conecta el nodo T al nodo D.



Todos los nodos han quedado conectados, por lo que esta es la solución (óptima) que se buscaba.

1.3 Sistemas automatizados existentes vinculados al campo de acción

Con el desarrollo de las nuevas tecnologías el universo de información ha crecido exponencialmente, provocando que los especialistas se vean en la



necesidad de hacer uso cada vez más frecuentes de las ventajas que nos brinda los sistemas informáticos en la investigación de operaciones en la solución de problemas de árbol de expansión mínima, apoyando la idea anterior, podemos citar el siguiente planteamiento:

"El trabajo profesional vinculado a la información en cualquier organización, exige el dominio de un conjunto de variables que están presentes en su tratamiento. Asimismo, el dominio de las funciones de la gestión y de algunas de las principales herramientas que desarrolladas en las últimas décadas, no deviene en mero elemento cultural del profesional que maneja información, sino en una necesidad imperiosa". [2]

En la actualidad existe una gran cantidad de software y núcleos matemáticos dedicados a la solución de problemas de árbol de expansión mínima como son Lingo, Lindo, Microsoft Excel, Matemática, PLeMath, Matlab, Maple. Los cuales independientemente que soportan métodos eficientes y bien validados que permiten una elevada eficacia en la soluciones de problemas, a medida que estos se desarrollan aumentan la demanda de hardware: memoria(RAM), procesador y capacidad de almacenamiento , necesitando computadoras con condiciones mas exigentes, Debido a que son de propósito general y no exactamente para resolver PAEM, carecen de integridad respecto a los conocimientos para el eficiente análisis e interpretación de los resultados, obliga al usuario a un profundo estudio de los diferentes manuales de usuarios de los mismo, además estas herramientas están ligadas a sistemas operativos propietarios como Microsoft Windows. Por estas razones el Ministerio de Educación Superior (MES) se a trazado una estrategia a mediano plazo para migrar a plataformas patentizadas baja licencias GNU.

1.4 Tendencias y tecnologías actuales

1.4.1 Lenguajes de programación para el desarrollo de aplicaciones

Teniendo en cuenta el desarrollo que han venido adquiriendo los deferentes lenguajes de programación para la creación de aplicaciones de escritorio y valorando los conocimientos alcanzados durante el estudio de la carrera. Se ha considerado como posibles propuestas los lenguajes C++, Delphi, Java, siendo



estos los más conocidos. Para los cuales se ha realizado una breve descripción de las características generales:

C++

Es un lenguaje de programación, diseñado a mediados de los años 1980, por Bjarne Stroustrup, como extensión del lenguaje de programación C.

El nombre C++ fue propuesto por Rick Masciatti en el año 1983, cuando el lenguaje fue utilizado por primera vez fuera de un laboratorio científico. Antes se había usado el nombre "C con clases".

Se puede decir que abarca tres paradigmas de la programación: la programación estructurada, la programación genérica y la programación orientada a objetos.

Además posee una serie de propiedades difíciles de encontrar en otros lenguajes de alto nivel:

- Posibilidad de redefinir los operadores (sobrecarga de operadores).
- Identificación de tipos en tiempo de ejecución.

Está considerado por muchos como el lenguaje más potente, debido a que permite trabajar tanto a alto como a bajo nivel, sin embargo es a su vez uno de los que menos automatismos (obliga a hacerlo casi todo manualmente al igual que C) lo que "dificulta" mucho su aprendizaje.

Delphi

Es un entorno de desarrollo de software (IDE) diseñado para la programación de propósito general con énfasis en la programación visual. El Delphi utiliza como lenguaje de programación una versión moderna de Pascal llamada Object Pascal. Es producido comercialmente por la empresa estadounidense CodeGear. En sus diferentes variantes, permite producir archivos ejecutables para Windows, Linux y la plataforma .NET.

Está basado en una versión moderna de Pascal, denominada Object Pascal. Borland en los últimos años defendía que el nombre correcto del lenguaje es también Delphi, posiblemente debido a pretensiones de marca, aunque en sus mismos manuales el nombre del lenguaje aparecía como Object Pascal, por lo que la comunidad de programadores no ha adoptado mayoritariamente este



cambio (supuesta aclaración, según Borland). Object Pascal expande las funcionalidades del Pascal estándar:

- Soporte para la programación orientada a objetos (habitualmente llamada POO) también existente desde Turbo Pascal 5.5, pero más evolucionada en cuanto a:
- Encapsulación: declarando partes privadas, protegidas, públicas y publicadas de las clases
- Propiedades: concepto nuevo que luego han adaptado muchos otros lenguajes. Las propiedades permiten usar la sintaxis de asignación para setters y getters.
- Simplificación de la sintaxis de referencias a clases y punteros.
- Soporte para manejo estructurado de excepciones, mejorando sensiblemente el control de errores de usuario y del sistema.
- Programación activada por eventos (event-driven), posible gracias a la técnica de delegación de eventos. Esta técnica permite asignar el método de un objeto para responder a un evento lanzado sobre otro objeto. Fue adoptada por Niklaus Wirth, autor del Pascal Original, e incorporada a otros de sus lenguajes como Component Pascal.

Java

Es un lenguaje de programación desarrollado por Sun Microsystems a principios de los años 1990. Las aplicaciones Java están típicamente compiladas en un bytecode, aunque la compilación en código máquina nativo también es posible. En el tiempo de ejecución, el bytecode es normalmente interpretado o compilado a código nativo para la ejecución, aunque la ejecución directa por hardware del bytecode por un procesador Java también es posible.

La implementación original y de referencia del compilador, la máquina virtual y las librerías de clases de Java fueron desarrolladas por Sun Microsystems en 1995. Desde entonces, Sun ha controlado las especificaciones, el desarrollo y evolución del lenguaje a través del Java Community Process, si bien otros han desarrollado también implementaciones alternativas de estas tecnologías de Sun, algunas incluso bajo licencias de software libre.

Entre noviembre de 2006 y mayo de 2007, Sun Microsystems liberó la mayor parte de sus tecnologías Java bajo la licencia GNU GPL, de acuerdo con las especificaciones del Java Community Process, de tal forma que prácticamente todo el Java de Sun es ahora software libre (aunque la biblioteca de clases de Sun que se requiere para ejecutar los programas Java todavía no es software libre).

Las características principales que nos ofrece Java son:

- **Simple**

Ofrece toda la funcionalidad de un lenguaje potente. Debido a que C y C++ son los lenguajes más difundidos, Java se diseñó para ser parecido a C++ y así facilitar un rápido y fácil aprendizaje.

Elimina muchas de las características de otros lenguajes para mantener reducidas las especificaciones del lenguaje y añadir características muy útiles como el garbage collector (reciclador de memoria dinámica). No es necesario preocuparse de liberar memoria, el reciclador se encarga de ello y como es un thread de baja prioridad, cuando entra en acción, permite liberar bloques de memoria muy grandes, lo que reduce la fragmentación de la memoria.

- **Orientado a objetos**

Con el objetivo de mantener la simplicidad del lenguaje. Java trabaja con sus datos como objetos y con interfaces a esos objetos. Soporta las tres características propias del paradigma de la orientación a objetos: encapsulación, herencia y polimorfismo. Las plantillas de objetos son llamadas, como en C++, clases y sus copias, instancias. Estas instancias, necesitan ser construidas y destruidas en espacios de memoria.

Incorpora funcionalidades como por ejemplo, la resolución dinámica de métodos mediante una interfaz específica llamada RTTI (RunTime Type Identification) que define la interacción entre objetos excluyendo variables de instancias o implementación de métodos. Las clases en Java tienen una representación en el runtime que permite a los programadores interrogar por el tipo de clase y enlazar dinámicamente la clase con el resultado de la búsqueda.

- **Distribuido**



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

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

- **Robusto**

Realiza verificaciones en busca de problemas tanto en tiempo de compilación como en tiempo de ejecución. La comprobación de tipos en Java ayuda a detectar errores, lo antes posible, en el ciclo de desarrollo. Java obliga a la declaración explícita de métodos, reduciendo así las posibilidades de error. Maneja la memoria para eliminar las preocupaciones por parte del programador de la liberación o corrupción de memoria.

También implementa los arrays auténticos, en vez de listas enlazadas de punteros, con comprobación de límites, para evitar la posibilidad de sobrescribir o corromper memoria resultado de punteros que señalan a zonas equivocadas. Estas características reducen drásticamente el tiempo de desarrollo de aplicaciones en Java.

- **Arquitectura neutral**

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

- **Seguro**

La seguridad en Java tiene dos facetas. En el lenguaje, características como los punteros o el casting se eliminan para prevenir el acceso ilegal a la memoria.



Cuando se usa Java para crear un navegador, se combinan las características del lenguaje con protecciones de sentido común aplicadas al propio navegador.

El código Java pasa muchos tests antes de ejecutarse en una máquina. El código se pasa a través de un verificador de byte-codes que comprueba el formato de los fragmentos de código y aplica un probador de teoremas para detectar fragmentos de código ilegal -código que falsea punteros, viola derechos de acceso sobre objetos o intenta cambiar el tipo o clase de un objeto-.

El Cargador de Clases también ayuda a Java a mantener su seguridad, separando el espacio de nombres del sistema de ficheros local, del de los recursos procedentes de la red. Esto limita cualquier aplicación del tipo Caballo de Troya, ya que las clases se buscan primero entre las locales y luego entre las procedentes del exterior.

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

- **Multihebra**

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

1.4.2 Fundamentación de la selección de lenguaje a utilizar

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

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

- Aritmética de punteros.



- Referencias.
- Registros (struct).
- Definición de tipos (typedef).
- Macros (#define).
- Necesidad de liberar memoria (free).

Aunque, en realidad, lo que hace es eliminar las palabras reservadas (struct, typedef), ya que las clases son algo parecido.

Además, el intérprete completo de Java que hay en este momento es muy pequeño, solamente ocupa 215 Kb de RAM.

Con respecto a delphi tiene ventajas en cuanto a seguridad ya que al tener la característica de ser interpretado el código Java pasa a través de un verificador de byte-codes que comprueba el formato de los fragmentos de código y aplica un probador de teoremas para detectar fragmentos de código ilegal -código que falsea punteros, viola derechos de acceso sobre objetos o intenta cambiar el tipo o clase de un objeto. El Cargador de Clases, separando el espacio de nombres del sistema de ficheros local, del de los recursos procedentes de la red. Esto limita cualquier aplicación del tipo Caballo de Troya, ya que las clases se buscan primero entre las locales y luego entre las procedentes del exterior.

1.4.3 Metodologías para el desarrollo de Sistemas Informáticos

Unified Modeling Language (UML) o Lenguaje Unificado de Modelado, es un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de software. Es utilizado para entender, diseñar, configurar, mantener y controlar la información sobre los sistemas a construir.

UML capta la información sobre la estructura estática y el comportamiento dinámico de un sistema. Un sistema se modela como una colección de objetos discretos que interactúan para realizar un trabajo que finalmente beneficia a un usuario externo.

El lenguaje de modelado pretende unificar la experiencia pasada sobre técnicas de modelado e incorporar las mejores prácticas actuales en un acercamiento estándar.



UML no es un lenguaje de programación. Las herramientas pueden ofrecer generadores de código de UML para una gran variedad de lenguaje de programación, así como construir modelos por ingeniería inversa a partir de programas existentes.

Es un lenguaje de propósito general para el modelado orientado a objetos. UML es también un lenguaje de modelado visual que permite una abstracción del sistema y sus componentes.

UML es un lenguaje de modelado que pueden usar todos los modeladores. No tiene propietario y está basado en el común acuerdo de gran parte de la comunidad informática.

UML no pretende ser un método de desarrollo completo. No incluye un proceso de desarrollo paso a paso. UML incluye todos los conceptos que se consideran necesarios para utilizar un proceso moderno iterativo, basado en construir una sólida arquitectura para resolver requisitos dirigidos por casos de uso.

Objetivos del UML

- Ser tan simple como sea posible pero manteniendo la capacidad de modelar toda la gama de sistemas que se necesita construir. UML necesita ser lo suficientemente expresivo para manejar todos los conceptos que se originan en un sistema moderno, tales como la concurrencia y distribución, así como también los mecanismos de la ingeniería de software, como son la encapsulación y componentes.
- Ser un lenguaje universal, como cualquier lenguaje de propósito general.
- Imponer un estándar mundial.

Fundamentación de la metodología a utilizar para el modelado

Existen varias metodologías que usan el lenguaje UML para indicar el camino a seguir para el desarrollo de sistemas de informáticos las cuales han ido evolucionando. Algunas de ellas son: OMT, XP, Microsoft Solution Framework, OBJECTORY, BOOCH, RUP y AUP.

Para el desarrollo de este proyecto hemos tomado en consideración utilizar la metodología del Proceso Unificado del Racional (RUP) para el cual se hace una caracterización a continuación.



Los aspectos que definen el Proceso Unificado se resumen en tres frases claves:

- dirigido por casos de uso
- centrado en la arquitectura
- iterativo e incremental

Un **caso de uso** es un fragmento de funcionalidad del sistema que proporciona al usuario un resultado importante. Los casos de uso representan los requisitos funcionales. Todos los casos de uso juntos constituyen el **modelo de casos de uso** que describe la funcionalidad total del sistema.

Los casos de uso guían el proceso entero de desarrollo.

La **arquitectura del software** es una vista del diseño completo con las características más importantes resaltadas, dejando los detalles de lado.

Cada producto tiene una función y una forma. Ninguna es suficiente por sí misma, debiendo interactuar entre ellas y evolucionar en paralelo. La función corresponde a los casos de uso y la forma a la arquitectura.

Para encontrar la forma, la arquitectura, los arquitectos deben trabajar sobre la comprensión de los casos de uso claves que vienen a suponer entre el 5 y el 10 por ciento del total de casos de uso.

El Proceso Unificado es **iterativo e incremental** lo que supone dividir el proyecto en pequeñas partes denominadas mini-proyectos. Cada mini-proyecto es una **iteración** que genera un **incremento**. Las iteraciones hacen referencia a pasos en el flujo de trabajo y los incrementos al crecimiento del producto.

Las iteraciones deben ser controladas, deben ser seleccionadas y ejecutarse de forma planificada. Por estas razones son consideradas mini-proyectos.

Una iteración trata un conjunto de casos de uso que amplían la utilidad del producto desarrollado hasta ese momento. Cada iteración gestiona **los riesgos** identificados más importantes.

Este proceso iterativo trae como beneficios:

- Reduce el riesgo a los costes de un solo incremento.
- Reduce el riesgo de no sacar el producto en plazo.



- Los trabajadores son más eficientes al trabajar para obtener resultados a corto plazo.
- Buena adaptación a los requisitos cambiantes.

La **arquitectura** proporciona la estructura sobre la que guiar las **iteraciones**, mientras que los **casos de uso** definen los objetivos y dirigen el trabajo de cada **iteración**.

El Proceso Unificado se repite a lo largo de una serie de **ciclos** que constituyen la vida de un sistema. Cada **ciclo** concluye con una **nueva versión del producto**. El **producto terminado** incluye *los requisitos, casos de uso, especificaciones no funcionales, casos de prueba, código fuente incluido en componentes ejecutables, manuales, el modelo de arquitectura y el modelo visual (modelos UML)*.

Cada **ciclo** consta de **cuatro fases: Inicio, Elaboración, Construcción y Transición**.

Cada **fase**, a su vez, se subdivide en **iteraciones**. Cada **fase** debe terminar en un **hito** que contemple la disponibilidad de ciertos modelos o documentos. El **hito** implica la toma de decisiones.

Una iteración típica dentro de una fase pasa por los siguientes flujos de trabajo:

- **Requisitos**
- **Análisis**
- **Diseño**
- **Implementación**
- **Prueba**

Descripción de las Fases del Ciclo:

- **Fase de INICIO**

Se desarrolla una descripción del producto final y se presenta el análisis de negocio para el producto.

Se construye un modelo de casos de uso simplificado.

Se esbozan los subsistemas más importantes lo que origina una arquitectura provisional.



Se identifican los riesgos más importantes.

Se planifica la fase de ELABORACIÓN.

Se estima el proyecto de manera aproximada.

- **Fase de ELABORACIÓN**

Se especifican en detalle la mayoría de los casos de uso del producto.

Se diseña la arquitectura del sistema a través de vistas de todos los modelos del sistema (análisis, diseño, implementación y despliegue), obteniéndose una

línea base de la arquitectura.

Se realizan los casos de uso más críticos identificados en la **Fase de INICIO.**

El jefe del proyecto ya está en disposición de planificar y estimar los recursos necesarios para terminar el proyecto.

- **Fase de CONSTRUCCION**

Se crea el producto.

La **línea base de la arquitectura** crece hasta convertirse en el sistema completo.

La descripción evoluciona hasta convertirse en un **producto** preparado para entregarse al usuario.

Al final de esta fase, el producto contiene todos los casos de uso acordados con el usuario, aunque puede que no estén libres de defectos.

Muchos de los defectos se descubrirán y solucionarán.

- **Fase de TRANSICIÓN**

Cubre el período en que el **producto** se convierte en versión beta.

Los desarrolladores corrigen los defectos detectados e incorporan algunas mejoras.

Esta fase conlleva a actividades de fabricación, formación a usuarios, corrección de defectos tras la entrega.

El equipo de mantenimiento valora los defectos tras la entrega en dos categorías: los que tienen suficiente impacto para justificar una versión incrementada y los que pueden corregirse en la siguiente versión normal.



1.5 Conclusiones

En este capítulo se realiza un análisis completo de las tecnologías que serán utilizadas a lo largo del desarrollo del sistema propuesto, y se fundamentan las elecciones del lenguaje, y la metodología a utilizar, para desarrollar una herramienta informática de apoyo al proceso de solución de problemas de árbol de expansión mínima, que utilice pocos recursos computacionales, sea multiplataforma, brinde al usuario seguridad y rapidez en los tiempos de respuesta, y libre de costo alguno.

Capítulo 2 Modelo del dominio

2.1 Introducción

Analizando la descripción de los procesos realizada en el capítulo I, llegamos a la conclusión de que el negocio que se está estudiando tiene muy bajo nivel de estructuración, RUP propone para estos casos realizar un modelo del dominio; que no es más que una representación visual de las clases conceptuales u objetos del mundo real en un dominio de interés, por lo que permite mostrar al usuario los principales conceptos que se manejan en el dominio del sistema en desarrollo. Este modelo va a contribuir posteriormente a identificar algunas clases que se utilizarán en el sistema. Tal modelo no incluye las responsabilidades que llevan a cabo las personas.

De modo que en el presente capítulo se especificará lo siguiente:

1. Definición de entidades y los conceptos principales, los cuales se definirán a partir de su símbolo, su intención, y su extensión.

Símbolo: Palabra que representa al concepto.

Intención: La definición del concepto.

Extensión: El conjunto de ejemplos a que se aplica el concepto.

2. Diagrama del Modelo del Dominio.
3. Representación de los requerimientos funcionales y no funcionales del sistema.

2.2 Definición de las entidades y los conceptos principales

Tabla 1 Entidades y conceptos del dominio.

Símbolo	Intención	Extensión
Usuario	Se le denomina usuarios a todas aquellas personas que interactúan con el sistema.	Personas
Modelo	Se refiere al conjunto nodos y arcos que forman un grafo.	Grafo
Algoritmos Matemáticos	Se refieren a los métodos para la solución de problemas de	Versión de Algoritmo de Prim



	Árbol de Expansión Mínima.	
Mostrar Solución	Es la solución obtenida una vez que se le aplica el algoritmo al modelo.	Solución optima (árbol de expansion minima)

2.3 Representación del modelo del dominio

Figura 1 Diagrama del Modelo del Dominio.

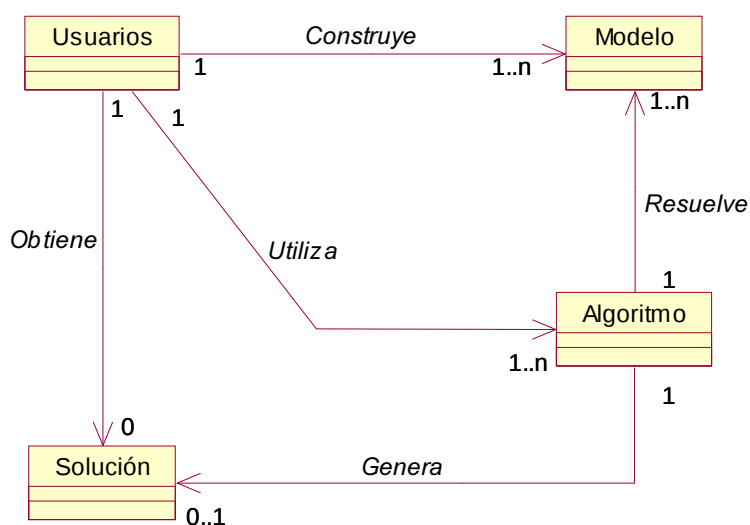


Diagrama del Modelo del Dominio.

2.4 Requerimientos Funcionales y no Funcionales del Sistema.

2.4.1 Requisitos Funcionales

Los requisitos funcionales indican el comportamiento del sistema. Posteriormente estos requisitos son modelados a través del diagrama de casos de uso del sistema.

Requerimientos funcionales “Gestión de modelo”.

1. Insertar nodo
 - 1.1 Editar nodo
2. Eliminar nodo
3. Insertar arco
 - 3.1 Editar arco



4. Cambiar peso

5. Crear grafo

Requerimiento funcional “Mover nodo”.

6. Mover nodo

Requerimiento funcional “Construir AEM”.

7. Construir AEM

Requerimiento funcional “Mostrar AEM”

8. Mostrar AEM

Requerimiento funcional “Guardar solución en un fichero”.

9. Guardar solución.

2.4.2 Requisitos no Funcionales

Los requerimientos no funcionales son propiedades o cualidades que el producto debe tener. Debe pensarse en estas propiedades como las características que hacen al producto atractivo, usable, rápido o confiable.

- *Apariencia o interfaz externa.*

La interfaz debe ser sencilla y amigable puesto que los usuarios pueden o no ser personas expertas. La respuesta del sistema ha de ser rápida.

- *Usabilidad.*

El sistema podrá ser usado por cualquier persona que posea conocimientos básicos en el manejo de la computadora.

- *Rendimiento.*

Esta aplicación debe tener un rendimiento óptimo. Debe ser rápida y el tiempo de respuesta debe ser el mínimo posible.

- *Portabilidad.*

El sistema es multiplataforma, todo depende de la portabilidad que alcance la maquina virtual de java

- *Ayuda y documentación en línea.*

El sistema cuenta con una opción de ayuda, la cual guía al usuario en el funcionamiento del sistema.



2.5 Conclusiones.

En este capítulo se realizó la definición de los requerimientos funcionales y no funcionales del sistema propuesto, obteniéndose a partir del análisis de los procesos del dominio. Gracias a esto se puede dar paso al diseño e implementación del sistema, tratando de que se cumplan todos los requerimientos y las funciones que se han definido en este capítulo.

Capítulo 3 Diseño e Implementación del Sistema

3.1 Introducción

Se ofrecen aspectos que dentro de la metodología de ingeniería del software empleada, brindan informaciones específicas para el correcto entendimiento de la solución propuesta. Se definirá los actores del sistema, se realizará una representación del diagrama de Casos de Uso con los que interactúa el usuario, así como los Diagrama de Clases, de Secuencia, de Componentes y de Despliegue.

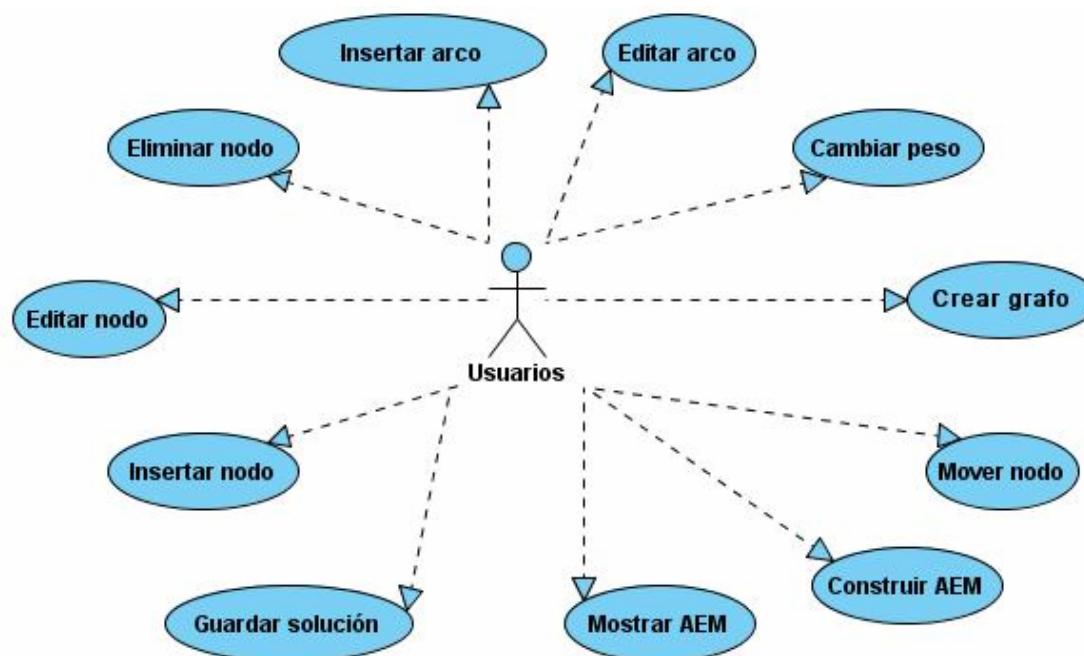
3.2 Actores del sistema a automatizar

Tabla 2. Definición de actores del sistema a automatizar

Nombre del actor	Descripción
Usuarios	Se le denomina usuarios a todas aquellas personas que interactúan con el sistema.

3.3 Diagrama de casos de uso del sistema a automatizar

Figura 2. Diagrama de Casos de Uso del Sistema.





3.4 Descripción de los casos de uso

A partir del diagrama de casos de uso del sistema procedemos a la descripción de cada caso de uso en particular, se puede apreciar una de estas descripciones en la tabla que aparece a continuación, las restantes se encuentran en el Anexo 1.

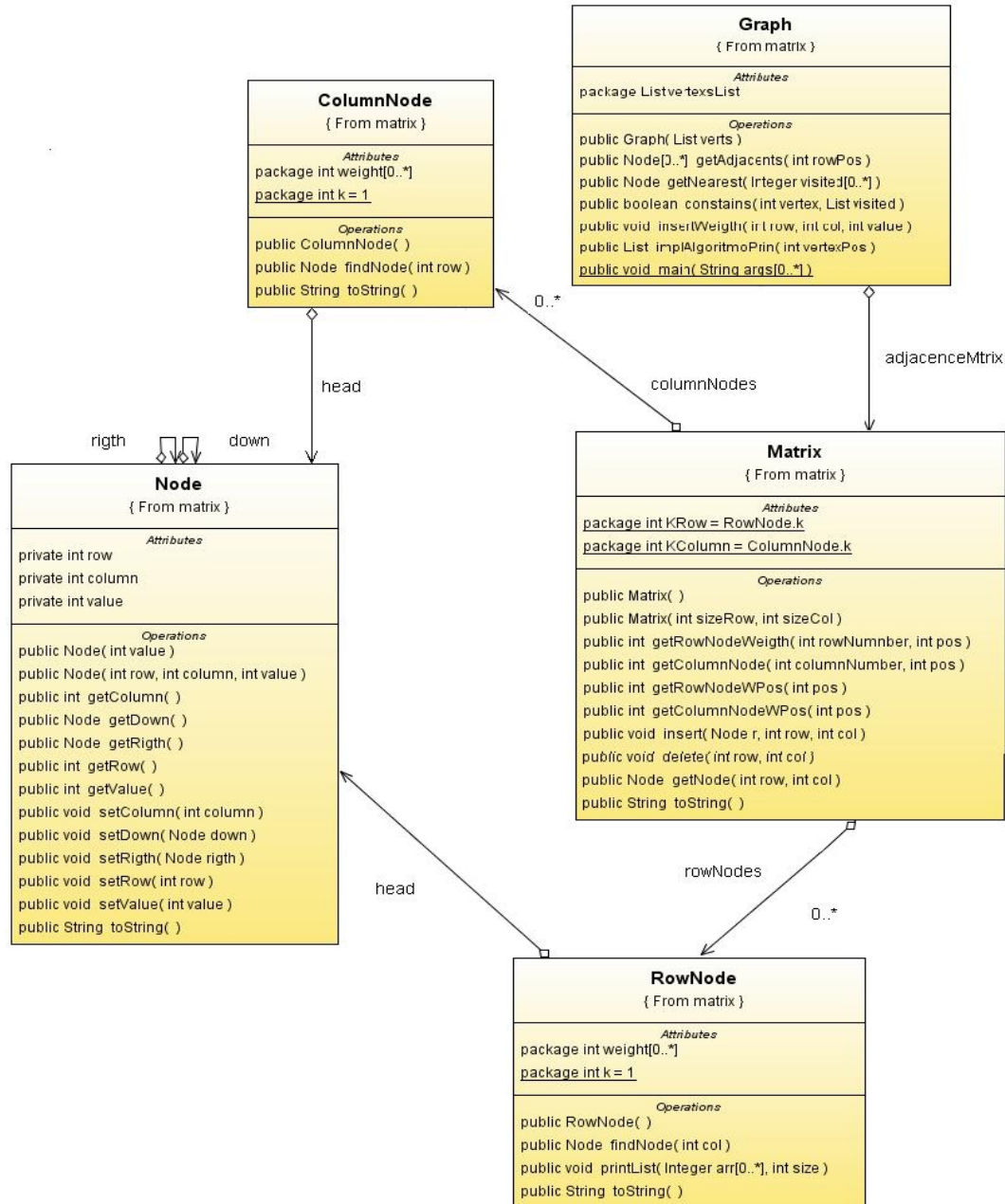
Tabla 3. Descripción del caso de uso (Construir AEM)

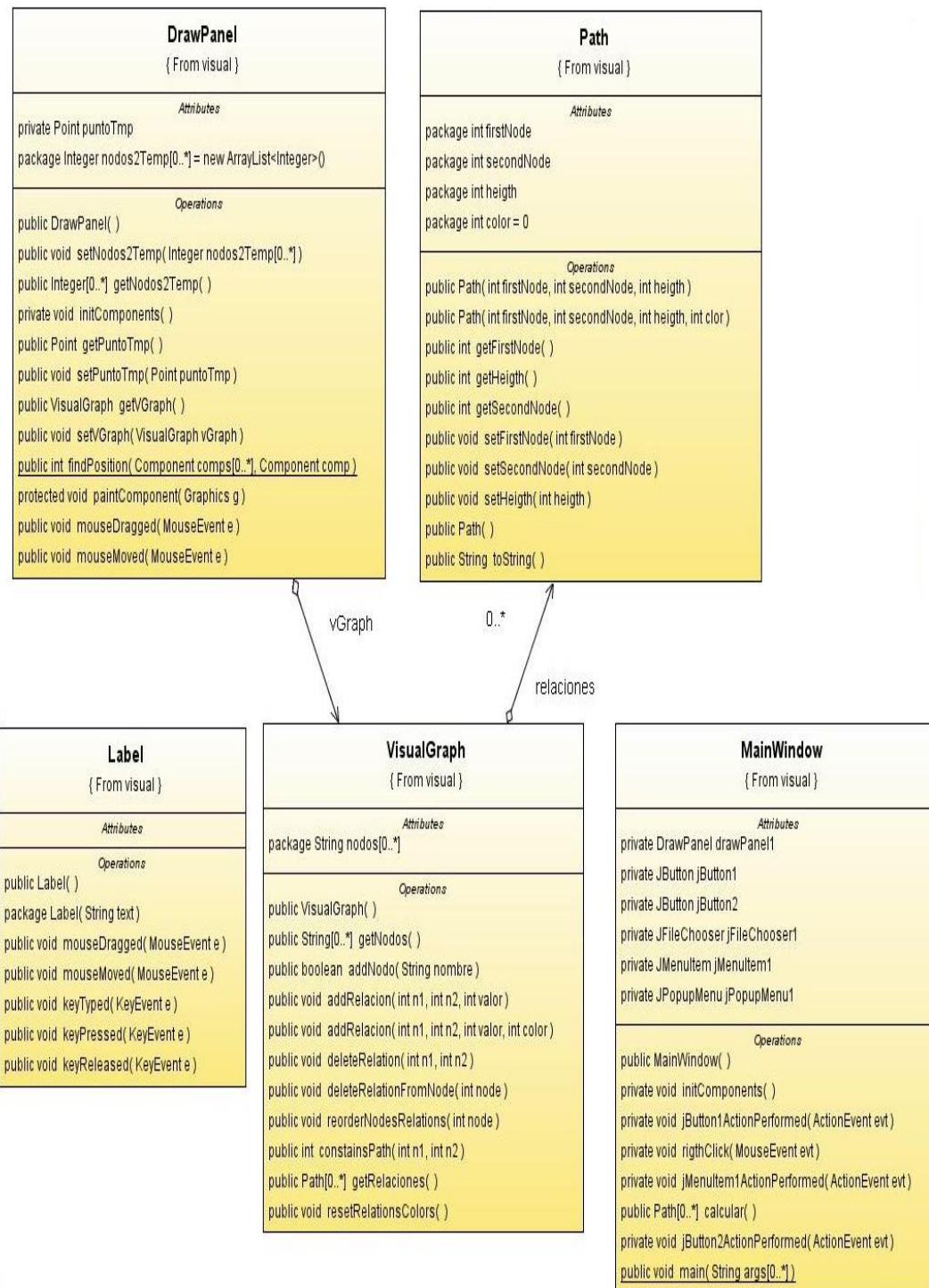
Nombre del caso de uso	Construir AEM.
Actores	Usuario (inicia).
Resumen	Este caso de uso lo inicia el usuario para obtener una solución aplicando el Algoritmo al grafo para extraer un Árbol de Expansión Mínima.
Referencias	R7
Precondiciones	Debe existir el grafo con el cual se trabajara.
Poscondiciones	El sistema devuelve un AEM.

3.5 Diagrama de clases del diseño

3.5.1 Diagrama de Clases General

Figura 1. Diagrama de clases del sistema







3.5.2 Descripción General de las Clases

En la siguiente tabla aparece la descripción de la clase “Graph”, las restantes descripciones aparecen en el Anexo 2.

Tabla 4. Descripción de la clase “Graph”

Nombre: Graph	
Tipo de clase: Entidad	
Atributo	Tipo
<i>vertexList</i>	List
<i>adjacenceMatrix</i>	Matrix
Para cada responsabilidad:	
Nombre	Descripción
boolean constains(int, List)	Verifica si el número del nodo se encuentra en la lista.
List getAdjacent(int)	Devuelve los nodos relacionados con el nodo especificado
Node getNearest(List)	Devuelve el nodo más cercano a la lista especificada.
void insertWeigth(int, int , int)	Inserta un arco con el peso especificado entre los vértices especificados.
List implAlgoritmoPrin(int)	Construye el árbol de expansión mínima.
Graph(List)	Construye un grafo vacio con los vértices especificados.

3.6 Principios de diseño

3.6.1 Interfaz de usuario

Figura 4. Interfaz de usuario “Insertar nodo”

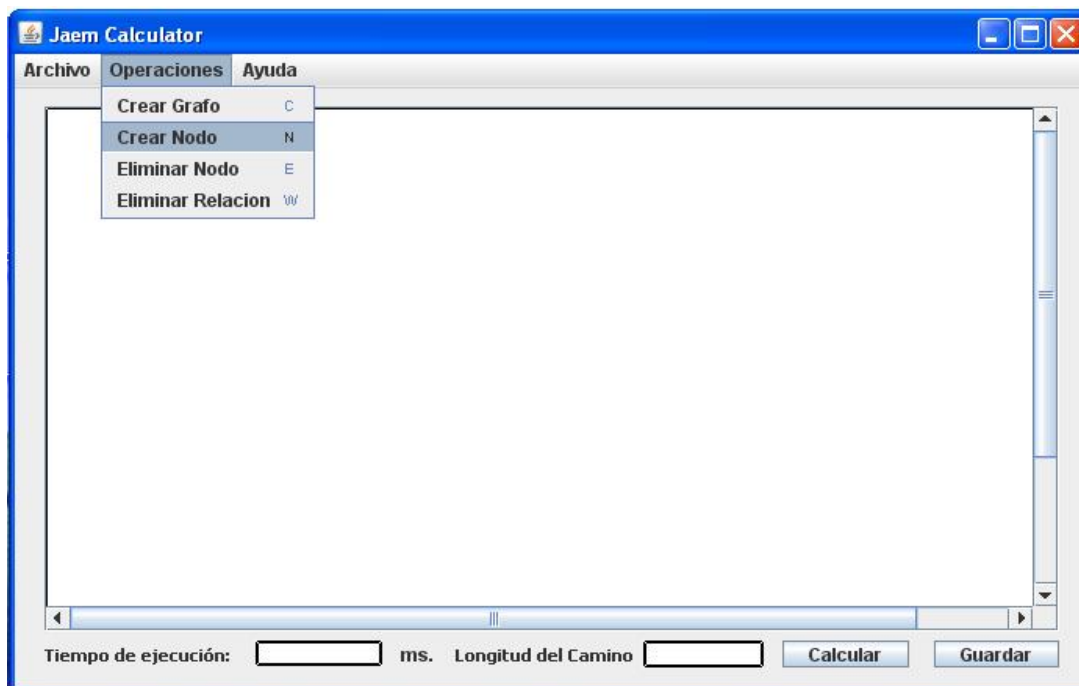


Figura 5. Interfaz de usuario “Editar nodo”

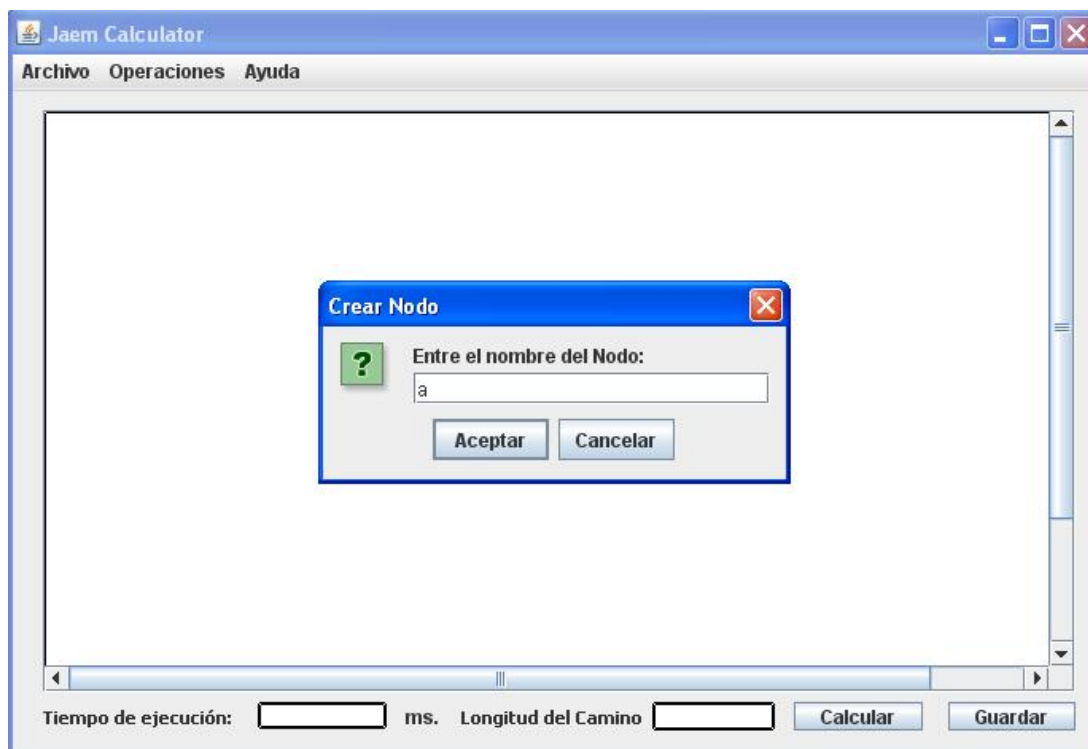




Figura 6. Interfaz de Usuario “Eliminar nodo”

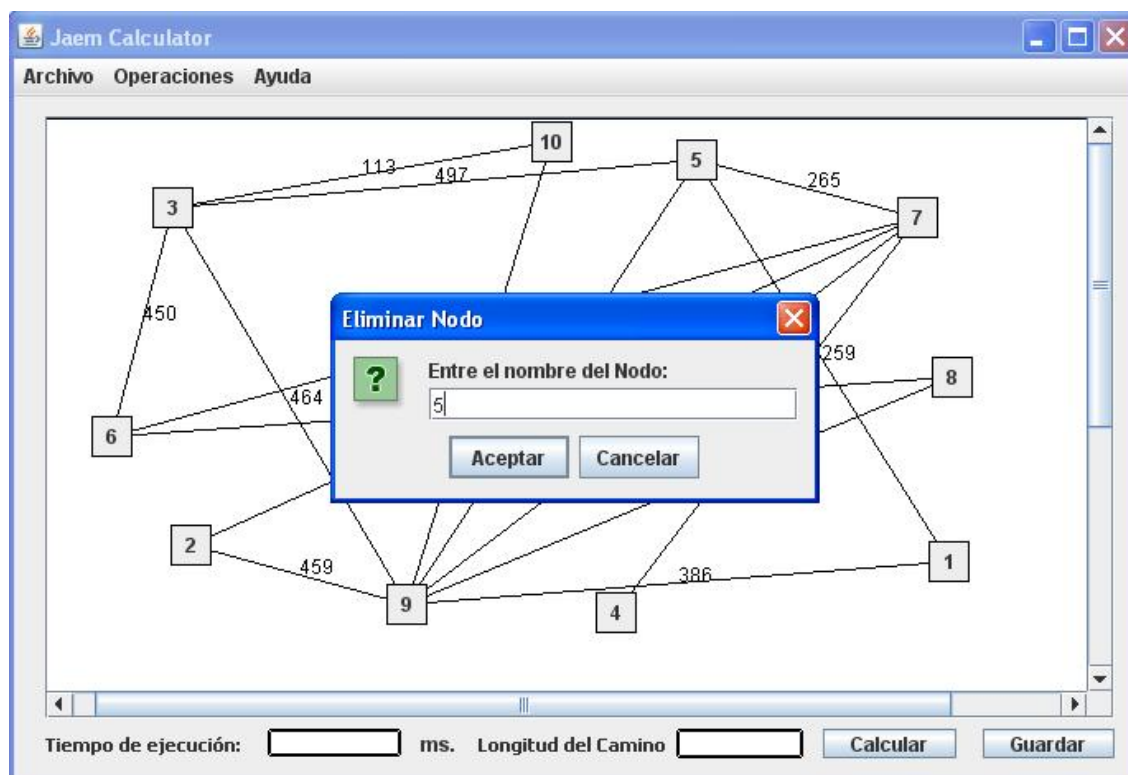


Figura 7. Interfaz de Usuario “Insertar arco”

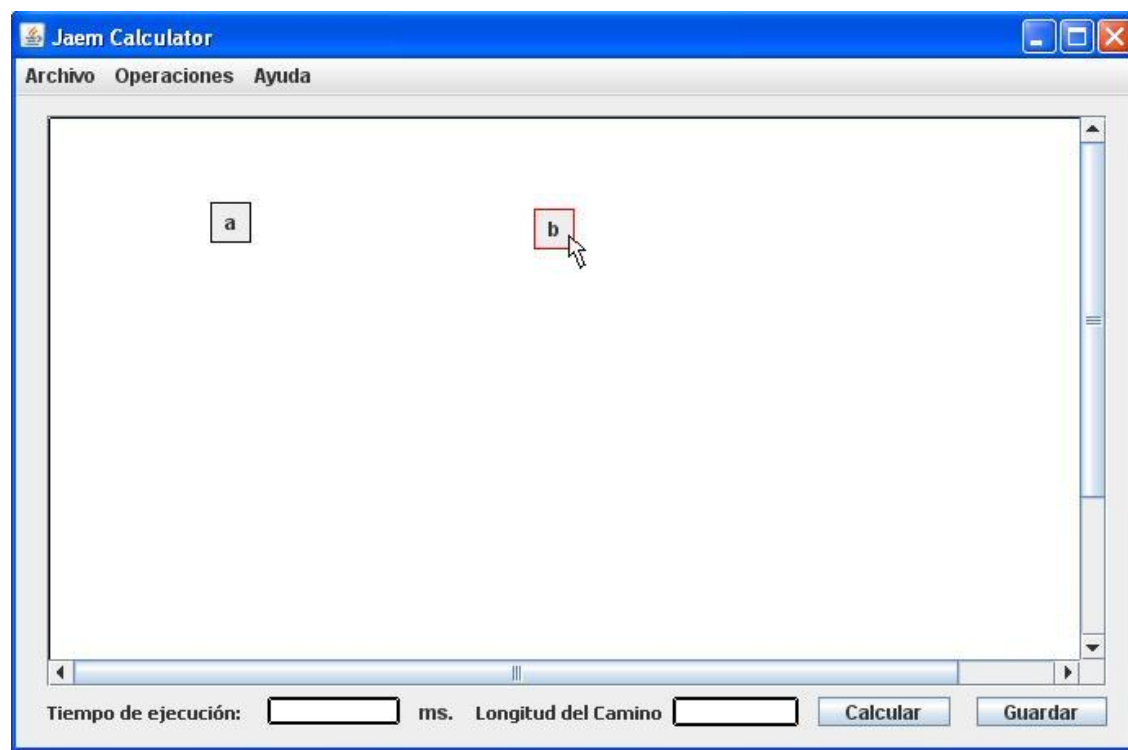


Figura 8. Interfaz de usuario “Editar arco”

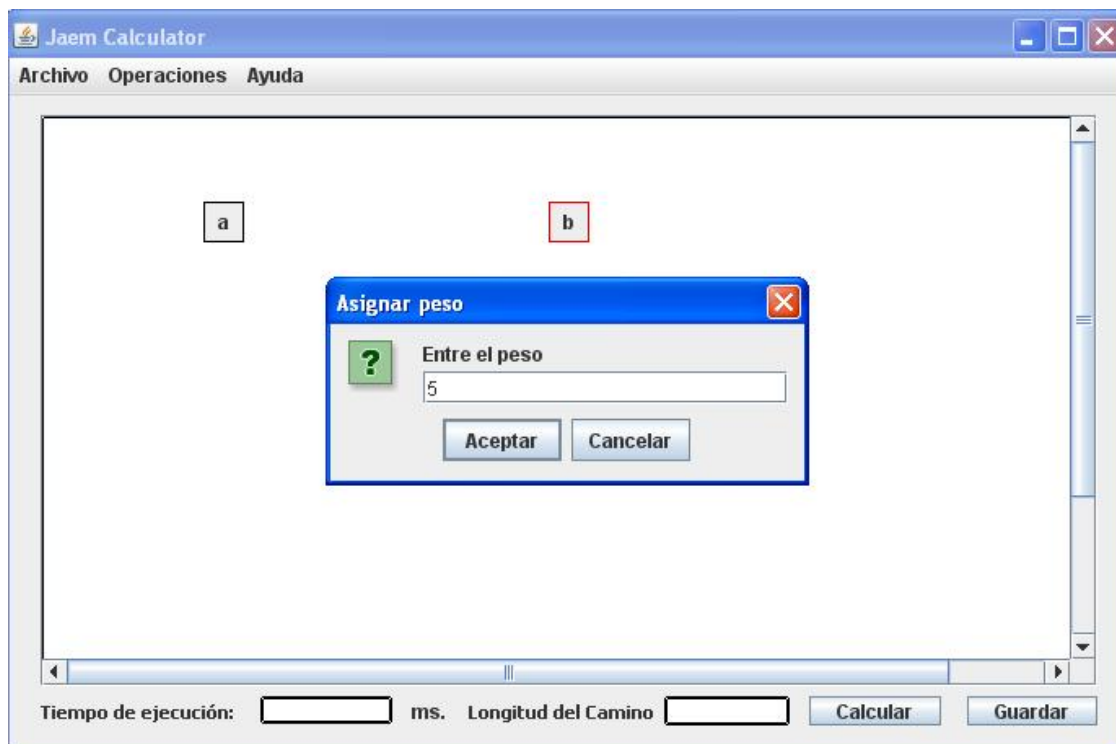


Figura 9. Interfaz de usuario “Cambiar peso”

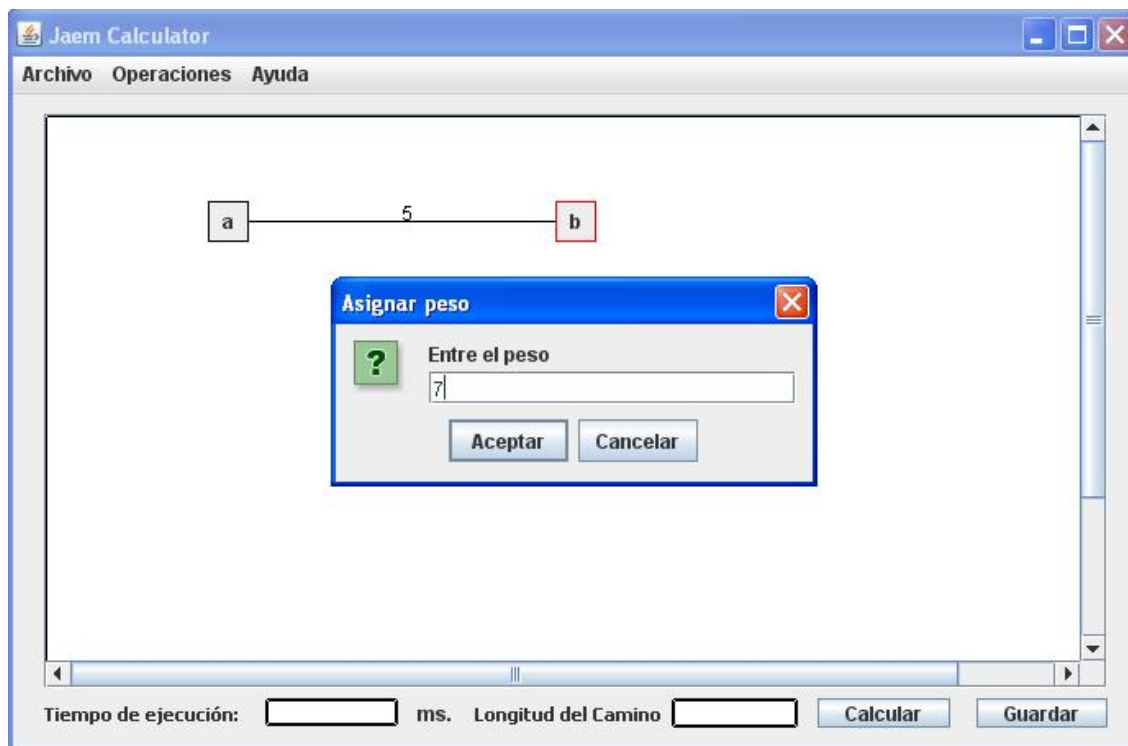




Figura 10. Interfaz de usuario “Crear grafo”

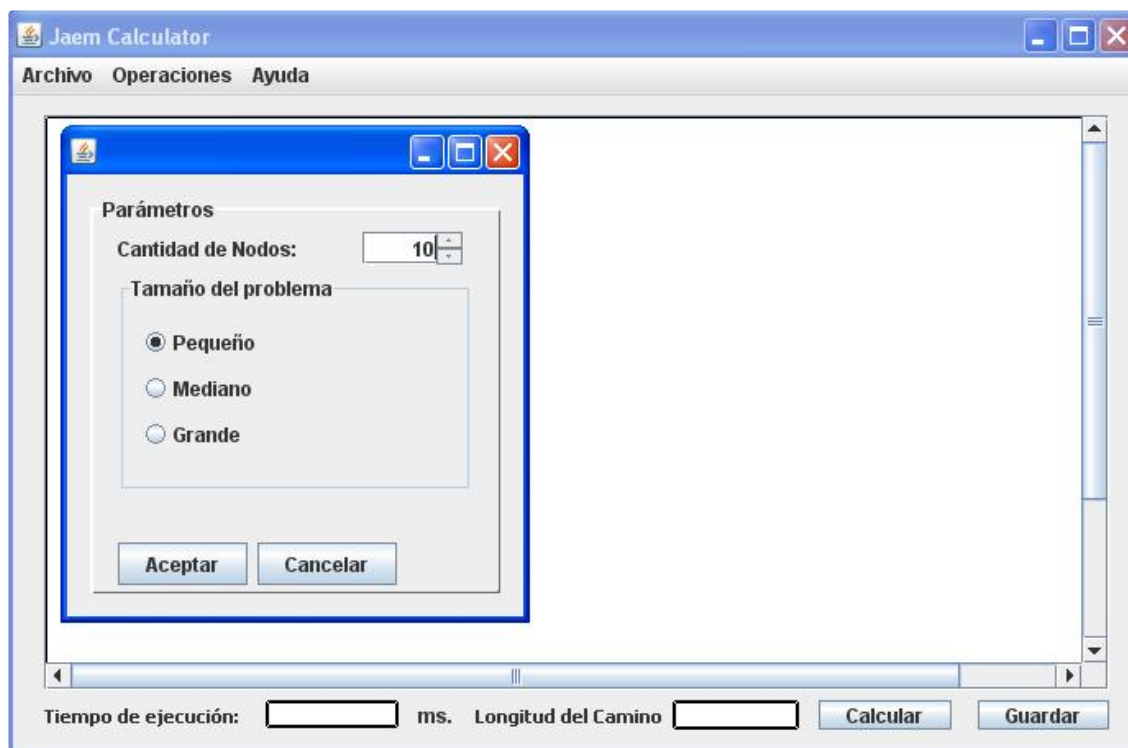


Figura 11. Interfaz de Usuario “Mover nodo”

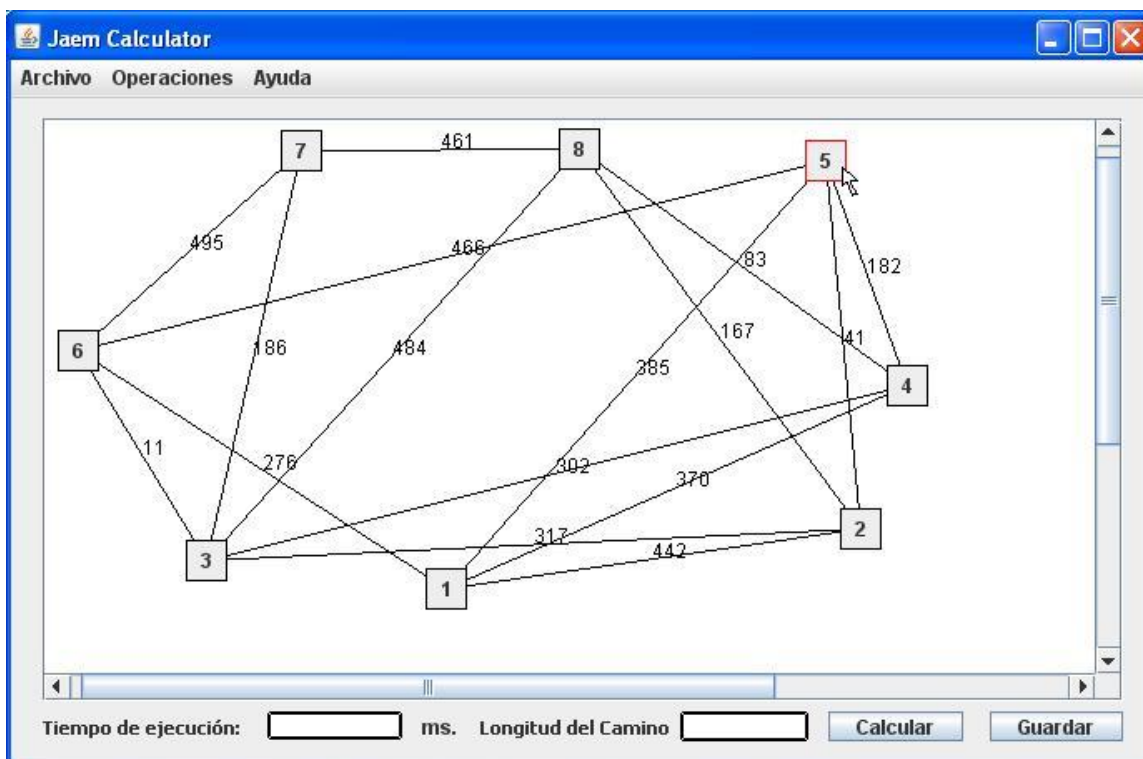




Figura 12. Interfaz de Usuario “Construir AEM”

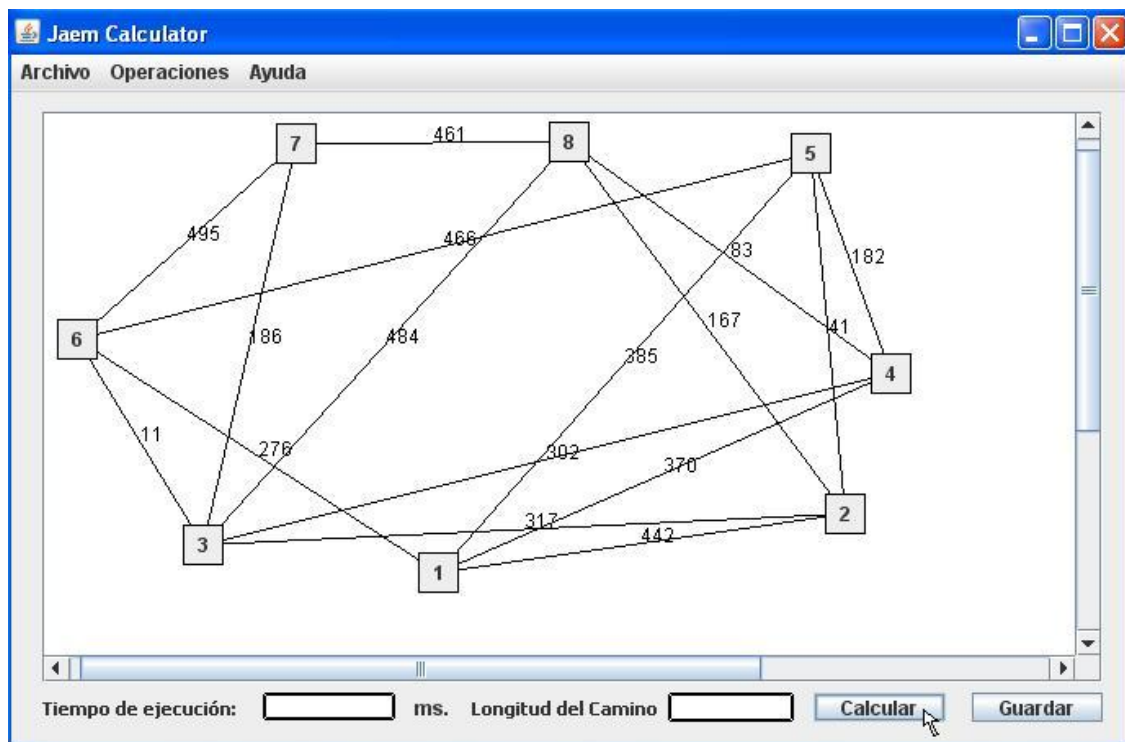
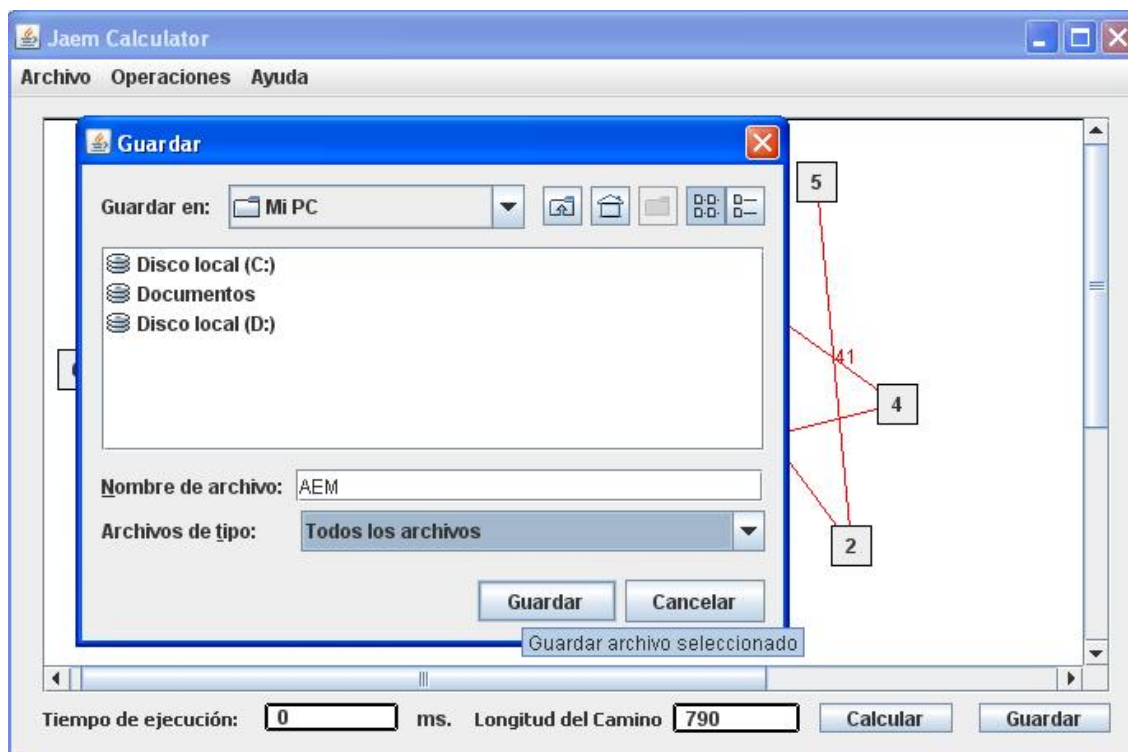


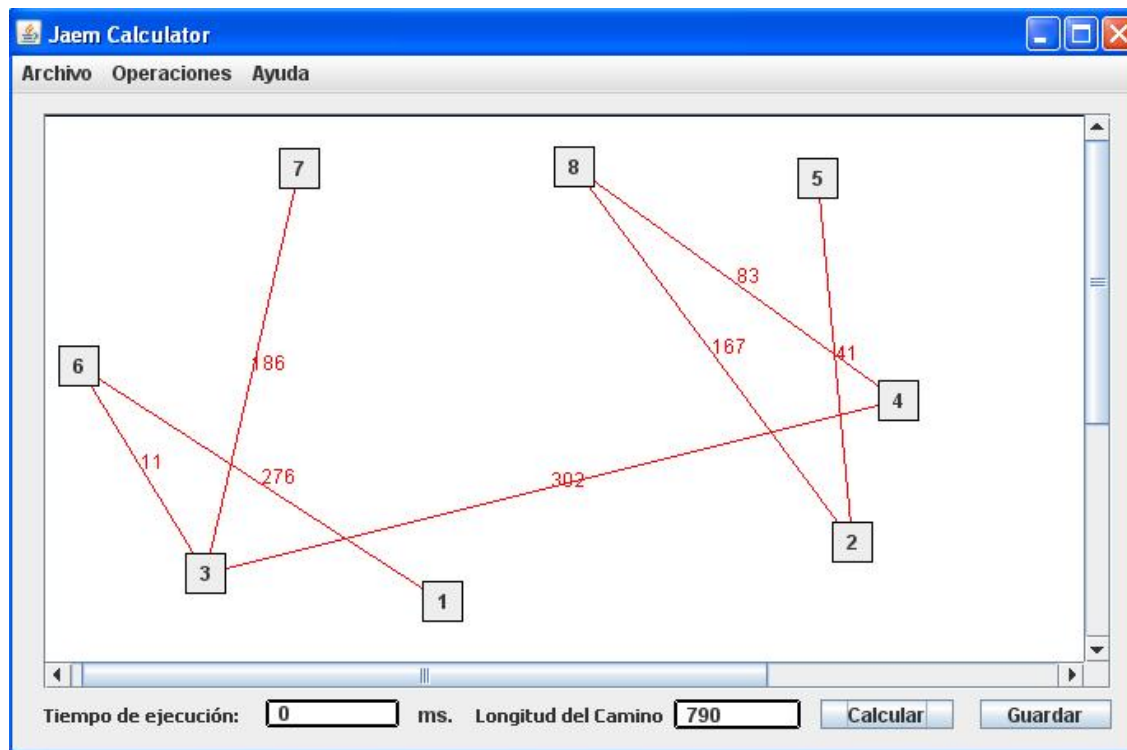
Figura 13. Interfaz de Usuario “Guardar solución”





3.6.2 Formato de salida de los reportes

Figura 14. Interfaz de usuario “Mostrar AEM”



3.6.3 Ayuda

El sistema contará con un manual de usuario que les guíara como utilizar el mismo, el cual estará en formato pdf.

3.7 Diagrama de Secuencia

Figura 15. Diagrama de secuencia “Insertar & editar nodo”.

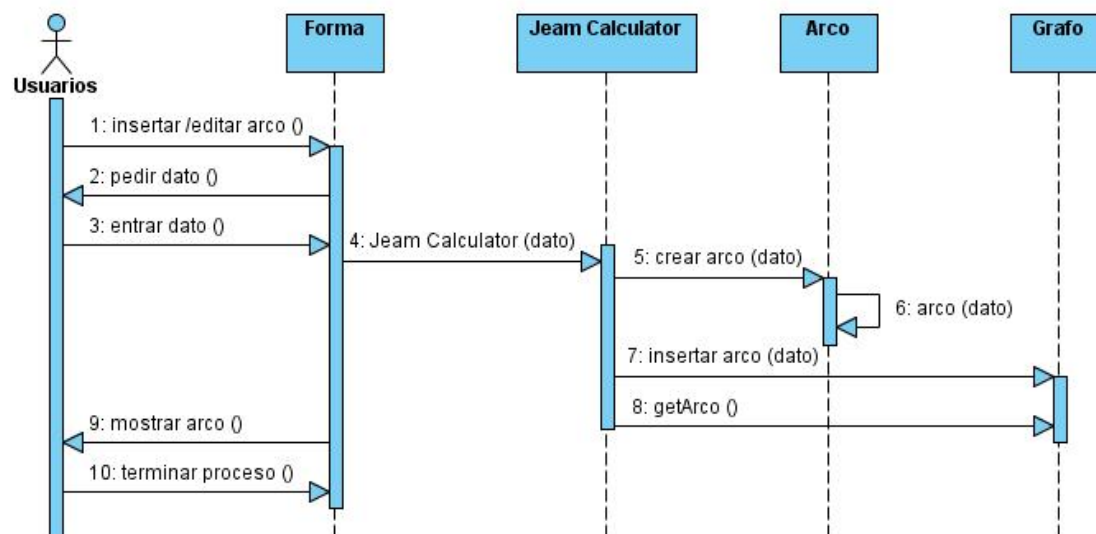


Figura 16. Diagrama de secuencia “Eliminar nodo”.

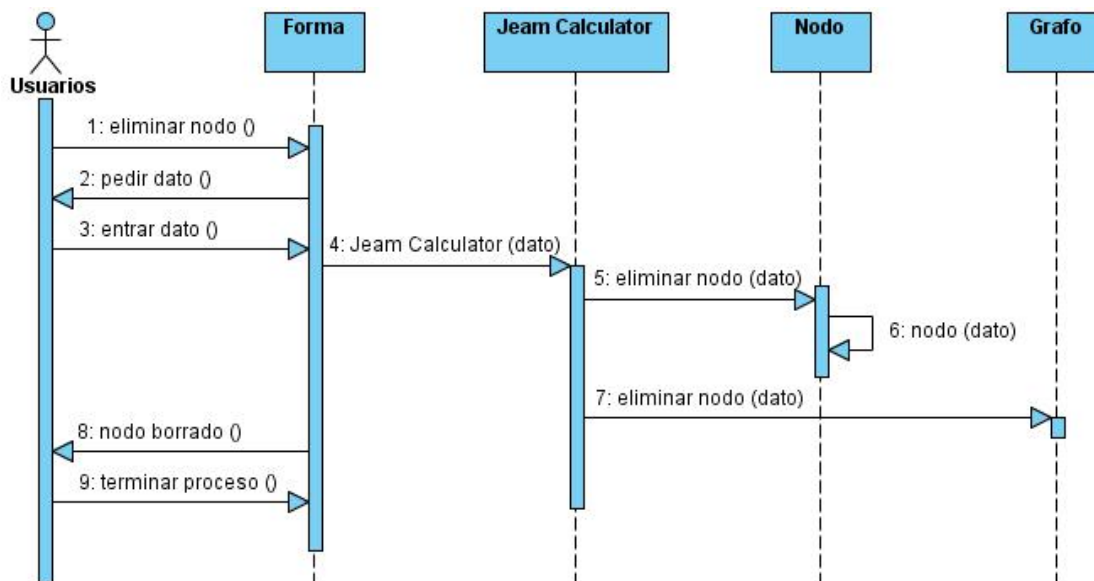


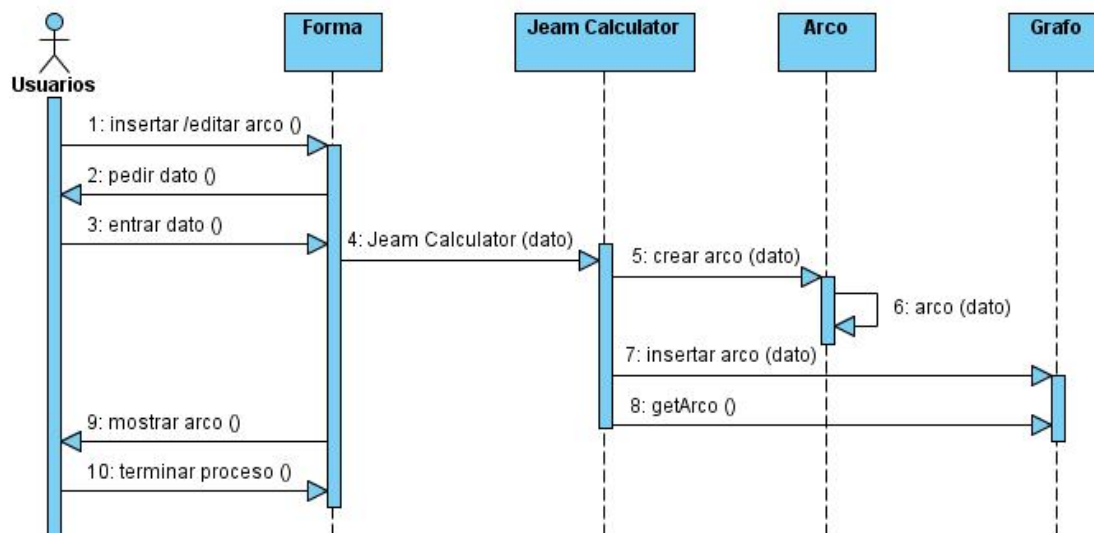
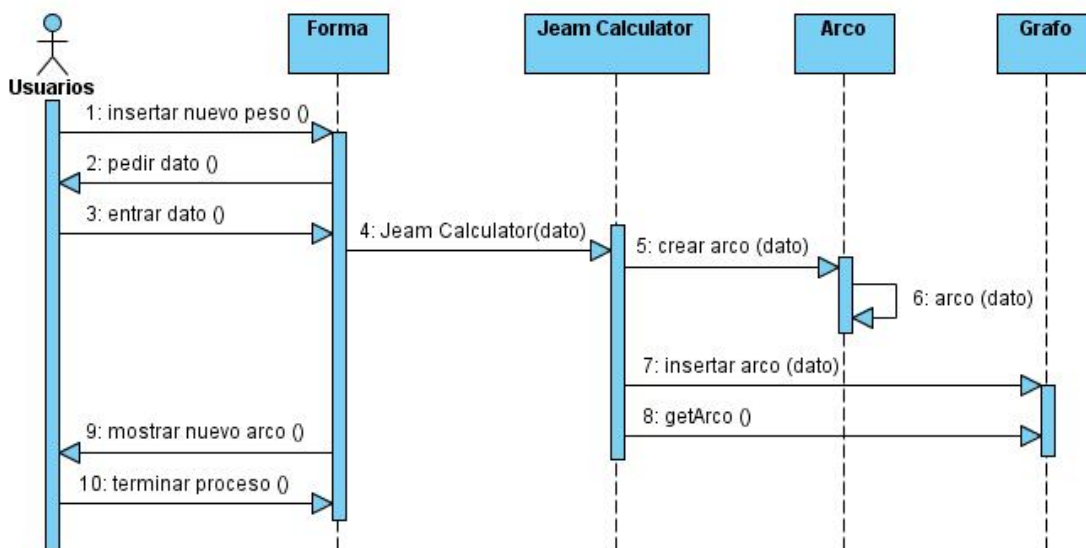
Figura 17. Diagrama de secuencia “Insertar & editar arco”.**Figura 18. Diagrama de secuencia “Cambiar peso”.**

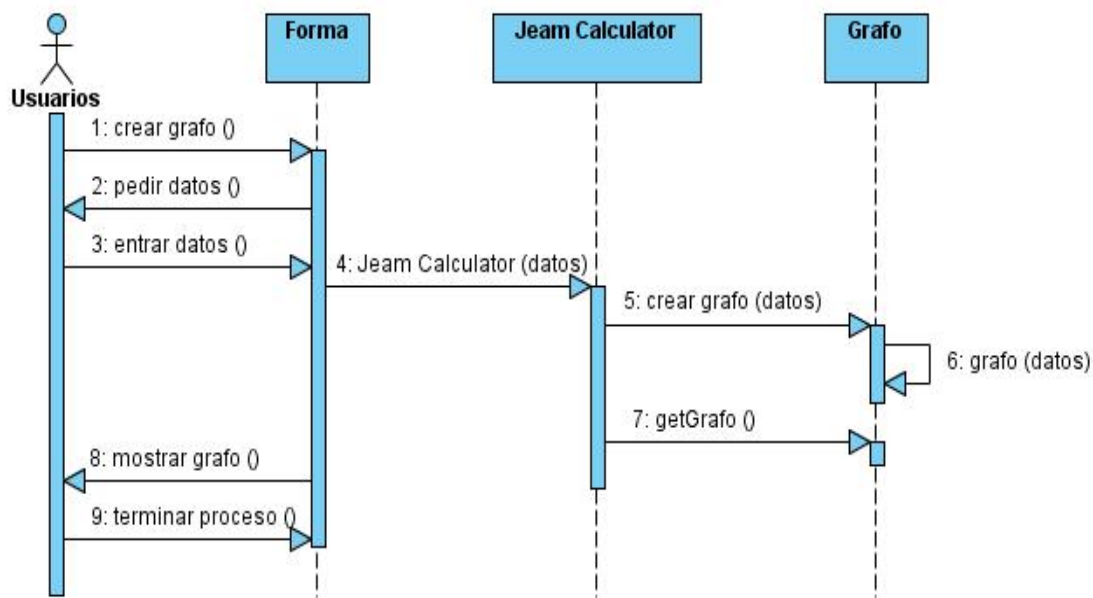
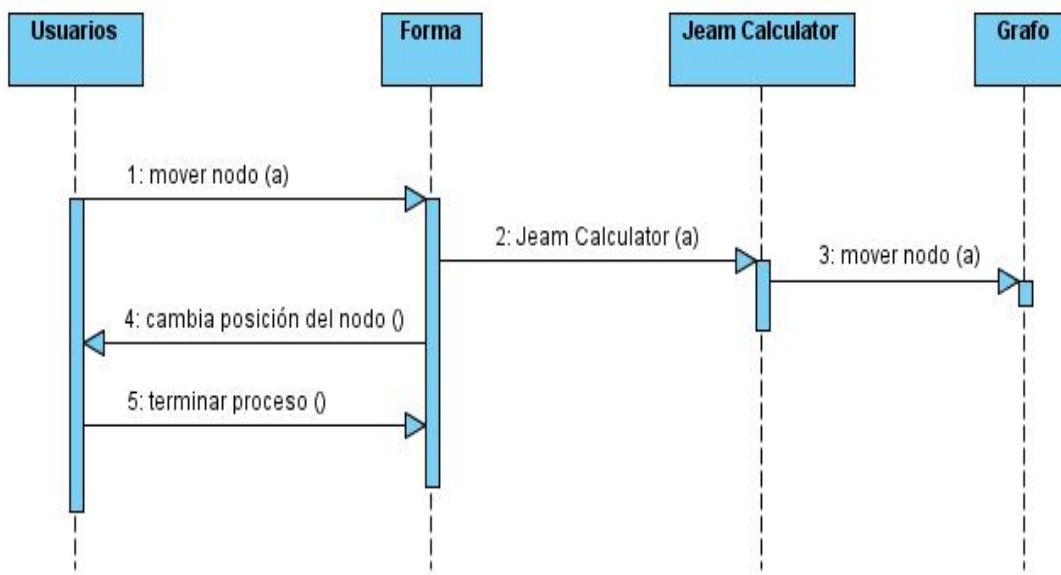
Figura 19. Diagrama de secuencia “Crear grafo”.**Figura 20. Diagrama de secuencia “Mover nodo”.**

Figura 21. Diagrama de secuencia “Construir AEM & Mostrar solución”.

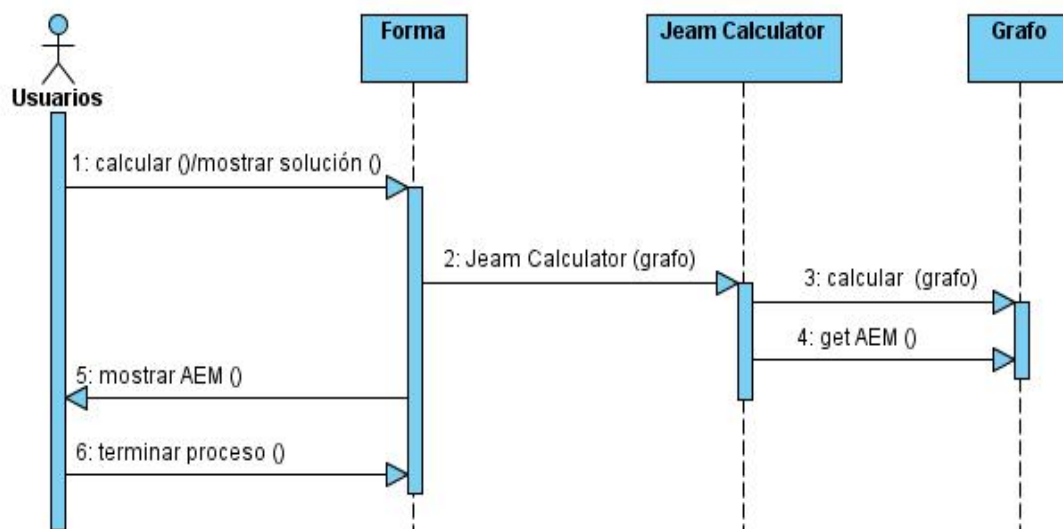
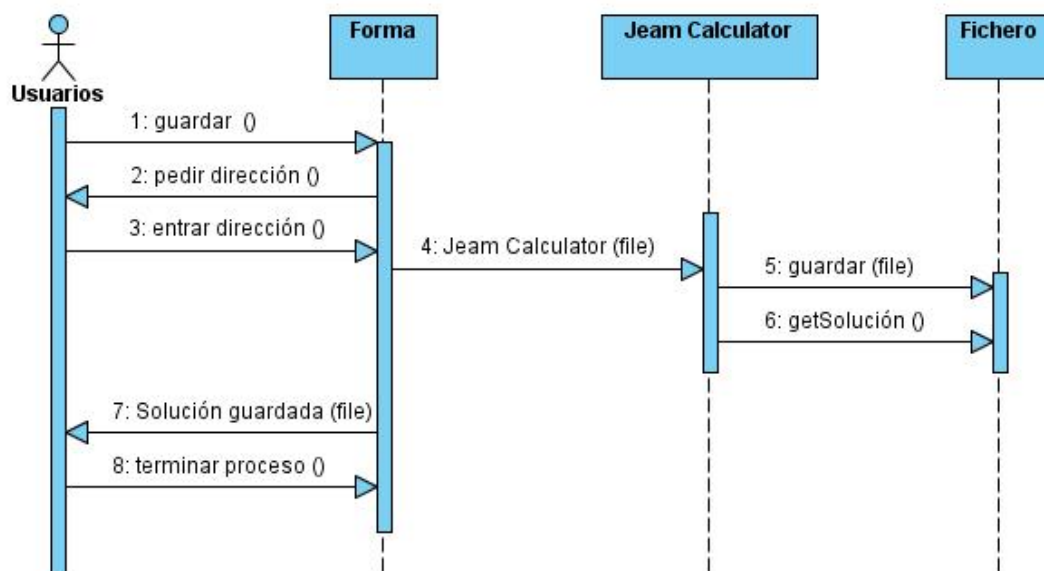
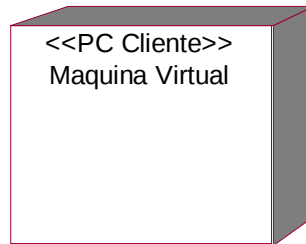


Figura 22. Diagrama de secuencia “Guardar solución”.



3.8 Diagrama de despliegue

Figura 23. Diagrama de despliegue

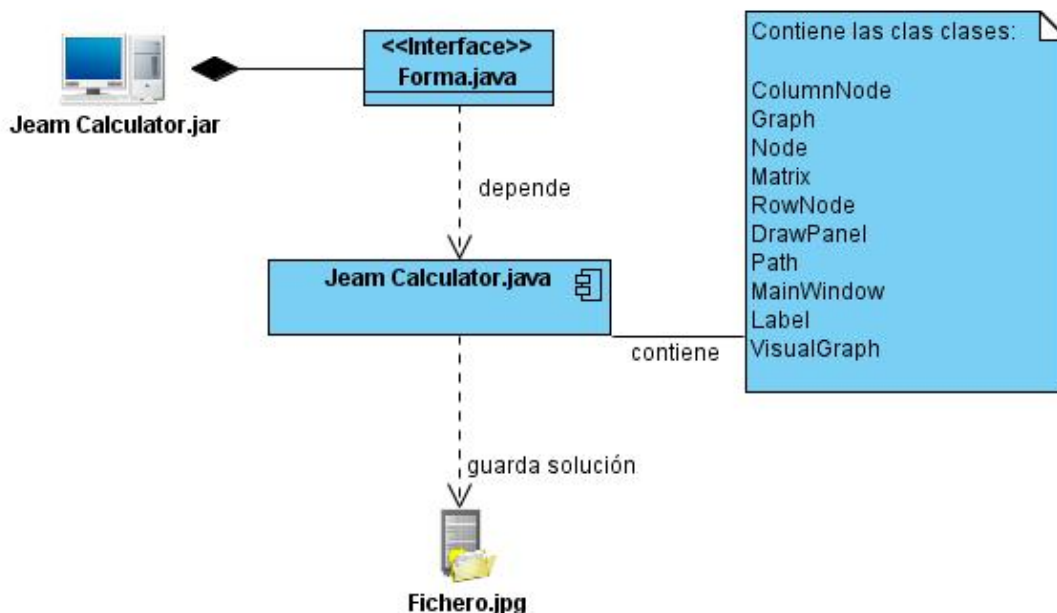


3.9 Diagrama de Componentes

Partiendo de la agrupación de los elementos del diseño a implementar por componente, se desarrollaron los diagramas de componentes con el propósito de modelar el sistema, para mostrar la organización y las dependencias entre los componentes.

A partir del diagrama de componentes se pudo cristalizar lo definido en los flujos anteriores, así como implementar el sistema propuesto durante la etapa de construcción.

Figura 24. Diagrama de componente.





3.10 Conclusiones

En este capítulo se ofrecen aspectos que dentro de la metodología de ingeniería de software empleada, brindan informaciones específicas para el correcto entendimiento de la solución propuesta a partir del análisis. Se definieron el actor del sistema, el Diagrama de Casos de Uso el cual representa las funcionalidades del sistema, el Diagrama de Clases que representa la estructura interna del sistema, los diagramas de secuencia con los que se logra delinear el orden de los procesos para la ejecución de cada caso de uso. Se representó la arquitectura física del sistema con el modelo de despliegue, compuesto por un nodo que representa una computadora cliente con la máquina virtual de java y la aplicación propuesta.



Capítulo 4 Estudio de factibilidad

4.1 Introducción

Para el estudio de factibilidad de este proyecto se utilizará la **Metodología Costo Efectividad (Beneficio)**, la cual plantea que la conveniencia de la ejecución de un proyecto se determina por la observación conjunta de dos factores:

- 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 la capacidad del proyecto para satisfacer la necesidad, solucionar el problema o lograr el objetivo para 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 de cumplimiento del objetivo).

Además se presenta la Valoración de Sostenibilidad del producto según su impacto administrativo, socio – humanista, ambiental y tecnológico.

4.2 Efectos Económicos

- Efectos directos
- Efectos indirectos
- Efectos externos
- Intangibles

Efectos directos:

➤ POSITIVOS:

- Se permitirá resolver problemas de AEM medianamente grandes, imposible de hacer a lápiz y papel.
- Se ganará en velocidad de computo agilizándose estos mediante la ayuda del ordenador mejorando grandemente el tiempo para solucionar un problema.
- Se dejará de utilizar papel y lápiz para resolver los problemas. Así como herramientas informáticas que a pesar que resuelven este tipo de



problemas, tienen varias limitaciones las cuales fueron mencionadas en capítulos anteriores.

➤ **NEGATIVOS:**

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

Efecto indirecto:

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

Efecto externo:

- Se contará con una herramienta disponible que apoyará a los administrativos a mejorar la toma de decisiones.

Intangibles:

- En la valoración económica siempre hay elementos perceptibles por una comunidad como perjuicio o beneficio, pero al momento de ponderar en unidades monetarias esto resulta difícil o prácticamente imposible.

A fin de medir con precisión los efectos, deberán considerarse tres situaciones:

• **SITUACIÓN SIN PROYECTO**

Para dar solución al PAEM sin proyecto debemos seguir los siguientes pasos:

1. Construcción del modelo.

En la construcción del modelo se deben seguir los siguientes pasos:

- a) Construcción y edición de los nodos.
- b) Construcción y edición de los arcos.

Nota: Una vez contruidos y editados los nodos y los arcos queda conformado el grafo.

2. Derivar una solución del modelo.

Para la solución del modelo debemos seguir los siguientes pasos:

- a) Se selecciona de manera arbitraria cualquier nodo y se conecta.



- b) Se identifica el nodo no conectado más cercano a un nodo no conectado y se conectan estos dos nodos. Este paso se repite hasta que se hayan conectado todos los nodos
- c) Empates: los empates para el nodo mas cercano distinto o para el nodo conectado mas cercano ,se pueden romper en forma arbitraria y el algoritmo todavía debe llevar a solución optima
- d) Verificar si estamos en presencia de una solución óptima.

Nota: La solución se hace más trabajosa según la complejidad del problema (grande, mediano, pequeño) y la cantidad de nodos.

- **OPTIMIZADA SIN PROYECTO**

La solución optimizada sin proyecto cumple estrictamente con lo antes reflejado en la situación sin proyecto, debido a que estos pasos responden a un algoritmo matemático optimizado.

- **SITUACIÓN CON PROYECTO**

Para llevar acabo la solución del modelo matemático con proyecto debemos seguir los siguientes pasos:

1. Construcción del modelo.

En la construcción del modelo se deben seguir los siguientes pasos:

- a) Construcción y edición de los nodos.
- b) Construcción y edición de los arcos.

Además cuenta con la opción de generar un grafo de forma automática.

Nota: Una vez contruidos y editados los nodos y los arcos queda conformado el grafo, igual queda conformado el grafo si se genera de forma automática.

2. Solucionar modelo.

- a) En el menú, seleccionar “Calcular”.

4.3 Beneficios y Costos Intangibles en el proyecto

COSTOS:

- Resistencia al cambio.



BENEFICIOS:

- Mejora en la calidad de la información por la integridad, oportunidad de la información y confiabilidad.
- Mayor comodidad de los usuarios
- Mejor imagen de la institución.

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 [Dra. Ana M. Gracia Pérez, UCLV].

Para la elaboración de la ficha se consideran los siguientes elementos de costo, desglosados en moneda libremente convertible y moneda nacional.

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: \$ 60.78.
5. Materiales directos: No procede.

Total: \$ 60.78.

- **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.

Costos en Moneda Nacional:

- **Costos Directos.**

1. Salario del personal que laborará en el proyecto: \$100.00.
2. El 12% del total de gastos por salarios se dedica a la seguridad social:
No procede.
3. El 0.09% de salario total, por concepto de vacaciones a acumular: No
procede.
4. Gasto por consumo de energía eléctrica: \$ 5.94.
5. Gastos en llamadas telefónicas: No procede.
6. Gastos administrativos: No procede.

- **Costos Indirectos.**

1. Know How: \$ 108,75.

Total: \$ 214.69.

Como se hizo referencia anteriormente, la técnica seleccionada para evaluar la factibilidad del proyecto es la Metodología Costo-Efectividad. Dentro de esta metodología la técnica de punto de equilibrio aplicable a proyectos donde los beneficios tangibles no son evidentes el análisis se basa exclusivamente en los costos. Para esta técnica es imprescindible definir una variable discreta que haga variar los costos. Teniendo en cuenta que el costo para este proyecto es despreciable, se toma como costo el tiempo en horas empleado para resolver un PAEM y la cantidad de nodos sería la complejidad del problema para lo cual se tienen los siguientes valores.

Problema pequeño (Pp): $f(x)/2 > P_g > 0$

$f(x)$ = aristas de un arco fuertemente conexo.

Valores de la variable (Solución manual):

- 1 Pp de hasta 10 nodos. (420000 milisegundos)
- 1 Pp de hasta 15 nodos. (780000 milisegundos)
- 1 Pp de hasta 20 nodos. (1200000 milisegundos)



- 1 Pp de hasta 25 nodos. (1740000 milisegundos)

Valores de la variable (Solución con el software):

- 1 Pp de hasta 50 nodos. (63 milisegundos)
- 1 Pp de hasta 100 nodos. (890 milisegundos)
- 1 Pp de hasta 150 nodos. (5062 milisegundos)
- 1 Pp de hasta 200 nodos. (16859 milisegundos)

Problema pequeño (Pp): $3/4f(x) > P_g \geq f(x)/2$

$f(x)$ = aristas de un arco fuertemente conexo.

Valores de la variable (Solución manual):

- 1 Pm de hasta 10 nodos. (960000 milisegundos)
- 1 Pm de hasta 15 nodos. (1440000 milisegundos)
- 1 Pm de hasta 20 nodos. (2400000 milisegundos)
- 1 Pm de hasta 25 nodos. (3600000 milisegundos)

Valores de la variable (Solución con el software):

- 1 Pm de hasta 50 nodos. (125 milisegundos)
- 1 Pm de hasta 100 nodos. (1875 milisegundos)
- 1 Pm de hasta 150 nodos. (10687 milisegundos)
- 1 Pm de hasta 200 nodos. (33968 milisegundos)

Problema grande (Pg) : $f(x) \geq P_g > 3/4f(x)$

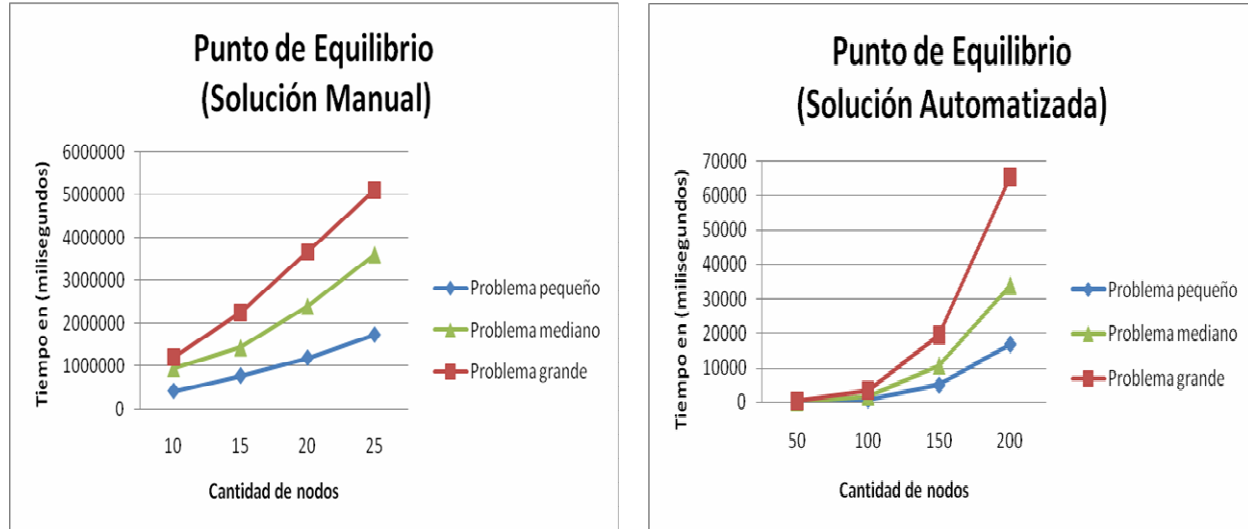
$f(x)$ = aristas de un arco fuertemente conexo.

Valores de la variable (Solución manual):

- 1 Pg de hasta 10 nodos. (1200000 milisegundos)
- 1 Pg de hasta 15 nodos. (2250000 milisegundos)
- 1 Pg de hasta 20 nodos. (3660000 milisegundos)
- 1 Pg de hasta 25 nodos. (5100000 milisegundos)

Valores de la variable (Solución con el software):

- 1 Pg de hasta 50 nodos. (230 milisegundos)
- 1 Pg de hasta 100 nodos. (3398 milisegundos)
- 1 Pg de hasta 150 nodos. (19446 milisegundos)
- 1 Pg de hasta 200 nodos. (65360 milisegundos)

Figura 25. Punto de equilibrio.

Punto de Equilibrio

Teniendo en cuenta los resultados reflejados en las gráficas en cuanto al Punto de Equilibrio queda demostrada la factibilidad del sistema, evidenciado por la relación entre la complejidad del problema (cantidad de nodos) y el tiempo que demora la solución del mismo de forma manual y automatizada.

4.5 Valoración de Sostenibilidad del Producto según su impacto administrativo, socio-humanista, ambiental y tecnológico.

La sociedad aspira a un mundo mejor en los aspectos económico, social y ambiental. El concepto que expresa este deseo es el de desarrollo sostenible.

El desarrollo sustentable es un proceso de cambio progresivo en la calidad de vida del ser humano, que lo coloca como centro y sujeto primordial del desarrollo, por medio del crecimiento económico con equidad social y la transformación de los métodos de producción y de los patrones de consumo y que se sustenta en el equilibrio ecológico y el soporte vital de la región. Este proceso implica el respeto a la diversidad étnica y cultural regional, nacional y local, así como el fortalecimiento y la plena participación ciudadana en convivencia pacífica con la naturaleza, sin comprometer y garantizar la calidad de vida de las generaciones futuras.



De aquí que hayamos hecho una valoración de sostenibilidad del sistema a obtener, de modo que quedaran bien definidas las perspectivas del mismo hacia ese desarrollo sostenible el cual se persigue:

Dimensión Administrativa

La confección de un producto informático, muchas veces, trae consigo un impacto administrativo, el cual no es más que cambios producidos en la administración de una entidad.

Los beneficios obtenidos mediante este trabajo son positivos en gran medida, no en el sentido monetario directamente, sino en cuanto a que la entidad no tuvo que mejorar el equipamiento dentro de los requerimientos y por este motivo no tuvieron gastos en ese sentido. El lenguaje utilizado en el sistema fue “Java”, el cual es un software libre.

Además, permite ahorro de recursos humanos e insumos, tales como materiales de oficinas, etc.

El gasto en que se incurre para su aplicación es mínimo pues el ISMM ya tenía creada la infraestructura o sea, los recursos informáticos sobre la que descansa el proyecto. Al resolver PAEM con seguridad propicia una disminución de errores al dar solución a este tipo de problemas, y de gastos por mala planificación, etc., ayudando considerablemente en la toma de decisiones administrativas. El sistema más que minimizar la mano de obra, la optimiza, por lo que disminuye considerablemente el tiempo en que normalmente se resolvían estos problemas.

Genera ingresos por cuestiones de ahorro de recursos, disminuye el por ciento de errores, lo que permite realizar una buena planificación. Permite una mejor distribución de los recursos humanos, además de implicar una mayor calidad en los servicios que los mismos prestan.

Dimensión Socio-Humanista

Cuando se hace alusión al impacto social, se hace referencia a cualquier alteración o cambio que se produce en las diferentes dimensiones de la realidad social de las comunidades ubicadas dentro del área de influencia del proyecto a desarrollar.



Con este producto se logra que los estudiantes y profesionales del ISMM se sientan cómodos, trabajando con mayor eficiencia y rapidez al tomar decisiones a través de la solución de PAEM.

Además, para un mejor desempeño de los usuarios con el sistema, se previó un sistema de ayuda y preparación del personal que será actor del sistema y tiene como objetivo también apalear el rechazo al cambio que puedan hacer los usuarios ante el sistema.

Al elevar la efectividad de la toma de decisiones mediante la solución de problemas de árbol de expansión mínima en el Instituto Superior Minero Metalúrgico “Dr. Antonio Núñez Jiménez (ISMM)”, mejora la calidad de vida de la sociedad y contribuye a la formación de responsabilidad, solidaridad y compromiso con el desarrollo sostenible del entorno.

Mejora la cultura profesional de los estudiantes y profesionales relacionados con la producción, obligándolos a superarse a través de cursos que avalen su capacidad en la explotación del Producto Informático.

El sistema no cierra ni genera empleos, pero en cierta medida minimiza la mano de obra, aunque la directiva principal es que optimiza la mano de obra.

Dimensión Ambiental

Se puede definir como cualquier alteración que se produzca en el medio ambiente al realizarse un proyecto o cualquier actividad humana.

Se ahorran recursos que generan daños al medio ambiente, como el papel; al disminuir el por ciento de errores, permite una buena planificación, lo que trae consigo una mejor administración de recursos tales como el petróleo, neumáticos, cableado de cobre, etc., los cuales son obtenidos mediante métodos mineros que afectan directamente al medio ambiente, esto incide favorablemente hasta en el proceso de reforestación de las áreas deforestadas; permite controlar más eficientemente la contaminación ambiental, teniendo un impacto indirecto favorable al medio ambiente.

No provoca daños físicos por movimientos repetitivos al teclado, mouse, etc. No usa colores agresivos a la vista, ni genera contaminación por ruido. Favorece



una interfaz agradable al entorno de usuario, además de personalizarla a las funciones y/o responsabilidades que el mismo tenga.

El producto o parte de el puede ser utilizado en otro producto, ya que es un software libre el cual esta a la disposición de todos.

Dimensión Tecnológica

Aunque el usuario tiene suficiente preparación como para trabajar con el producto, “la toma de decisiones mediante la solución de PAEM” incluye capacitación del personal, propiciando la superación profesional.

La organización cuenta con la infraestructura necesaria para implantar el Producto Informático. El producto se ha diseñado para que la escalabilidad del sistema a través de la coexistencia con otros productos informáticos, más que una limitante sea una virtud del mismo.

4.6 Conclusiones

En este capítulo se realizó el estudio de factibilidad mediante la Metodología Costo Efectividad (Beneficio), se analizaron los efectos económicos, los beneficios y costos intangibles, se calculó el costo de ejecución del proyecto mediante la ficha de costo arrojando como resultado \$60.78 CUC y \$214.69 MN ,y se realizó una Valoración de Sostenibilidad del producto según su impacto administrativo, socio – humanista, ambiental y tecnológico demostrándose la conveniencia de la elaboración del sistema .



Conclusiones

La investigación realizada parte del problema de cómo elevar la efectividad de la toma de decisiones mediante la solución de PAEM en el Instituto Superior Minero Metalúrgico “Dr. Antonio Núñez Jiménez (ISMM).

Al hacerse un estudio bibliográficos de los referentes teóricos sobre el problema en cuestión y diagnosticar la situación actual que presentan los sistemas informáticos. Se considera necesaria la elaboración de una propuesta de un sistema informático para la solución de problemas de árbol de expansión mínima, que utilice pocos recursos computacionales, sea multiplataforma, brinde al usuario seguridad y rapidez en los tiempos de respuesta, y libre de costo alguno.

Por otra parte se cuenta con una herramienta multiplataforma ampliando así su compatibilidad con otros sistemas operativos.

El sistema fue implementado sobre herramientas patentizadas por licencia GNU/GPL, por lo que garantiza su uso sin costos adicionales de licencias.

Constituye un material didáctico y metodológico para los profesores que imparten la disciplina de Investigación de Operaciones, así como una herramienta mas para la toma de decisiones a través de problemas árbol de expansión mínima.



Recomendaciones

Con el objetivo de perfeccionar y beneficiarnos con el sistema se recomienda:

1. Proponer el sistema a los profesores que imparten la asignatura de investigación de operaciones, como una herramienta más para la enseñanza de los problemas de árbol de expansión mínima.
2. Proponer el sistema a los profesionales de diferentes departamentos del ISMM, que trabajan con problemas de redes, como una herramienta más para mejorar la toma de decisiones.
3. Realizar un estudio más profundo de este sistema en vista a perfeccionarlo en nuevas versiones del software.
4. Crear la ayuda y documentación en línea, para que el usuario cuente con una amplia documentación.
5. Realizar las pruebas concernientes a la compatibilidad de la aplicación con Sistemas Operativos que soporten Java.



Referencias bibliográficas

- [1]. Schein EH. Process consultation. Cambridge: Addison-Wesley Publishing Company, 1988. pp. 81.
- [2]. Ponjuán Dante G. Gestión de información en las organizaciones: principios, conceptos y aplicaciones. Santiago de Chile: Cecapi, 1998. pp. XXII.



Bibliografía

1. [Matemática Discreta Tomo II.](#)
2. [Introducción a la Investigación de Operaciones Tomo II.](#)
3. [Dr. Sc. Ramon Rodríguez Betancourt, C.D.M.A.G., Programación Matemática para Economista Tomo1, 1992, La Habana.](#)
4. [Jacobson, G.B., James Rumbaugh., El Proceso Unificado de Desarrollo Software. 1999: Addison Wesley.](#)
5. [Herrera et al. 1994. Algoritmos Genéticos: Fundamentos, extensiones y aplicaciones. Universidad de Granada. España.](#)
6. [Fonseca, Carlos & Fleming, Peter. 1995. Multiobjective optimization and multiple constraint handling with evolutionary algorithms. USA.](#)
7. [Goldberg, David. 1997. Genetic algorithms in search, Optimization & Machine Learning. Addison-Wesley Co. Inc. Reading. Massachusetts, USA.](#)
8. [Michalewicz, Zbignew. 1994. Genetic Algorithms + Data Structures = Evolution Programs. Springer-Verlag. Alemania.](#)
9. [García R., Agustín. 2000. Un algoritmo genético para el Problema del Árbol de Expansión Mínima. Paper de Informática y Ciencias de la Computación. México.](#)
10. [Complejidad algorítmica de Prim en](#)
<http://www.dma.fi.upm.es/gregorio/grafos/primkruskal/prim/prim.html>
11. [Lenguaje de programación C en](#)
[http://es.wikipedia.org/wiki/Lenguaje_de_programaci%C3%B3n_C.](http://es.wikipedia.org/wiki/Lenguaje_de_programaci%C3%B3n_C)
12. [C++ en](#) [http://es.wikipedia.org/wiki/C%2B%2B.](http://es.wikipedia.org/wiki/C%2B%2B)
13. [Delphi en](#) [http://es.wikipedia.org/wiki/Delphi.](http://es.wikipedia.org/wiki/Delphi)
14. [Lenguaje de programación Java en](#)
[http://es.wikipedia.org/wiki/Lenguaje_de_programaci%C3%B3n_Java.](http://es.wikipedia.org/wiki/Lenguaje_de_programaci%C3%B3n_Java)
15. [Java Básico.pdf.](#)
16. [Tutor Java.pdf.](#)
17. [Michelle Waite. Data Structures and Algorithms in Java.pdf.](#)



18. [Martín Pérez Mariñán. Estructuras de datos en Java: Collections.pdf.](#)
19. [Robert Lafore. Data Structures and Algorithms. pdf](#)
20. [B.A.Sc., M.A.Sc., Ph.D., P.Eng. Bruno R. Preiss. Data Structures and](#)
21. [Algorithms with Object-Oriented Design Patterns in Java.pdf](#)
22. [Victor Marzal. Curso de estructuras de datos.pdf](#)
23. [Jonathan Knudsen, Patrick Niemeyer. Learning Java, 3rd Edition.pdf](#)
24. [El lenguaje de Programación Java.pdf](#)
25. [Curso de Programación Avanzada en Java en Español.pdf](#)
26. [Mark Allen Weiss. Estructuras de datos en Java. Volumen I primera y](#)
[segunda parte.pdf.](#)
27. [Mark Allen Weiss. Estructuras de datos en Java. Volumen II tercera](#)
[parte.pdf.](#)
28. [Mark Allen Weiss. Estructuras de datos en Java. Volumen III cuarta y](#)
[quinta parte.pdf.](#)
29. [Dra. Ana Ma. Gracia Pérez, UCLV .Ficha De Costo de un Producto](#)
[Informático.pdf.](#)
30. [Dra.Rita Valoración de Sostenibilidad del Producto según su impacto](#)
[administrativo, socio – humanista, ambiental y tecnológico.pdf.](#)



Glosario de términos

Vértice: Es usualmente llamado nodo, o punto. Es usualmente representado por un círculo. En las redes de transporte, estos deberían ser las localidades o las ciudades en un mapa.

Arista: Es usualmente llamado arco o flecha. Este podría ser directo o indirecto. La cabeza es el destino, y la cola el origen. La cabeza y la cola son nodos que pueden estar tanto al origen como al final. En las redes de transporte, los arcos podrían ser los caminos, los canales de navegación en un río, o los patrones de vuelo de un avión. Los arcos proporcionan la conectividad entre los nodos. Una calle de una sola dirección podría ser representada por un arco, mientras que una calle de dos direcciones podría representada por un arco sin dirección o por dos arcos que apuntan a direcciones opuestas.

Diagrama de clases (*class diagram*): diagrama de objetos que describe las clases en forma de esquema, patrón o plantilla, de muchas de las posibles instancias de datos.

GNU/GPL: Conjunto de programas desarrollados por la Fundación por el Software Libre; es de uso libre.

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.

TIC (Tecnologías de la Información y la Comunicación): Esta expresión engloba el conjunto de tecnologías que conforman la sociedad de la información: informática, Internet, multimedia, etcétera, y los sistemas de telecomunicaciones que permiten su distribución.

UML: Es el lenguaje de modelado de sistemas de software más conocido en la actualidad; es el estándar internacional aprobado por la OMG (Object Management Group). UML son un grupo de especificaciones de notación orientadas a Objeto, las cuales están compuesta por distintos diagramas, que representan las diferentes etapas del desarrollo de un proyecto de software.



Anexo 1 Descripción de los casos de uso del sistema

Tabla 5. Descripción del caso de uso “Insertar Nodo”.

Nombre del caso de uso	Insertar Nodo
Actores	Usuario(inicia)
Resumen	Este caso de uso lo inicia el usuario cuando decide introducir un nodo al sistema
Referencias	R1
Precondiciones	-
Poscondiciones	Queda insertado el nodo en el sistema listo para ser procesado
Requisitos especiales	Se debe validar que sean introducidos todos los datos

Tabla 6. Descripción del caso de uso “Editar Nodo”.

Nombre del caso de uso	Editar Nodo
Actores	Usuario(inicia)
Resumen	Este caso de uso lo inicia el usuario editando el nodo insertado en el sistema
Referencias	R1.1
Precondiciones	Debe haber insertado un nuevo nodo en el sistema
Poscondiciones	Queda editado el nodo
Requisitos especiales	Se debe validar que hallan introducidos todos los datos anteriormente



Tabla 7. Descripción del caso de uso “Eliminar nodo”

Nombre del caso de uso	Eliminar nodo
Actores	Usuario(inicia)
Resumen	Este caso de uso lo inicia el usuario cuando decide eliminar un nodo del sistema.
Referencias	R2
Precondiciones	Debe haber insertado al menos un nodo en el sistema.
Poscondiciones	Queda eliminado el nodo del sistema.
Requisitos especiales	Se debe validar que sean eliminados todos los datos

Tabla 8. Descripción del caso de uso “Insertar Arco”

Nombre del caso de uso	Insertar Arco.
Actores	Usuario (inicia).
Resumen	Este caso de uso lo inicia el usuario cuando decide conectar dos nodos.
Referencias	R3
Precondiciones	Deben existir al menos dos nodos.
Poscondiciones	Queda insertado el arco en el sistema listo para ser procesado.
Requisitos especiales	Se debe validar que sean introducidos todos los datos.

Tabla 9. Descripción del caso de uso “Editar Arco”.

Nombre del caso de uso	Editar Arco.
Actores	Usuario (inicia).
Resumen	Este caso de uso lo inicia el usuario al asignarle un peso al arco



	insertado en el sistema.
Referencias	R3.1
Precondiciones	Debe haber insertado un nuevo arco en el sistema.
Poscondiciones	Queda editado el arco.
Requisitos especiales	Se debe validar que sean introducidos correctamente los datos.

Tabla 10. Descripción del caso de uso “Cambiar peso”

Nombre del caso de uso	Cambiar peso.
Actores	Usuario (inicia).
Resumen	Caso de uso iniciado por el usuario cuando se ve en la necesidad de cambiar el valor de un arco determinado.
Referencias	R4
Precondiciones	Debe existir el arco en el sistema.
Poscondiciones	Cambia el valor del arco.
Requisitos especiales	Se debe validar que sean introducidos correctamente los datos



Tabla 11. Descripción del caso de uso “Crear grafo”

Nombre del caso de uso	Crear grafo.
Actores	Usuario (inicia).
Resumen	Caso de uso iniciado por el usuario cuando se ve en la necesidad de que el sistema genere un grafo con características determinadas por el propio usuario (cantidad de nodos, tamaño del problema (pequeño, mediano, grande)).
Referencias	R5
Precondiciones	-
Poscondiciones	Se crea un grafo.
Requisitos especiales	Se debe validar que sean introducidos correctamente los datos

Tabla 12. Descripción del caso de uso “Mover nodo”

Nombre del caso de uso	Mover nodo.
Actores	Usuario (inicia).
Resumen	El usuario lo inicia cuando le es necesario cambiar la posición de un nodo determinado.
Referencias	R6
Precondiciones	Debe existir el nodo.
Poscondiciones	Cambia la posición del nodo seleccionado.
Requisitos especiales	Se debe tener en cuenta que no afecte la imagen de otros nodos o arcos.

Tabla 13. Descripción del caso de uso “Mostrar Solución”.

Nombre del	Mostrar solución
-------------------	------------------



caso de uso	
Actores	Usuario(inicia)
Resumen	Este caso de uso lo inicia el usuario cuando decide ver la respuesta.
Referencias	R8
Precondiciones	Debe haber aplicado el Algoritmo de Prim al grafo insertado y editado.
Poscondiciones	El sistema muestra el Árbol de Expansión Mínima.
Requisitos especiales	

Tabla 14. Descripción del caso de uso “Guardar solución”.

Nombre del caso de uso	Guardar solución
Actores	Usuario(inicia)
Resumen	Este caso de uso lo inicia el usuario cuando obtiene la solución y la guarda en una dirección específica.
Referencias	R9
Precondiciones	Debe haber construido el AEM.
Poscondiciones	Queda guardado el AEM en un fichero.
Requisitos especiales	Se debe validar que sean introducidos correctamente los datos.



Anexo 2 Descripción general de las clases

Tabla 15. Descripción de la clase “Draw Panel”

Nombre: DrawPanel	
Tipo de clase: Entidad	
Atributo	Tipo
puntoTmp	Point
vGraph	VisualGraph
nodos2Temp	ArrayList
Para cada responsabilidad:	
Nombre	Descripción
DrawPanel()	Construye un objeto de la clase.
Int findPosition(Component [], Component)	Devuelve la posición donde se encuentra el componente especificado en el arreglo de componente.
Void paintComponent(Graphics)	Método que se encarga de dibujar todos los componentes (nodos y arcos) en el panel de dibujo.
Void mouseMoved(MouseEvent)	Cambia el color del nodo donde se encuentra el mouse.

Tabla 16. Descripción de la clase “Label”

Nombre: Label	
Tipo de clase: Entidad	
Atributo	Tipo
Para cada responsabilidad:	
Nombre	Descripción
Label()	Construye un label con un texto por defecto.



Label(String)	Contruye un label con el texto especificado.
void mouseMoved(MouseEvent)	Cambia el color de línea del label.
void mouseDragged(MouseEvent)	Arrastra el label a la posición del mouse.
Void keyPressed(KeyEvent)	Verifica si se presiono la letra DEL y elimina el label.

Tabla 17. Descripción de la clase “MainWindow”

Nombre: MainWindow	
Tipo de clase: Entidad	
Atributo	Tipo
drawPanel1	DrawPanel
jButton1	JButton
jButton2	JButton
jFileChooser1	JFileChooser
jMenuItem1	JMenuItem
jPopupMenu1	JPopupMenu
Para cada responsabilidad:	
Nombre	Descripción
MainWindow()	Construye la ventana principal.
void jButton1ActionPerformed(ActionEvent)	Calcula el AEM y repinta el panel de dibujo.
void rigthClick(ActionEvent)	Crea un menú para la creación un nuevo nodo.
jMenuItem1ActionPerformed(ActionEvent)	Crea un nuevo nodo en el panel de dibujo.
ArrayList calcular()	Calcula el AEM y repinta el panel

Autor: Alexis Legrá Quintero



	de dibujo.
void jButton2ActionPerformed(ActionEvent)	Muestra una ventana para escoger donde se guardara la imagen con el AEM resultante.

Tabla 18. Descripción de la clase “VisualGraph”

Nombre: VisualGraph	
Tipo de clase: Entidad	
Atributo	Tipo
nodos	ArrayList
relaciones	ArrayList
Para cada responsabilidad:	
Nombre	Descripción
VisualGraph()	Construye un grafo.
boolean addNodo(String)	Annade un nodo al grafo.
void addRelacion(int,int,int)	Crea un arco entre dos nodos.
void addRelacion(int,int,int)	Crea un arco entre dos nodos con un color específico.
deleteRelation(int,int)	Elimina la relación entre dos nodos.
deleteRelationFromNode(int)	Elimina todas las relaciones de un nodo.
reorderNodesRelations(int)	Reordena las relaciones de un nodo.
constainsPath(int,int)	Verifica la existencia de la relación entre dos nodos.
resetRelationsColors()	Pone el color de todas relaciones del grafo en negro.

