UNIVERSIDAD DE HOLGUÍN "Oscar Lucero Moya"

FACULTAD DE INFORMÁTICA-MATEMÁTICA

Algoritmos Genéticos para la Generación de Bases de Conocimientos destinadas a la asistencia decisional en la Cadena de Suministros.

Trabajo para optar por el título de Ingeniero Informático

Autor: Reynier Reynerio Marrero Brito

Tutor: Ing. Irlán Grangel Gonzáles

Holguín

Junio de 2010

Declaración de Autoría

Declaro que soy el único autor de este Trabajo de Diploma, y autorizo a la Facultad de Informática Matemática de la Universidad de Holguín a que hagan uso del mismo como estimen pertinente.

Para que así conste firmo la presente a los 7 días del mes de junio de 2010.

---------- ----------

Firma del autor **Firma del Tutor**

Agradecimientos

Gracias a mi mamá, papá, Ronny y mi tío Manolo por haberme dado el apoyo que tanto necesité para poder culminar mis estudios.

Gracias a mi hermano Reymer, mi cuñada Yaneliys y a mi novia Yuneisy por los consejos y el ánimo que me dieron para poder terminar mi carrera.

Gracias a mis amigos de mi pueblo por haberme ayudado tanto, y siempre me han hecho sentir el deseo de ser alguien en la vida.

Gracias a mis amigos de GUILD Tarkan, Kreachers, Grendel, Dios, Wraxall, Brutalus, Cuza, Yandi, Esteban, Patica y aquellos que no estuvieron en la UHO pero si en la UCI, decirles que siempre se les recuerda por su apoyo y su confianza en mí.

Gracias a mi tutor Irlán Grangel por la oportunidad, su apoyo y la confianza que tuvo.

Dedicatoria

A mi familia.

Knowledge is Power (Sir Francis Bacon).

Resumen

Como resultado de la investigación conducida por Sáez Mosquera (2008), está disponible actualmente un framework para la asistencia decisional en procesos estratégicos de gestión logística, con enfoque de Cadenas de Suministro

El enfoque abordado en el framework propuesto plantea el uso del Modelo de Referencia de las Operaciones en la Cadena de Suministro (SCOR por sus siglas en inglés) creado en el año 1996 por parte del Supply Chain Council[1] y aporta una ontología[2] que hace uso del modelo mencionado para crear una base semántica común en los procesos de la Cadena de Suministros. Dicha base semántica es usada para la formulación de problemas decisionales y sus soluciones, a través de relaciones causa-efecto, formando di-grafos que son expresados y almacenados como documentos RDF (Resource Description FrameWork, por sus siglas en inglés).

En la investigación realizada por (Velázquez, 2009) se obtuvo un sistema informático capaz de gestionar visualmente problemas decisionales basados en la ontología SCOR y almacenarlos en una base de datos.

Debido a la necesidad de obtener una Base de Conocimientos que en un tiempo relativamente corto pueda representar la mayor cantidad de problemas decisionales se necesita una forma de generar, combinaciones posibles de problemas que, usando problemas ya diseñados por los decisores que están en uso de la herramienta, generar otros correctamente validados semánticamente y que permitan ampliar el conocimiento ya estructurado por la ontología. Por lo que se diseña un Algoritmo Genético que permite satisfacer la necesidad anterior descrita.

[1] Council, S.C. *Supply Chain Council*. 2008 [cited 2008 2008-02-05]; Available from: http://www.suply-chain.org.

[2] Gruber (1995): una ontología es una especificación de una conceptualización consensuada.

Summary

As a result of the investigation driven by Sáez Mosquera (2008), it is available at the moment a framework for the attendance decisional in strategic processes of logistical administration, with focus of Supply Chains.

The focus approached in the proposed framework outlines the use of the Model of Supply Chain Operations Reference (SCOR) created in the year 1996 on the part of the Supply Chain Council and it contributes an ontology that makes use of the aforementioned pattern to create a common semantic base in the processes of the Supply Chains. This base semantics is used for the formulation of problems decisionales and its solutions, through relationships cause-effect, forming di-graphs that are expressed and stored as RDF (Resource Description FrameWork, for its initials in English) documents.

In the investigation carried out for (Velázquez, 2009) was obtained a computer system able to visually administrate desitionals problems based on the ontology SCOR and to store them in a data base (DB).

Due to the necessity of obtaining a Base of Knowledge (BK) that at in a time relatively short it can represent the biggest quantity in problems desitionals a form it is needed of generating, combinations possible of problems that, already using problems designed by the users that are in use of the tool, to generate other correctly validated semantically and that they allow to already enlarge the knowledge structured by the ontology. That’s why was design a genetic algorithm that allows to satisfy the previous necessity.

Índice de Contenido

Índice de figuras

Índice de tablas

Introducción

Para muchos no es ajeno el hecho de que en las tareas de planificación de la producción, transporte, servicios y otras ramas económicas, aparecen una multitud de posibles decisiones a tomar por parte de las autoridades administrativas.

En el ámbito empresarial la toma de decisiones es, por naturaleza, compleja. En el escenario estratégico, la complejidad aumenta como consecuencia de la incertidumbre. Los encargados de la toma de decisiones en las organizaciones, enfrentan este problema desde muchas perspectivas y enfoques diferentes pero en la mayoría de los casos prácticos, utilizan complejos procesos de búsqueda e identificación de situaciones anteriores, tanto para comprender el problema que enfrentan como para la formulación de la solución.(Sáez Mosquera, 2008)

Como parte de la tesis doctoral de Sáez Mosquera [2008] se creó una ontología[3], con el objetivo de mejorar la gestión de la información dentro del contexto empresarial de la Cadena de Suministro (CS). Esta ontología fue creada basándose en el Modelo de Referencia de las Operaciones en la Cadena de Suministro (SCOR por sus siglas en inglés). En la investigación realizada por (Velázquez, 2009) se obtuvo una herramienta informática que gestiona visualmente las formulaciones de problemas decisionales y sus soluciones, permitiendo a los ejecutivos empresariales formular situaciones problémicas que requieren toma de decisiones en el contexto de la CS. Esta solución permite a los ejecutivos ir almacenando en la base de datos (BD) los problemas que se presentan y que requieren la toma de decisiones. Estos son formulados como un grafo dirigido (di-grafo), donde son relacionadas las causas, que explican, mediante relaciones causales, los síntomas manifestados en los problemas decisionales que se analizan, con los efectos en la situación que se está analizando. El sistema obtenido en la investigación ayuda a la visualización, formulación y almacenamiento en una BD de los problemas decisionales por parte de los decisores.

[3] No es más que un artefacto de ingeniería utilizado para describir ciertos dominios a través de conceptos y asunciones explicitas asociadas a dichos conceptos SÁEZ MOSQUERA, I. (2008) Procedimientos y arquitectura de apoyo para la asistencia decisional en procesos estratégicos de gestión logística. *Departamento de Ingeniería Industrial.* Santa Clara, Universidad Central "Marta Abreu" de Las Villas..

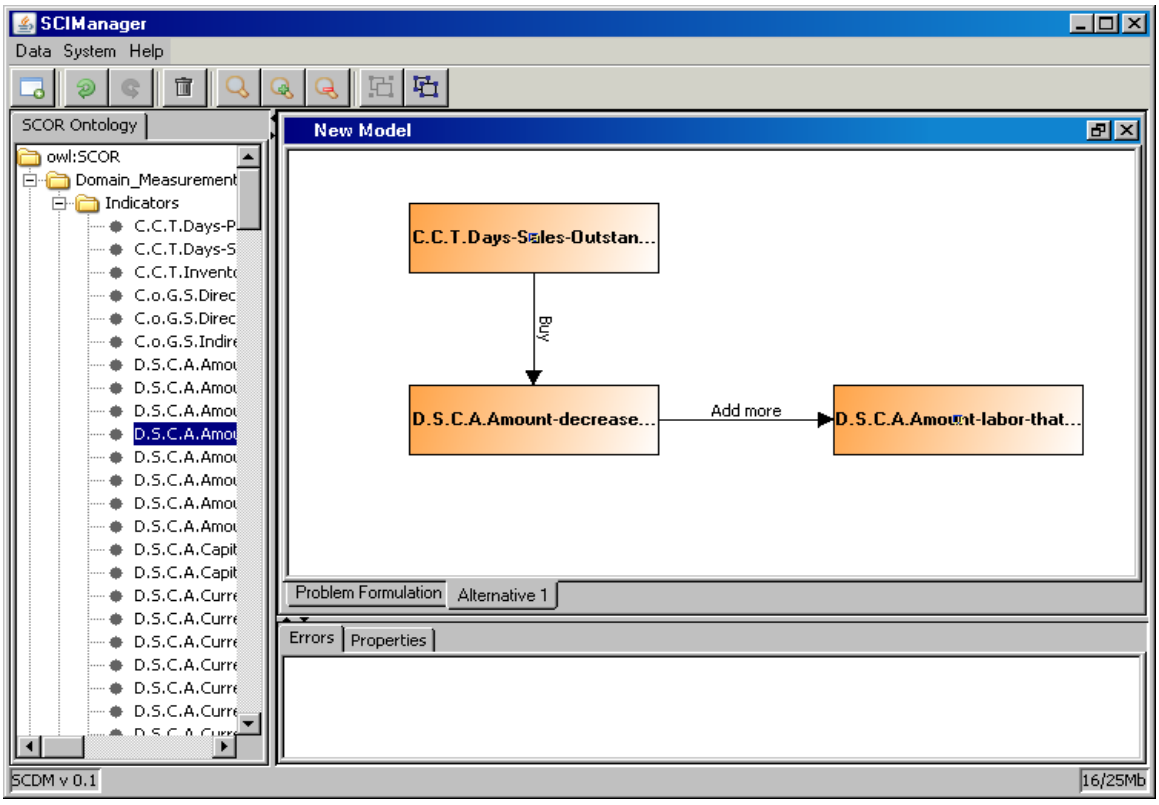


Fig. 1.1: Imagen tomada del Sistema informático obtenido. Fuente: (Velázquez, 2009).

Para la utilización del framework para la asistencia decisional se necesita contar con una Base de Conocimientos (BC) de formulaciones de problemas. El proceso de obtención de una BC es difícil debido a que la BC debe de ser poblada con formulaciones de problemas generadas por expertos en el dominio. Dada la complejidad y el tiempo requerido para lograr lo antes expuesto se necesita obtener una BC que en un tiempo relativamente corto pueda representar la mayor cantidad de problemas decisionales. Por lo que se necesita una forma de generar, la mayor cantidad de combinaciones posibles de problemas que, usando los indicadores y la semántica almacenada en la ontología y la propia semántica que aportan los problemas ya diseñados y almacenados por los decisores que están en uso de la herramienta, permita “corregir” los que no estén dentro de los parámetros ya definidos y generar otros que permitan ampliar el conocimiento implícito en las formulaciones de problemas decisionales y estructurado por la ontología.

De todo esto surge el **problema científico**: ¿Cómo generar conocimiento útil para la formación de una BC de problemas decisionales y sus soluciones a partir del conocimiento implícito en las formulaciones de los mismos?

La investigación se propone como **Objetivo General:** La implementación de Algoritmos Genéticos (AG) que en el contexto definido por la ontología SCOR permita generar y validar nuevos conocimientos a partir de las formulaciones de problemas decisionales y sus soluciones para la creación de una BC que permitirá la asistencia a la toma de decisiones con enfoque a la CS.

El **Objeto de estudio** de la investigación se definió como: La construcción de bases de conocimiento de problemas decisionales en el contexto de la toma de decisiones estratégico-logística con enfoque de CS.

Se define como **Campo de acción:** Utilización de los AG para la obtención de bases de conocimientos de problemas decisionales en el contexto de la toma de decisiones estratégico-logística con enfoque de CS.

Para guiar esta investigación fue necesario plantear las siguientes **preguntas científicas:**

- ✓ ¿Cuáles son los antecedentes y beneficios de la utilización de los AG en conjunto con las ontologías y la fundamentación teórica de estos?
- ✓ ¿Cómo utilizar la semántica almacenada en la ontología SCOR, en conjunto con la propia semántica de las formulaciones de los problemas decisionales y la potencia generativa de los AG?
- ✓ ¿Cómo generar nuevos conocimientos a partir del conocimiento ya almacenado en una BC?
- ✓ ¿Qué ventajas traería a la asistencia para la toma de decisiones la creación de una BC en un contexto específico (SCOR)?
- ✓ ¿Será sostenible el sistema obtenido?

Para lograr los objetivos trazados con esta investigación se llevaron a cabo las siguientes tareas:

1- Fundamentar el objeto de estudio de la investigación.

2- Estudiar teóricamente los AG.

3- Estudiar el uso de los AG en conjunto con el dominio definido por una ontología.

4- Estudiar el uso y creación de las BC.

5- Capturar los requerimientos del sistema informático a obtener.

6- Valorar la sostenibilidad del sistema informático.

7- Implementar el AG sobre el framework para la asistencia decisional en procesos estratégicos de gestión logística, con enfoque de Cadenas de Suministro

8- Probar el sistema.

Para realizar las tareas se emplearon los siguientes métodos de investigación:

Métodos Empíricos

- ✓ Estudio de documentos para la obtención de elementos que justifiquen el problema planteado y fundamenten su solución.

Métodos Teóricos

- ✓ Se utilizaron métodos teóricos como análisis y síntesis para el procesamiento de la información, lo que permitió la obtención de conocimiento y la simplificación de la información a procesar.

- ✓ El método Histórico Lógico permitió el estudio de la evolución del problema y la existencia de herramientas similares al que se pretende elaborar. De existir alguna, conocer los resultados alcanzados tras su aplicación.

- ✓ El enfoque sistémico permite analizar el problema y convertirlo en un sistema que responde a las necesidades planteadas.

- ✓ La modelación se utilizó en todas las etapas desarrollo del sistema.

Capitulo 1 Fundamentación teórica del tema.

1.1 Introducción

Los sistemas que gestionan información son generalmente artefactos de conocimiento que capturan y representan el conocimiento sobre ciertos dominios (Graciela Elisa Barchini, 2006). En este capítulo se abordará el fundamento de cómo generar nuevos conocimientos a través de un AG en el caso particular de la CS (Sáez Mosquera, 2008), de forma tal que se permita la creación de una BC correctamente validada en el dominio de la ontología SCOR.

1.2 Gestión del Conocimiento y Toma de Decisiones

La Gestión del Conocimiento (GC) se corresponde con actividades de creación, organización, clasificación y distribución del conocimiento (Guarino and Welty, 2000). La mayoría de estos lenguajes están definidos sobre una sintaxis basada en LISP.

Para los propósitos de la presente investigación y como resultado del análisis de las definición aportada por Mosquera, la GC se definió como: un proceso bien definido que involucra a toda la organización (y a su entorno), alcanzada a través del capital intelectual de la organización, que es mejorado a expensas de la creación de activos de conocimientos, producto del aumento de la disponibilidad y calidad de la información y la capacidad de la organización de convertir esta en nuevos cuerpos formales de conocimiento (Sáez Mosquera, 2008). De esta forma, el enunciado anterior resume la naturaleza y contenido de los procesos de gestión del conocimiento. Resulta importante destacar que esta definición establece un nexo entre información y contexto: la información sólo será relevante en un contexto particular de la organización, fuera del cual, perdería valor expresivo, reduciendo la efectividad del proceso de toma de decisiones si llegara a ser utilizada (Cespón Castro et al.). De esta forma, el par información-contexto en los procesos de toma de decisiones, conformará la noción de conocimiento, a partir del cual puede quedar fundamentada la relevancia de una información en alguna de las fases del proceso (Meagher and Wait, 2004).

Para muchos no es ajeno el hecho de que en las tareas de planificación de la producción, transporte, servicios y otras ramas económicas, aparecen una multitud de posibles decisiones a tomar por parte de las autoridades administrativas.

Debido a la gran interrelación que existe entre los diversos organismos y ministerio del ámbito económico, muchas de estas posibles decisiones pertenecen al contexto de la CS. En la actualidad existe un gran volumen de información vinculada con la CS que hace más compleja para los directivos la toma de decisiones(Sáez Mosquera, 2008). Para llegar a la toma de decisiones se necesita contar con un conocimiento representativo a lo largo del tiempo, que ilustre los problemas a los que se han tenido que enfrentar los administrativos y las soluciones que se han generado para los mismos.(Griffiths et al., 1999) Este conocimiento según (Sáez Mosquera, 2008) se representará usando formalismo de grafos. Estos grafos estarán conformados por indicadores definidos por elementos de la Ontología SCOR.

1.3 Bases de Conocimientos

Mientras cada día se avanza hacia un mundo cada vez más informatizado los productos y servicios de las organizaciones se han vuelto extremadamente más complejos usando componentes no materiales. El trabajo de las organizaciones se ha incrementado basado en el conocimiento(Martínez Delgado et al., 2003).

Las BC son repositorios centralizados de información (en una computadora o en una organización) contiene conceptos, datos, objetivos, requerimientos reglas y especificaciones. Una librería o una base de datos relacional acerca de un sujeto especifico son considerados ejemplos de BC las cuales son usadas para almacenar gran cantidad de información útil(D. Martinez, 2002). Una BC bien organizada puede disminuir drásticamente el tiempo de búsqueda de información por parte del software que se utilice como medio de consulta de la información por parte de los usuarios (John Mylopoulos, 1984).

Existen software que permiten la creación de BC son llamados software de administración de conocimientos (KMS por sus siglas en inglés). Las BC son un dinámico recurso que por sí solas tienen la capacidad de aprender de la información contenida como parte de la inteligencia artificial (IA) o sistemas expertos (Cater-steel and Al-Hakim, 2008), no son una colección de información estática. En el futuro se

espera que Internet se convierta en sí misma una gran BC a través de la Web Semántica.(Zilli et al., 2008)

Los Sistemas Basados en el Conocimiento (SBC) son un modelo computacional del más alto nivel que el paradigma de la programación convencional. El conocimiento que se almacena en la BC puede ser de diferente tipo: conocimiento simbólico sobre cómo resolver los problemas del dominio el cual se puede representar mediante diversas formas de representación del conocimiento (reglas de producción, frames, redes semánticas, etc.)(encyclopedia, 2005), probabilidades o frecuencias que modelan como se relacionan los valores de los diferentes rasgos que caracterizan el dominio, pesos de una red neuronal, casos o ejemplos de problemas del dominio, etc. Mientras más información o conocimiento se tenga sobre un dominio dado se tendrá más posibilidad de inferir con el conocimiento almacenado nuevos conocimientos (Wikipedia, 2009a). Las BC generalmente suelen centrase en un dominio dado del conocimiento para obtener la mayor cantidad de información de ese dominio de ahí sale la ventaja de usar Ontologías para generar BC utilizándolas para validar la información y permitir hacer inferencias sobre el conocimiento almacenado (Eschenbach and Gruninger, 2008).

1.4 Ontologías

Recientemente se ha desatado un creciente interés en las ontologías como artefactos para la representación del conocimiento y como componentes críticos en la gestión del conocimiento (Fensel, 2001), la Web Semántica, el comercio electrónico entre otros campos. Sin embargo, existen otros conceptos como especificaciones de lo que existe, o de lo que se puede decir sobre el mundo, que se han utilizado desde la filosofía aristotélica. Según el concepto en Filosofía una ontología es una teoría sobre la naturaleza de la existencia, de qué tipo de cosas existen. Parte de la metafísica que trata del ser en general y de sus propiedades trascendentales(Abad, 2003). Así una ontología como disciplina estudia dichas teorías. Se confunde a menudo con la epistemología, que trata del conocimiento y el conocer (Firestone, 1998).

En el siglo XVII el término ontología se usa como sinónimo de metafísica, concretamente como la rama de la metafísica que trata con la naturaleza del ser (Andrea Bellandi, 2007). El término ontología es adoptado por la inteligencia artificial a

finales de la década de los 80s para compartir y reutilizar conocimiento, mientras que en la segunda mitad de los 90s se incorpora a la ingeniería web para la inclusión de descripciones semánticas explícitas de recursos (contenidos y servicios)(Eschenbach and Gruninger, 2008). En ambas disciplinas una ontología se materializa en un documento o un fichero que define formalmente las relaciones entre términos(Martin Hepp, 2008). En el campo de la inteligencia artificial lo que existe es aquello que se puede representar (Norvig, 1995). Cuando el conocimiento de un dominio se representa mediante un formalismo declarativo, el conjunto de objetos que pueden representarse se denomina universo del discurso. Este conjunto de objetos y las relaciones existentes entre ellos se reflejan en un vocabulario de representación con el que un programa basado en conocimiento representa el conocimiento (Chris Welty, 2006).

Por tanto, en el contexto de la inteligencia artificial se puede describir la ontología de un programa mediante la definición de un conjunto de términos de representación (Ceccaroni, 2001). Así, en una ontología, las definiciones asocian los nombres de las entidades en el universo de discurso (clases, relaciones, funciones u otros objetos) con texto legible desde un punto de vista humano, describiendo lo que significan los nombres, y axiomas formales que restringen la interpretación y el uso adecuado de estos términos permitiendo hacer inferencias sobre el conocimiento representado con la ontología. Las ontologías para representar conocimiento precisan los siguientes componentes:

• **Conceptos:** son las ideas básicas que se intentan formalizar. Los conceptos pueden ser clases de objetos, métodos, planes, estrategias, procesos de razonamiento, etc.

• **Relaciones:** representan la interacción y enlace entre los conceptos, formando la taxonomía del dominio. Las relaciones básicas son: subclase-de, parte-de, conectada-a.

• **Funciones:** son un tipo concreto de relación donde se identifica un elemento mediante el cálculo de una función que considera varios elementos de la ontología.

• **Instancias:** se utilizan para representar objetos determinados de un concepto.

• **Axiomas:** son teoremas que se declaran sobre relaciones que deben cumplir los elementos de la ontología. Especifican las definiciones de los términos en la ontología y las restricciones de sus interpretaciones. Los axiomas deben proveerse para definir la semántica o el significado de los términos (Zilli et al., 2008).

1.5 Algoritmos Genéticos

La naturaleza tiene una forma increíble de crear individuos a partir de individuos ya existentes. Cada nueva generación de organismos presenta características similares a la de los organismos de las generaciones anteriores. Si el medio ambiente va cambiando lentamente las especies pueden ir evolucionando junto con él, pero un cambio brusco en el medio ambiente puede acabar ocasionando que alguna especie desaparezca. A veces los individuos presentan mutaciones aleatorias y aunque estas pueden ser causa de una muerte rápida para el individuo algunas contribuyen a la creación de nuevas especies. La publicación de Darwin "*The Origin of Species on the Basis of Natural Selection"* fue el mejor punto de partida para la historia de la ciencia. Esta demostró que si se estudia la madre naturaleza se puede utilizar sus enseñanzas para la creación de sistemas que utilicen inteligencia artificia(D. Martinez, 2002).

El término AG se usa por el hecho de que estos simulan los procesos de la evolución darwiniana a través del uso de operadores genéticos que operan sobre una población de individuos que "evoluciona" de una generación a otra (Mitchell, 1999). Desde finales de los años 50 y principios de los 60 se desarrollaron trabajos por algunos biólogos para simular sistemas genéticos en una computadora, dentro de estos trabajos es destacable el de Fraser, entre los años 60 y 62, donde se llegó a algo bastante parecido a los AG, trabajando con cadenas y fenotipos. A pesar de estos trabajos, se reconoce al profesor John Holland de la Universidad de Michigan como el creador de los AG con su trabajo sobre teoría de los sistemas adaptativos, en el año 62. Es Holland el primero que adaptó la idea de la genética a sistemas artificiales. El nombre de AG lo uso por primera vez Bagley en el año 67, y el mismo es el autor de un trabajo importante sobre la utilización de AG en juegos. Holland siguió investigando en este campo y en el año 75 introdujo mejoras importantes en los AG como es el escalado y publicó un libro sobre el tema que es material de referencia clásico. En la actualidad, existen una gran cantidad de problemas en los que se usan los AG y han llegado a consolidarse como métodos de búsqueda y de optimización (Wikipedia, 2009b).

Los AG son modelos abstractos de la genética natural y los procesos evolutivos. Los mismos incluyen conceptos como cromosomas, genes, cruce, mutación, etc. (Mitchell, 1999).

El proceso principal de un AG se describe de la forma siguiente:

Al inicio se generan aleatoriamente soluciones o cromosomas para el problema que se resuelve. Luego se realizan iteraciones. Cada iteración consiste de varios pasos, en ellos se seleccionan buenas soluciones y se realizan cruces, ocasionalmente también mutaciones. Mediante la selección de buenas soluciones se obtienen a partir de ellas mejores soluciones, como ocurre en la evolución natural. Un AG puede ser visto como una estructura de control que organiza o dirige un conjunto de transformaciones y operaciones diseñadas para simular estos procesos de evolución (Whitley, 1993).

Los AG se pueden considerar métodos de búsqueda aleatoriamente guiados. En alguna medida se realiza una búsqueda a ciegas pues el proceso no tiene ninguna información sobre el problema que se está tratando de resolver, exceptuando un valor de aptitud de los individuos. También son presentados como un modelo de aprendizaje automatizado (machine learning) basado en la evolución natural; viendo el aprendizaje como una competencia entre una población de candidatos a ser solución de un problema (Norvig, 1995). Otros lo incluyen entre las componentes de la llamada Computación blanda (SC por sus siglas en inglés).

Los AG son aplicables en la solución de diferentes tipos de problemas. Entre ellos están los de control, de clasificación, biología molecular (Norvig, 1995).

También han dado lugar a la llamada Programación genética, la cual es un sub campo de los AG donde las soluciones son programas de computadoras. Otro término surgido a partir de los AG es el de Algoritmos Meméticos; estos son algoritmos de búsqueda en los cuales se combina la idea de los AG con técnicas de búsqueda local (Wikipedia, 2009b).

Antes de de aplicar un AG a un problema hay que tener bien claro lo siguiente:

1- ¿Qué es la función de calidad?

2- ¿Cómo se representa un individuo?

3- ¿Cómo son escogidos los individuos?

4- ¿Cómo se reproducen los individuos?

5- ¿Cómo mutan los individuos?

La función de calidad depende del problema pero en cualquier caso es una función que recibe como entrada un individuo y devuelve un número como salida que significa el grado de perfección del individuo, básicamente esta función mide la cantidad de defectos de los individuos y sirve de medidor para escoger los mejores individuos para la reproducción. En los AG clásicos los individuos son representados como una cadena de caracteres sobre un alfabeto finito donde cada carácter representa un gen del individuo. Normalmente los individuos se representan sobre el alfabeto (0,1) en este caso el individuo sería representado como una cadena de bits (Whitley, 1993). Los individuos son escogidos con el objetivo de ir mejorando la especie para ello se escogen los mejores individuos y se cruzan dada una probabilidad de cruzamiento esta probabilidad es dada por la función de calidad. Los individuos escogidos para ser reproductores se cruzan entre sí para generar nuevos individuos que heredaran genes de ambos progenitores. Al mismo tiempo los individuos resultantes tienen una probabilidad de mutar que no es más que alguno de sus genes cambian aleatoriamente (O'Reilly, 2002).

1.5.1 Componentes de los AG.

a) Cromosomas

Los cromosomas representan soluciones potenciales al problema (Mitchell, 1999). O sea, es un valor o conjunto de valores que puede son candidatos a ser la solución correcta de un problema. Los mismos se representan usualmente como cadenas de caracteres sobre un alfabeto finito (0.1...,9) (Ferrer, 2007 z), etc. Cada componente de esa cadena es equivalente a un gen, y estos representan parámetros o variables del problema. La selección de una representación está fuertemente asociada con el diseño de los operadores genéticos.

b) Población.

Una población es un conjunto de cromosomas, cada de ellos se denomina individuo o elemento de la población. La misma representa un conjunto de soluciones candidatas del problema (Whitley, 1993). La población inicial puede ser obtenida por diferentes vías, entre ellas está generarlas aleatoriamente o formarlas a partir del criterio experto. Una vía alternativa es usar un enfoque basado en casos.

c) Calidad de la solución.

La calidad o grado de aptitud de la solución (Fitness Function) es una medida que permite comparar las soluciones para determinar cuál es mejor. Usualmente se expresa como una función que tiene como argumento el cromosoma y le asigna a este un valor de aptitud (O'Reilly, 2002), por ejemplo, en el problema del viajero vendedor el costo (quizás expresado en distancia) del camino indicado por ese cromosoma. Este grado de aptitud puede ser determinado usando fórmulas complejas, modelos de simulación, a partir de experimentos, etc. Usualmente este valor se normaliza, es decir, se calcula la suma de los valores de calidad calculados a todos los individuos y luego se divide la calidad de cada cromosoma por este valor.

d) Operadores genéticos.

Son los que permiten crear nuevas poblaciones a partir de una existente. Los operadores más conocidos son los de selección o reproducción, cruce y mutación. También hay otros menos utilizados como el de inversión (Norvig, 1995).

El operador de selección o reproducción permite crear un conjunto intermedio de individuos a partir de la población actual. El conjunto se considera intermedio pues son individuos de una generación o población con los cuales se creará la siguiente usando los otros operadores genéticos. El operador de reproducción realiza esta selección basado en la calidad de cada individuo de la población actual. Las soluciones o individuos de mayor calidad tienen más posibilidades de ser seleccionados para formar el conjunto intermedio (tal y como sucede con el principio de la selección natural). Un método para implementar la reproducción es el llamado de la ruleta, en el cual a cada individuo se le asigna una sección del perímetro de la ruleta proporcionalmente a su grado de aptitud. Un paso de selección es análogo a un giro

de la ruleta, seleccionando el individuo correspondiente según el arco del círculo donde se detiene (Mitchell, 1999).

El operador de cruce permite combinar pares de individuos para formar soluciones nuevas intercambiando partes de un par de cromosomas. El cruce se puede realizar usando un punto o dos puntos de cruce estos puntos de cruce permiten al algoritmo saber que cromosomas de los individuos cruzar y así obtener una nueva combinación de cromosomas que formarán al nuevo individuo o solución (Whitley, 1993). El operador de mutación cambia el valor de un grupo de los genes del cromosoma por otros valores del dominio del gen. Este operador permite crear nuevo cromosoma que quizás sea inalcanzable por cruce, preservando la diversidad.

e) Parámetros del algoritmo genético.

Entre los parámetros que caracterizan un algoritmo genético están:

1- Tamaño de la población: indica la cantidad de soluciones que forman la población.

2- Número de generaciones: cantidad de iteraciones que realizara el algoritmo hasta alcanzar una solución satisfactoria.

3- Probabilidad de cruce: Valor entre 0 y 1 que permite determinar si procede o no el cruce. Como la frecuencia de cruce es alta este debe ser un valor alto (pc = 0.7, pc = 0.91, $etc.$).

4-Probabilidad de mutación: Valor entre 0 y 1 que permite determinar si procede o no la mutación. Como la frecuencia de mutación es pequeña este debe ser un valor pequeño (pc = 0.007, pc = 0.0091, $etc.$) (Mitchell, 1999).

1.5.2 Pasos de un algoritmo genético.

Existen diversas formulaciones para un algoritmo genético. Una relativamente simple es la siguiente:

P1- Inicializar la población inicial. Generar un conjunto de soluciones aleatoriamente (o usando otra vía).

P2: Reproducción.

a. *Calcular la calidad de todos los individuos de la población actual.*

b. *Crear una población intermedia (conjunto intermedio de soluciones).*

P3: Cruce.

Realizar los pasos siguientes hasta que la nueva población alcance el tamaño establecido:

a. *Tomar dos individuos del conjunto intermedio. Generar un número entre 0 y 1. Si este valor es menor que la probabilidad de cruce entonces ir P3b; sino ir a P3c.*

b. *Seleccionar los puntos de cruce. Realizar el intercambio de genes de los dos individuos seleccionados en P3a entre los puntos de cruce. Colocar los nuevos cromosomas en la nueva población.*

c. *Colocar copias de estos cromosomas en la nueva población*.(Mitchell, 1999)

P4: Para los individuos de la nueva población aplicar el operador de la mutación siempre que el número entre 0 y 1 generado sea menor que la probabilidad de mutación.

Repetir los pasos P2, P3 y P4 hasta alcanzar la condición de terminación:

(La condición de terminación puede ser un número de generaciones determinado, un criterio basado en la calidad de las soluciones, o un criterio sobre la razón de cambio de la calidad de las soluciones).

Este esquema puede ser mejorado manteniendo en memoria el mejor individuo encontrado hasta ese momento, de modo que al terminar se pueda seleccionar ese como solución del problema. Nótese que es necesario mantenerlo en memoria pues el funcionamiento del AG no garantiza que este se mantenga a través de todas las generaciones hasta el final (O'Reilly, 2002).

1.6 Herramientas utilizadas para desarrollar la aplicación

Para llevar a cabo el desarrollo de la aplicación informática se pretende utilizar múltiples herramientas que proveerán rapidez y eficiencia al desarrollo. A continuación

se explicarán las principales características y ventajas que se tuvieron en cuenta para su uso.

1.6.1 Proceso Unificado de Desarrollo de Software y el Lenguaje Unificado de Modelación

En la actualidad ha tomado gran auge en el desarrollo de software la aplicación de metodologías con Lenguaje Unificado de Modelación (UML5 por sus siglas en inglés). Su utilización ha introducido eficiencia y productividad en el desarrollo de software. Una de estas metodologías es el Proceso Unificado de Desarrollo de Software (RUP por sus siglas en inglés). RUP como se dijo anteriormente utiliza UML permitiendo la modelación visual en los artefactos que define, lo que permite además incorporar al proceso de desarrollo de software un mejor control de los requerimientos y cambios (Cater-steel and Al-Hakim, 2008). Sus características fundamentales son:

• Dirigidos por casos de uso.

• Centrado en la arquitectura.

• Iterativo e incremental.

Estas características hacen único al Proceso Unificado, facilitan la obtención de un sistema en su completo desenvolvimiento en correspondencia con las necesidades de los usuarios, brindan amplias posibilidades para el manejo eficiente del tiempo de diseño e implementación y minimizan el riesgo de obtención de un mal producto (o un producto no deseado) porque el sistema puede validarse con el cliente en cada iteración. Hoy en día, UML está consolidado como el lenguaje estándar en el análisis y diseño de sistemas de cómputo. Mediante UML es posible establecer la serie de requerimientos y estructuras necesarias para plasmar un sistema de software previo al proceso intensivo de escribir código.

Algunas de las ventajas en el uso de UML son las siguientes:
• Facilita la asimilación y entendimiento por parte del equipo de desarrollo.
• Minimiza el tiempo invertido en el desarrollo de la arquitectura.
• La trazabilidad y documentación del proyecto se realiza de una forma ordenada.
• Efectividad y productividad en labores de diseño arquitectónico y mantenimiento.

1.6.2 Visual Paradigm for UML 2.3

Para la modelación en todas las etapas de desarrollo será utilizada la herramienta Visual Paradigm for UML 2.3. Esta es una herramienta CASE para el desarrollo con UML. La misma posee una versión (Community Edition) en el mercado para su uso libre. Posee además funcionalidades que lo hacen atractivo para los desarrolladores como:

• Generación de código y la base de datos a partir de los diagramas

UML realizados.

• La realización de Ingeniería Inversa.

• Generación automática de informes en formato PDF, Word o HTML.

1.6.3 Lenguajes de programación

El avance de las nuevas tecnologías de la información y las comunicaciones ha permitido el surgimiento de una variedad de herramientas que facilitan el desarrollo de software para los más variados usos. Los lenguajes de programación han permitido la evolución de estas herramientas. CASE: Computer Aided Software Engineering

Entre los lenguajes que figuran en la lista de los más apropiados para desarrollar cualquier tipo aplicación se encuentran C++, Java y C#. Todos ellos gozan de una excelente reputación en el ámbito académico y tienen una gran potencia en lo que a efectos de programación orientada a objetos y arquitectura distribuida se refiere.

Java

De los lenguajes antes mencionados se seleccionó Java. Las características fundamentales que conllevaron a seleccionar como lenguaje de programación a Java se enumeran a continuación:

Experiencia de programación: El hincapié que se ha hecho sobre el lenguaje de programación Java durante los últimos años en la facultad de Informática y Matemática perteneciente a la Universidad de Holguín “Oscar Lucero Moya” ha sido una pieza clave para la formación de todos los estudiantes y por ende este factor es

muy significativo para la selección de dicho lenguaje. Entre algunas de las ventajas de este lenguaje se encuentra el buen modelo de programación orientada a objetos, su continua renovación y mejora de sus compiladores e intérpretes.

Amplio respaldo de la comunidad Open Source: A diferencia de otros lenguajes de programación puramente comerciales, el código fuente de todas las APIs de java está a la disposición de los programadores. Esta característica, potenciada por la política de abundante y buena documentación llevada a cabo por Sun Microsystems, hace que cualquier duda o proyecto de envergadura que se quiera afrontar en Java cuente con mayores facilidades.

Facilidad de obtención de IDEs: No sólo el código fuente de Java es de libre distribución, sino que existen numerosos entornos de programación para él totalmente gratuitos. Todos ofrecen las garantías y características más que suficientes para desarrollar proyectos informáticos de elevada complejidad.

Entre las alternativas existentes se destacan las siguientes:

• JBuilder Fundation: es un IDE gratuito desarrollado por Borland.

• NetBeans: software libre desarrollado por Sun Microsystems sin objetivos comerciales.

• Eclipse: software libre incluido por IBM como parte de su solución

• WebShepere.

Para desarrollar esta investigación se utilizó Eclipse debido a la experiencia del programador con el mismo.

Continuidad del trabajo

Como ya se ha venido diciendo existe ya un Sistema Informático (SI) obtenido en la investigación de Armando Carracedo Velázquez, dicho SI está hecho con el lenguaje de programación Java por lo que permitirá a esta Investigación dar continuidad al trabajo que se viene realizando con el objetivo de la asistencia decisional con enfoque a la cadena de suministro.

1.6.4 Gestor de Base de Datos PostgreSQL.

Uno de los SGBD más utilizados hoy en día es PostgreSQL. Surgió del paquete de Postgres desarrollado en la Universidad de Berkeley en California. Con más de una década de desarrollo como respaldo, PostgreSQL es uno de los sistemas de BD libre más avanzados que está disponible en el mundo hoy. PostgreSQL ha sido utilizado para implementar muchas aplicaciones de producción e investigación donde se incluye un sistema de análisis, un paquete de control del rendimiento de un motor de avión, una BD para el seguimiento de asteroides, una BD de información médica y varios sistemas de información geográfica. PostgreSQL ha sido utilizado también como una herramienta educacional en varias universidades. El modelo relacional sustituyó modelos previos, en parte por su simplicidad. Sin embargo, esta simplicidad hace muy difícil la implementación de ciertas aplicaciones. PostgreSQL ofrece una potencia adicional sustancial al incorporar los siguientes cuatro conceptos adicionales básicos en una vía en la que los usuarios pueden extender fácilmente el sistema:

- ✓ Clases: Esta es una de las características fundamentales de PostgreSQL y es la colección de instancias de un objeto. Cada instancia tiene la misma colección de atributos y cada atributo es de un tipo específico, además cada instancia tiene un identificador de objeto (OID).
- ✓ Herencia: Una clase puede heredar de ninguna o varias otras clases y una consulta puede hacer referencia a todas las instancias de una clase.
- ✓ Tipos: Pueden ser definidos nuevos tipos de datos además de los convencionales.
- ✓ Funciones: Las funciones de conjunto calculan un único resultado a partir de múltiples filas de entrada.

Otras características que aportan potencia y flexibilidad adicional son:

- ✓ Restricciones.
- ✓ Disparadores.
- ✓ Reglas.
- ✓ Integridad transaccional.

Estas características colocan a PostgreSQL en la categoría de las BD identificadas como objeto-relacionales.

En PostgreSQL se utilizan las vistas como tablas virtuales, como una tabla que no existe físicamente en la BD, pero aparece al usuario como si existiera.

Cuando se habla de una tabla base, se refiere a que realmente hay un equivalente almacenado para cada fila en la tabla en algún sitio del almacenamiento físico.

Uno de los principios del modelo relacional es que los atributos de una relación son atómicos. PostgreSQL no contiene esta restricción; los atributos pueden contener subvalores a los que puede accederse desde el lenguaje de consulta. Por ejemplo, se pueden crear atributos que sean vectores de longitud fija o variable de algunos de los tipos base.

PostgreSQL constituye una herramienta importante, útil y mundialmente muy utilizada en gran variedad de sistemas informáticos con disímiles características.

1.7 Conclusiones parciales

- Como parte del estudio realizado de cómo generar nuevos conocimientos a través de técnicas de IA utilizando una ontología se optó por utilizar e implementar un AG aprovechando la potencia generativa de estos

- El uso de ontologías y BC en los SI es una práctica que está en gran auge en las ciencias informáticas, todo ello se debe a las siguientes posiciones teóricas:

1. Proveen un contexto común entre los desarrolladores de sistemas y elimina diferencias entre los puntos de vista de los mismos.

2. Resuelve el problema de interoperabilidad semántica haciendo un buen uso de la información.

3. Establece correspondencia y relación entre los diferentes dominio de entidades información.

4. Almacenan grandes volúmenes de información de un dominio dado.

5. Permiten validar la información para que esta tenga sentido y sea reutilizable en cualquier momento.

6. Los SI hacen uso extensivo de ellas por su potencialidad de inferir nuevos conocimientos.

- Dado que Armando Carracedo obtuvo una herramienta para gestionar los problemas decisionales y almacenarlos como di-grafos se utilizara esta herramienta para implementar sobre ella un modulo de IA. Donde se implemente el AG que se propone para generar nuevos problemas decisionales utilizando la semántica de los problemas ya almacenados permita construir una BC que esté poblada con la mayor cantidad de problemas decisionales en un tiempo relativamente corto.

- A través del estudio realizado se concluyó que el lenguaje Java se ajusta para lograr los requisitos que debe poseer el algoritmo que se pretende implementar.

- Se decide utilizar como SGBD el PostgreSQL por las ventajas que ofrece, además de ser software libre.

Capitulo 2 Descripción de la solución propuesta.

2.1 Introducción

El objetivo de este capítulo es mostrar la solución que se propone para resolver el problema científico que se encontró. En él se explica detalladamente la solución encontrada a través de los diferentes diagramas UML así como la justificación de la solución escogida entre las soluciones candidatas para ser implementada en el framework de asistencia decisional.

2.2 Propuestas de AG

Para determinar el AG final que se implementó sobre el framework de asistencia decisional inicialmente fue necesario implementar dos AG llamados AG1 y AG2 con el fin de optar por el candidato que se acercara lo más posible a la solución ideal. Siempre teniendo en cuenta un grupo de indicadores o requisitos que los AG candidatos debían cumplir, y que gracias a ellos fue posible medir a los candidatos para optar por el más indicado a ser parte de la solución.

2.2.1 Características comunes entre AG1 y AG2.

Entre los algoritmos candidatos presentados existen características semejantes que están presentes en cada uno de ellos y se presentan a continuación.

Proceso de selección

Este proceso cada uno de los AG candidatos escoge de una lista de problemas decisionales cargados desde una BD, según un criterio de semejanza los problemas que pasarán a al cruzamiento, estos se agrupan en dúos formando parejas[4]. Este proceso arroja como resultado una lista de parejas de problemas seleccionados para cruzarse entre ellos.

[4] Dúo de problemas decisionales de los cuales se obtendrán problemas hijos.

Proceso de cruzamiento

Este proceso ocurre en cada pareja con una probabilidad de cruzamiento de 1. Para que todas las parejas generen hijos y así incrementar la cantidad de problemas generados por el AG. Es aquí donde se obtienen una lista de problemas hijos[5] generados por el cruzamiento de las parejas. Cada pareja va generando hijos mientras se cumpla una condición de parada o función de calidad, o sea que la cantidad de hijos que son generados por una pareja va a estar regida por la función de calidad.

Función de calidad

Esta función es utilizada por todos los AG candidatos y tiene como objetivo ir midiendo la calidad de los hijos obtenidos para así saber cuando una pareja está generando hijos con mejor o peor calidad. Esta funciona de la siguiente manera:
Básicamente esta función realiza una comparación de la calidad del último hijo obtenido con la calidad del anterior y determina si la calidad del último hijo o alternativa es mayor que la calidad del anterior hijo Para ello al menos tiene que haber dos (2) hijos para que la función comience a evaluar los hijos obtenidos. La calidad va a estar dada por el número de limitaciones que tengan los individuos evaluados estas limitaciones son las relaciones $A -> B$ que el individuo tenga con Frecuencia Total[6], (FT) = 0.0, esta sería la frecuencia de aparición de esa relación en la BC completa y en el caso de que sea 0.0 indica que esa relación no está contenida en la BC y entonces es muy probable que sea una relación no válida. El número de limitaciones de una alternativa o hijo va a ser el número de relaciones que posea con $FT = 0.0$

El número de limitaciones tiende a crecer como el cáncer. La frase es más que una metáfora. El crecimiento del cáncer, en los estadios primarios, va de (10^9) células a (10^{12}). El cambio en la potencia es 3, o lo que es casi lo mismo decir que el crecimiento de las células es tres veces mayor. Siendo así, entonces es sensato suponer que si el número de limitaciones de las alternativas o hijos generados excede esta proporción, entonces el crecimiento de las limitaciones sería exponencial y el algoritmo debe parar de generar propuestas.

[5] Son los problemas decisionales o alternativas obtenidas por los diferentes procesos de los AG.
[6] La frecuencia Total es la frecuencia de aparición de una relación A->B en toda la BC.

Las técnicas para determinar si una serie de números ajusta a una distribución exponencial requieren de una muestra grande para tener significación estadística la conclusión sobre la distribución de la que provienen. Esto hace lidiar con el mismo problema, Se necesitaría generar muchas soluciones para tener una serie lo suficientemente larga para confirmar su comportamiento estadístico. Los logaritmos ayudan en este momento.

Si $a^C = b$, entonces el $\log_a b = c$. Esta es la regla general de los logaritmos. Como la antes mencionada es una función exponencial de la forma 10^n la base que se toma es Euler. Entonces, se deduce que:

Siendo b_i el número de limitaciones de una alternativa *i*, se tiene:

$e^c = b$ entonces, el $Lnb = \mu X$

μX en una distribución exponencial debe mantenerse constante, para que se mantenga el carácter monótono creciente de esta distribución. Siendo así, entonces:

$Lnb1 = Lnb2$ Para dos alternativas o hijos consecutivos.

Entonces, se debe esperar que:

$Lnb1/Lnb2 = c$, donde c tiene un comportamiento monótono creciente, pero con una cota superior en 1 y b1, b2 son el número de limitaciones para el penúltimo y último hijo generados respectivamente.

Si se asume que 1 es el 100%, es decir, cuando el cociente llega a ser 1 el algoritmo ha alcanzado un nivel crítico porque el número de limitaciones de una alternativa a la otra se mantiene constante (se encontraría en la asíntota de la exponencial). Si se permite que el algoritmo llegue a este punto (algo imposible por la propia naturaleza de una asíntota) entonces el costo computacional y el número de individuos generados es muy grande. Entonces es de suponer que cuando la función de calidad encuentre que se cumpla $Lnb1/Lnb2 = c$, siendo $c > 1$ devuelva verdadero en otro caso devuelve falso. Pero existe la posibilidad de que el número de limitaciones de las dos últimas alternativas sean 0, o sea que no presentan ninguna relación con $FT = 0.0$ entonces al evaluar esto quedaría $Ln\,0/Ln\,0 = c$ y la expresión queda indefinida por lo que se hace una salvedad en caso de que ocurra esta situación. Cuando la cantidad de

limitaciones de las dos alternativas a comparar es 0 entonces se compara la mayor FT de ambas alternativas, verificando que la FT de la última alternativa sea siempre mayor que la alternativa anterior esto quiere decir que se están generando alternativas con mayor frecuencia por lo tanto con mejor validez.

Proceso de mutación

Este proceso ocurre después del proceso de cruzamiento y necesita una lista de problemas hijos obtenidos en el cruzamiento y de acuerdo a una probabilidad de ocurrencia de la mutación de 0.25 se seleccionan los hijos que pasan a ser mutados aleatoriamente. Esto consiste en cambiar indicadores que forman al problema hijo por indicadores de la ontología. Este proceso escoge la cantidad y los indicadores a mutar por los hijos seleccionados de forma aleatoria. También los indicadores de la ontología se escogen aleatoriamente para remplazar estos indicadores seleccionados.

Hijos obtenidos

Los hijos obtenidos por los algoritmos es la lista de hijos (válidos y no válidos) obtenidos en el proceso cruzamiento unido a los hijos (válidos y no válidos) obtenidos en el proceso de mutación que pasan por un proceso de validación para eliminar a las alternativas no válidas. Los hijos obtenidos en el proceso de mutación no sustituyen a los hijos originales sino que pasan también a formar parte de los hijos obtenidos porque si se sustituyen se estaría eliminando la posibilidad de un hijo válido. O sea que los hijos obtenidos después de la ocurrencia de todos los procesos de los algoritmos y que conforman los problemas a presentar al usuario van a ser los obtenidos en el proceso de cruzamiento que sean válidos unidos a los hijos válidos obtenidos por el proceso de mutación.

Proceso de validación

Este proceso es el mismo para los algoritmos y determina cuando un hijo es válido o no. Para ello se tienen en cuenta los siguientes criterios:

1- El hijo en cuestión no puede tener ciclos en el grafo de relaciones que lo representa.

2- El hijo no puede tener en su representación o más de un grafo que lo represente o sea todas las relaciones tienen que estar contenidas en un mismo grafo.

2.2.2 Algoritmo Genético AG1

Como el objetivo principal del AG1 en esta situación problémica es obtener la mayor cantidad de hijos válidos posibles. El AG1 intenta maximizar el número de parejas formadas para el cruzamiento, así al tener mayor cantidad de combinaciones de parejas posibles sin importar las características de los individuos que la forman, respecto al análisis estructural de la formulación. Al existir un mayor número de parejas para reproducirse se tiene una mayor posibilidad de obtener un mayor número hijos. Esto no quiere decir que todos sean válidos pero maximizando la cantidad de estos aumentan las posibilidades. Entre las características peculiares de este algoritmo se encuentran:

1- Se escogen todas las combinaciones posibles de parejas de individuos para la reproducción.

2- En el proceso de cruzamiento, las causas raíces de los padres se cruzan también o sea las causas raíces de los hijos es una combinación de indicadores pertenecientes a las causas raíces de los padres.

3- En el proceso de mutación los hijos a mutar también mutan los indicadores que pertenecen a las causas raíces.

Este algoritmo se ejecuta obedeciendo los pasos lógicos que a continuación se exponen:

1- Se cargan las formulaciones de problemas decisionales existentes (almacenados en la BD).

2- Se seleccionan las parejas que pasarán a cruzarse para obtener nuevos hijos a través de un método selección. Este método realiza la selección de manera que forma parejas con todas las posibles combinaciones de individuos de la población, lo único que debe cumplir este método es no formar parejas ya existentes y no formar parejas de un individuo con sí mismo obteniéndose $n \times (n-1)/2$ parejas siendo n el número

total de individuos de la población. Con este método se obtiene una lista de parejas y se pasa al cruzamiento de las mismas.

3- Con la lista de parejas a cruzar se ejecuta el proceso de cruzamiento con una probabilidad de cruce de 1 de cada pareja. Este es un proceso que ocurre en las parejas y tiene como objetivo obtener hijos por cada pareja mientras que la cantidad de hijos ya obtenidos sea menor que 2 ó la función de calidad sea verdadera. Cuando estas dos condiciones no se cumplen el método termina y devuelve los hijos generados hasta el momento. Los nuevos individuos hijos se obtienen heredando aleatoriamente indicadores de cualquiera de los dos padres siempre que estos sean del mismo nivel[7] del indicador hijo que se está formando, o sea que para formar un nuevo indicador del individuo hijo este indicador va a ser igual a cualquier indicador de los padres siempre que sean del mismo nivel. Este procedimiento se realiza para obtener los indicadores de cada nivel del individuo hijo incluyendo las raíces o el nivel 0.

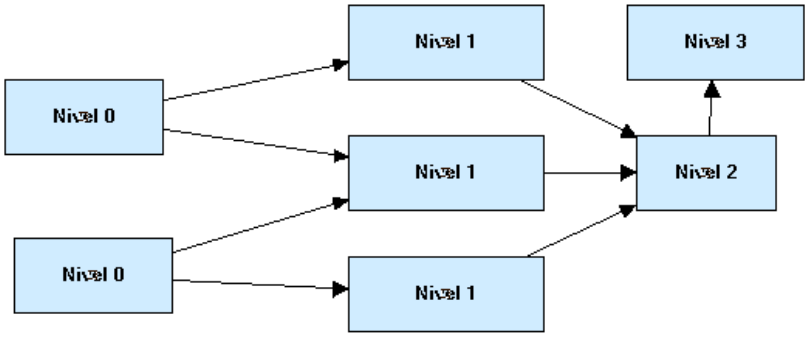


Fig. 2.1: Imagen de los niveles de un problema decisional. Fuente: Elaboración propia.

Con los hijos obtenidos por cada pareja se construye una lista y se pasa al proceso de mutación.

4- Con la lista de los nuevos individuos obtenidos por el cruzamiento de las parejas se pasa al proceso de mutación. Este proceso recorre el arreglo de hijos y por cada hijo se genera un número aleatorio y se comprueba si este se encuentra entre 0 y la probabilidad de mutación, que es un valor previamente definido, y es invariable igual a

[7] Profundidad en que se encuentra un indicador en el caso que se refiera al problema en general es el nivel máximo.

0.25, de ser positiva la comprobación el hijo en cuestión pasa a ser mutado por el proceso de mutación. Este ejecuta la mutación de individuo de la siguiente manera:

- ✓ Genera un número aleatorio entre 1 y la cantidad de indicadores del individuo que será la cantidad de indicadores a mutar por el hijo esto incluye a los indicadores que son causas raíces.
- ✓ Se escogen aleatoriamente de la lista de todos los indicadores de la ontología la cantidad de indicadores a mutar.
- ✓ Se escogen los indicadores del individuo que se van a sustituir de forma siempre que sean igual a la cantidad indicadores a mutar.
- ✓ Se sustituyen aleatoriamente los indicadores seleccionados del individuo por los indicadores cargados aleatoriamente de la ontología.

Ya mutados estos individuos no pasan a sustituir a los individuos originales, en este caso los hijos mutados se añaden a la lista de hijos ya obtenidos convirtiéndose en un hijo más obtenido por el AG, para así aumentar la cantidad de hijos obtenidos en total.

5- Después de la mutación los hijos pasan por el proceso de validación que eliminará los hijos que no son válidos.

6-Se almacenan los hijos obtenidos (en la BD) aumentando la BC y por tanto la población para la próxima corrida del algoritmo.

En la tabla 2.1 y en la Fig. 2.2 se puede apreciar los resultados obtenidos por el AG1 en tres ejecuciones consecutivas sobre una población inicial de 20 problemas decisionales.

Tabla 2.1 Resultado de las pruebas al algoritmo AG1.

Ejecución	Pob. Inicial	Cant. Parejas	Cant. Hijos Obtenidos	Cant. Hijos Válidos	Tiempo de Ejecución
1	20	190	710	410	00:04:22:40 (h: m: s: ms)
2	430	500	2237	1149	01:38:52:33 (h: m: s: ms)
3	2579	500	2625	1106	01:42:10:26 (h: m: s: ms)

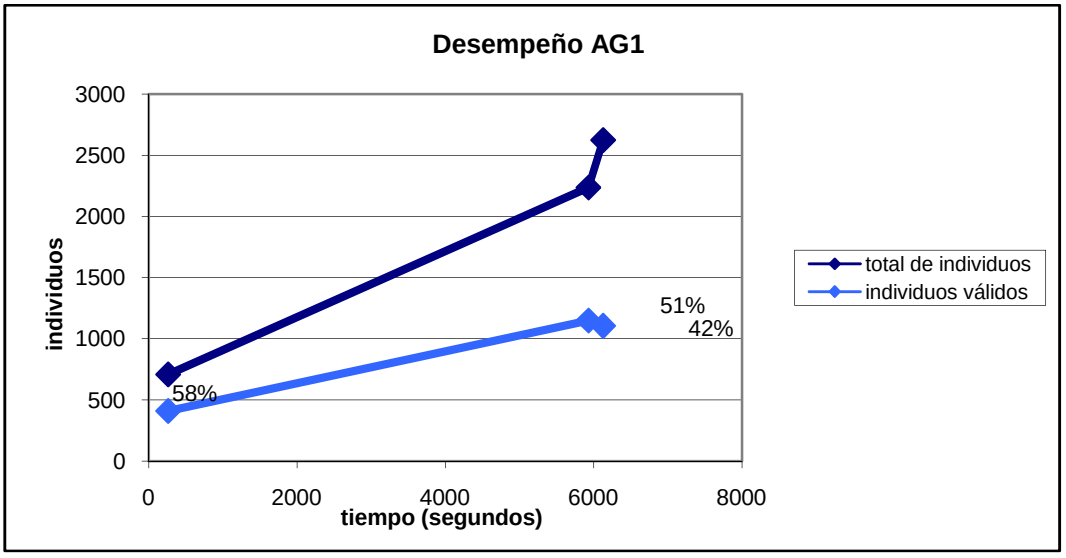


Fig. 2.2. Imagen del desempeño del AG1. Fuente: Elaboración propia

2.2.3 Algoritmo Genético AG2

Este algoritmo tiene como su principal objetivo maximizar los hijos válidos por ello se lleva a cabo un proceso de selección cuidadoso y garantizando con la selección de las parejas que los hijos tengan muy buenas probabilidades que sean válidos. Con una selección más específica de las parejas la cantidad de ellas formadas será mucho menor que la cantidad obtenida con el AG1. Este algoritmo tiene las siguientes características:

1- El proceso de selección de parejas se realiza uniendo los problemas semejantes. En este caso la semejanza de dos problemas se determina cuando tienen las mismas causas raíces [8]tienen el mismo nivel de profundidad.

2- En el proceso de cruzamiento los hijos heredan las mismas causas raíces de los padres.

3- Las causas raíces de los hijos a mutar se mantienen, no mutan.

Este algoritmo se ejecuta obedeciendo los pasos lógicos que a continuación se exponen:

[8] Las causas raíces son aquellos indicadores que son causas y nunca son efecto en una formulación de un problema decisional.

1- Se cargan las formulaciones de los problemas decisionales (están almacenados en la BD).

2- Se seleccionan las parejas que pasarán a cruzarse para obtener nuevos hijos a través del proceso de selección. Este realiza la selección de manera que forma parejas con todas las posibles combinaciones de individuos de la población siempre que estos sean semejantes, también deben cumplir este método es no formar parejas ya existentes y no formar parejas de un individuo con sí mismo. Con este proceso se obtiene una lista de parejas y se pasa al cruzamiento de estas parejas.

3- Con la lista de parejas a cruzar se ejecuta el proceso de cruzamiento de cada pareja con una probabilidad de 1, este es un proceso propio de las parejas y tiene como objetivo obtener hijos por cada pareja mientras que la cantidad de hijos ya obtenidos sea menor que 2 y la función de calidad sea verdadera. Cuando estas dos condiciones no se cumplen el método termina y devuelve los hijos obtenidos hasta el momento. Los nuevos individuos hijos se obtienen heredando aleatoriamente indicadores de cualquiera de los dos padres siempre que estos sean del mismo nivel del indicador hijo que se está formando o sea que para formar un nuevo indicador del individuo hijo este indicador va a ser igual a cualquier indicador de los padres siempre que sean del mismo nivel. Este procedimiento se realiza para obtener los indicadores de cada nivel del individuo hijo pero excluyendo a las raíces o el nivel 0 que en este caso como los padres tienen las mismas causas raíces el hijo hereda estas. Con las alternativas obtenidas por cada pareja se construye una lista y se pasa al proceso de mutación.

4- Con la lista de las nuevas alternativas se pasa al proceso de mutación este proceso recorre la lista de hijos y por cada hijo se genera un número aleatorio y se comprueba si este se encuentra entre 0 y la probabilidad de mutación que es un valor previamente definido y es invariable e igual a 0.25, de ser positiva la comprobación el hijo en cuestión pasa a ser mutado. Este proceso ejecuta la mutación de individuo de la siguiente manera:

- ✓ Genera un número aleatorio entre 1 y la cantidad de indicadores del individuo sin contar los indicadores que son causas raíces que será la cantidad de indicadores a mutar.

- ✓ Se escogen aleatoriamente de la lista de todos los indicadores de la ontología la cantidad de indicadores igual al número de indicadores a mutar.

- ✓ Se escogen los indicadores del individuo que se van a sustituir de forma aleatoria siempre que sean igual a la cantidad indicada de indicadores a mutar y que no sean indicadores que pertenezcan a las causas raíces.

- ✓ Se sustituyen aleatoriamente los indicadores seleccionados del individuo por los indicadores cargados aleatoriamente de la ontología.

Los individuos que mutaron no pasan a sustituir a los individuos originales en este caso las alternativas mutadas se añaden a la lista de hijos ya obtenidos en el cruzamiento para así aumentar la cantidad de hijos obtenidos en general.

En la tabla 2.2 y en la Fig. 2.3 se puede apreciar los resultados obtenidos por el AG2 en tres ejecuciones consecutivas sobre una población inicial de 20 problemas decisionales (los mismos utilizados por el AG1).

Tabla 2.2 Resultado de las pruebas al algoritmo AG2.

Ejecución	Pob. Inicial	Cant. Parejas	Cant. Hijos Obtenidos	Cant. Hijos Válidos	Tiempo de Ejecución
1	20	13	225	164	00:00:36:34 (h: m: s: ms)
2	184	500	3045	2968	00:10:50:91 (h: m: s: ms)
3	3152	500	3247	3192	00:12:49:82 (h: m: s: ms)

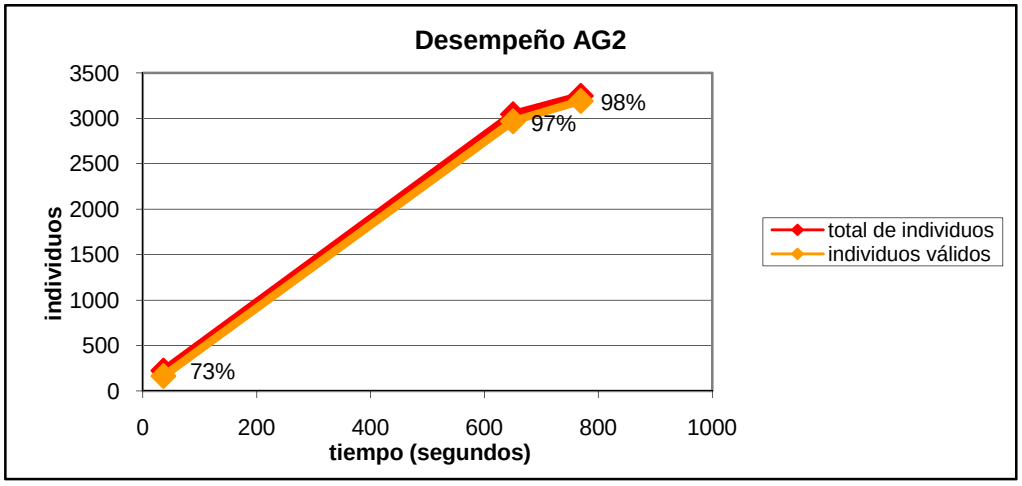


Fig. 2.3. Imagen del desempeño del AG2. Fuente: Elaboración propia.

2.2.4 Comparación a AG1 con AG2

Para hacer una comparación entre los AG se establecieron una serie de indicadores a medir. Estos indicadores fueron escogidos sobre la base de las características importantes de los algoritmos en su funcionamiento sobre el framework de asistencia decisional. Entre los requisitos o indicadores que se utilizaron para la elección están:

1- Cantidad total de hijos generados.

2- Cantidad de hijos válidos obtenidos.

3- Porciento de hijos válidos generados.

4- Tiempo total de ejecución.

5- Promedio de tiempo por hijo válido

Los indicadores antes mencionados se miden utilizando como entrada o comienzo una misma población inicial para todos los AG, así como se mantiene constante para cada uno de los AG el número de ejecuciones con las que fueron probados. Para el objetivo perseguido en la investigación se probó 3 ejecuciones.

Tabla 2.3 Comparación entre AG1 y AG2.

Algoritmo	Cant. Total de Hijos Generados	Cant. Total de Hijos Válidos	% de Hijos Válidos	Tiempo total de ejecución	Promedio de Tiempo x Hijo Válido
AG1	5572	2665	47.8 %	03:25:24:99 (h: m: s: ms)	00:00:04:62 (h: m: s: ms)
AG2	6517	6324	97.03 %	00:24:17:07 (h: m: s: ms)	00:00:00:23 (h: m: s: ms)

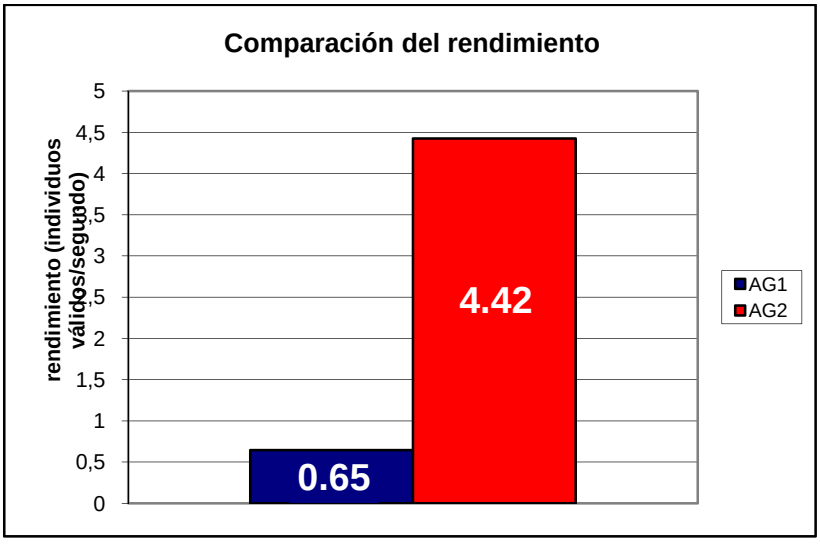


Fig.2.4. Comparación en cuanto a hijos válidos por segundo. Fuente: Elaboración propia.

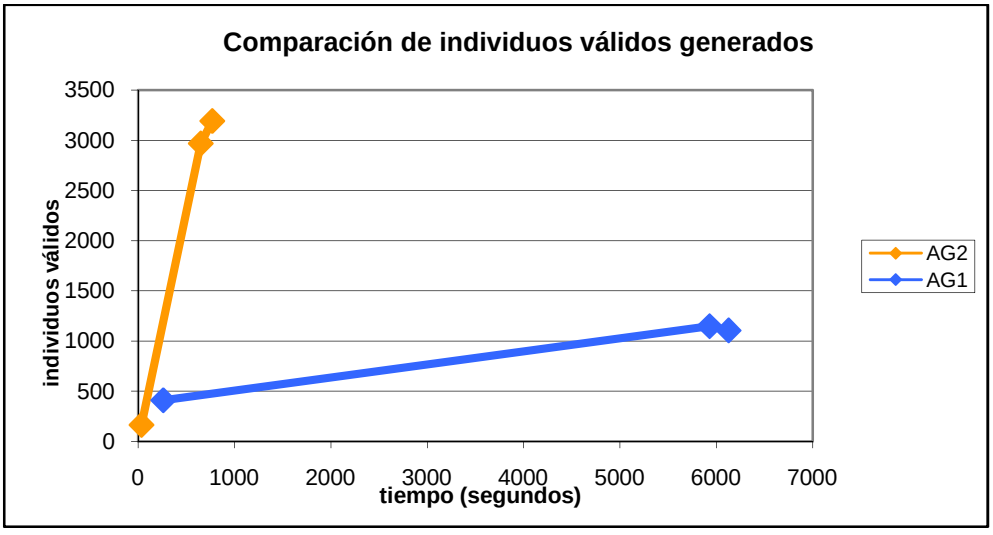


Fig. 2.5 Comparación de hijos validos contra tiempo de ejecución. Fuente: Elaboración propia.

Por los resultados obtenidos en las pruebas a los algoritmos AG1 y AG2 se llega a la conclusión que el mejor algoritmo para ser implementado sobre el framework decisional es el AG2 por la cantidad de hijos válidos que genera y la eficiencia con que lo hace.

2.3 Modelo del dominio

El modelado del dominio es una actividad clásica del análisis orientado a objetos. Con este se logra organizar y definir los conceptos importantes relacionados con el dominio que se estudia. Esta actividad permite obtener los objetos principales del dominio, sus atributos y relaciones.

En la Tabla 2.4 se muestran los conceptos que se consideran más importantes para el desarrollo del sistema que se pretende elaborar.

Tabla 2.4 Principales conceptos del dominio.

Conceptos	Definición
Problema decisional	Desviación de los valores planificados en las actividades de una CS y que son representados en forma de di-grafos.
Ontología SCOR	Conjunto de variables, conceptos e indicadores que caracterizan las actividades de la CS, basado en el Modelo de Referencia de las Operaciones en la Cadena de Suministro(SCOR)
Indicadores de calidad de la solución	Elementos a medir en la solución para determinar el grado de calidad de la misma.
Algoritmo Genético	Algoritmo que basa su funcionamiento en la simulación de los procesos de la genética.
Hijos	Resultados obtenidos por el AG y que representan nuevos problemas decisionales.

2.3.1 Diagrama de Clases del Modelo del Dominio

El diagrama de clases del dominio no constituye un esquema de clases de programación sino que presenta un diccionario visual de términos importantes del dominio.

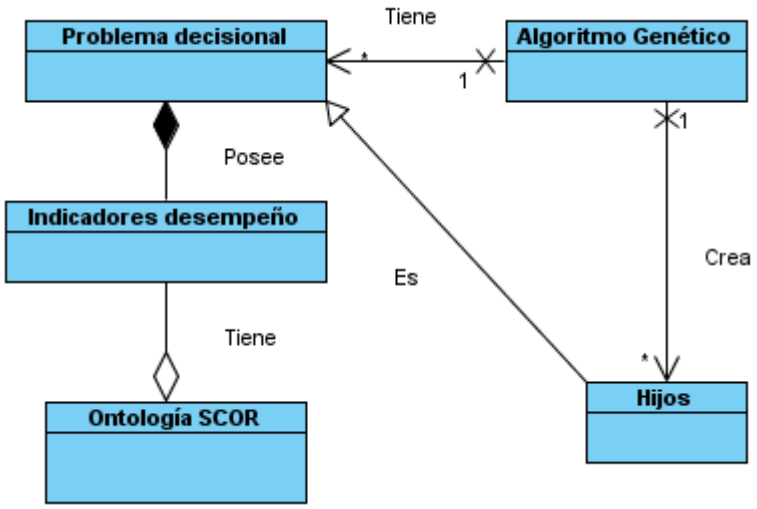


Fig. 2.6 diagrama de clases del dominio, representando las relaciones entre los principales conceptos definidos en la tabla 2.4. Fuente: Elaboración propia.

2.4 Requerimientos del Sistema

Los requerimientos son las condiciones o capacidades que un sistema debe satisfacer. Comprenden necesidades de información y control, funcionalidad del producto y comportamiento, rendimiento general, diseño, restricciones de la interfaz y otras necesidades especiales. El propósito de la gestión de requerimientos es establecer un entendimiento común entre el usuario y el desarrollador de software de los requerimientos del usuario que serán abordados por el proyecto de software y así satisfacer las necesidades de los clientes. Los requerimientos se clasifican en requerimientos funcionales y no funcionales (Wynekoop and Russo, 1997).

2.4.1 Requerimientos funcionales

Los requerimientos funcionales especifican acciones que el sistema debe ser capaz de realizar, sin tomar en consideración ningún tipo de restricción física. Los requerimientos funcionales especifican el comportamiento de entrada y salida del sistema y surgen de la razón fundamental de la existencia del producto.

R1- Cargar problemas decisionales.

R2- Generar nuevos problemas decisionales.

R3- Validar los nuevos problemas decisionales obtenidos.

R4- Guardar en una BD los problemas válidos obtenidos.

2.4.2 Requerimientos no funcionales del sistema

Los requerimientos no funcionales explican las características que de una u otra forma pueden limitar el sistema. Describen atributos del sistema o del ambiente en que va a ser utilizado. Los requerimientos no funcionales también añaden funcionalidad al producto, pues hacen que un producto sea fácil de usar, seguro e interactivo. Sin embargo, la razón fundamental de que esta funcionalidad sea parte del producto es brindarle a este las características deseadas y que le aportan calidad al software (Pérez, 2007).

Apariencia o Interfaz Externa

• La interfaz de la herramienta debe ser intuitiva, simple de usar, con uso de colores agradables al usuario.

• El lenguaje a utilizar será el inglés.

• El sistema dará aviso de forma clara y precisa de cada acción indebida ejecutada por el usuario.

Rendimiento

• El sistema debe ser eficiente y preciso en la respuesta al usuario para cada acción que este ejecute.

Portabilidad

• El sistema será multiplataforma, o sea que podrá ser utilizado tanto en Windows, como Linux o cualquier sistema operativo que tenga la máquina virtual para Java.

Software

• El sistema requiere JDK 1.5 o superior en la plataforma que se utilice.

• El sistema requiere gestor de base de datos PostgreSQL 8.0 o superior.

Hardware

• El sistema requiere procesador Intel Pentium 4 a 1.8 MHz mínimo.

• El sistema requiere 256 MB de memoria RAM mínimo.

2.5 Modelo de Casos de Uso del Sistema

El modelado de los casos de uso del sistema se realiza definiendo los casos de uso, estos son requerimientos funcionales descritos desde la perspectiva de usuarios del sistema. Los casos de uso definidos para el sistema propuesto se organizaron en un paquete, el principal que contiene los casos de uso relacionados con los principales requerimientos del sistema.

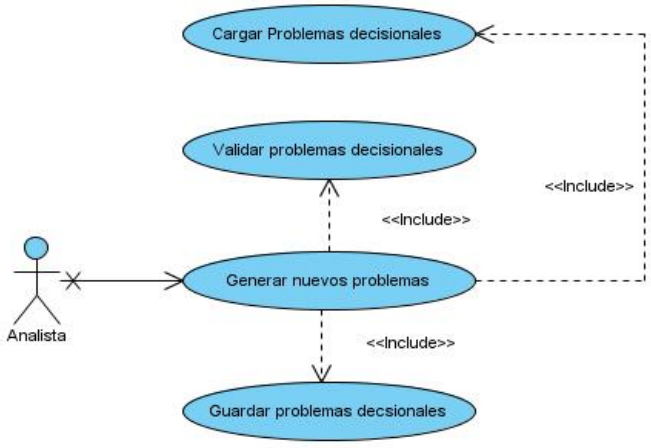


Fig. 2.7 diagrama de casos de uso del sistema. Fuente: Elaboración propia.

2.5.1 Descripción de los principales Casos de Uso del Sistema

Tabla 2.5 descripción del caso de uso Cargar problemas decisionales.

Nombre del caso de uso	Cargar problemas decisionales
Actores	
Propósito	Cargar los problemas decisionales desde una base de datos.
Referencias	R1
Resumen	El caso de uso lo inicia cuando se manda a cargar los problemas desde una base de datos ya que estos representan a los individuos de la población inicial. El caso termina cuando son cargados los problemas decisionales.
Precondiciones	Tener conexión con una BD en un servidor Postgres SQL.
Pos condiciones	Los problemas generados pasan a formar la nueva población.

Tabla 2.6 Descripción del caso de uso Generar nuevos problemas decisionales.

Nombre caso de uso	Generar nuevos problemas decisionales
Actores	Analista inicia
Propósito	Generar nuevos problemas a partir de problemas ya creados.
Referencias	R1,R3
Resumen	El caso de uso se inicia cuando el Analista accede a la opción generar nuevos problemas. El caso de uso termina cuando se generan los nuevos problemas decisionales.
Precondiciones	Haber cargado los problemas que forman parte de la población.

Pos condiciones	Una vez generados se pasa al proceso de validación.

Tabla 2.7 Descripción del caso de uso validar problemas decisionales.

Nombre del caso de uso	Validar problemas decisionales
Actores	
Propósito.	Determinar si un problema decisional esta semánticamente bien formulado
Resumen	El caso de uso se inicia cuando el AG solicite la verificación de la valides de un problema decisional obtenido. Termina cuando se informa la valides del problema en cuestión.
Referencias	R3
Precondiciones	Tienen que proporcionarle como entrada un problema decisional.
Pos condiciones.	Retorna verdadero en caso de ser válido en caso contrario falso.

Tabla 2.8 Descripción del caso de uso Guardar problemas válidos.

Nombre del caso de uso	Guardar problemas válidos
Actores	
Propósito	Salvar en una BD los problemas válidos obtenidos.
Resumen	El caso de uso se inicia cuando el AG termina su ejecución y valida los problemas hijos entonces termina cuando se guardan los problemas válidos resultantes.
Referencias	R4
Precondiciones	Haber corrido el Ag para la obtención de nuevos problemas.

Pos condiciones	Los nuevos problemas quedan salvados

2.6 Modelo del Diseño

Luego que se definen los casos de uso del sistema y se realiza su análisis existe una actividad muy importante en la modelación de un software, esta es el Modelo del Diseño. Con el modelo del diseño se logra una comprensión más profunda de los requisitos no funcionales del software y sus restricciones. Se crea además una entrada adecuada para las actividades de implementación.

2.6.1 Diagramas de Clases del Diseño

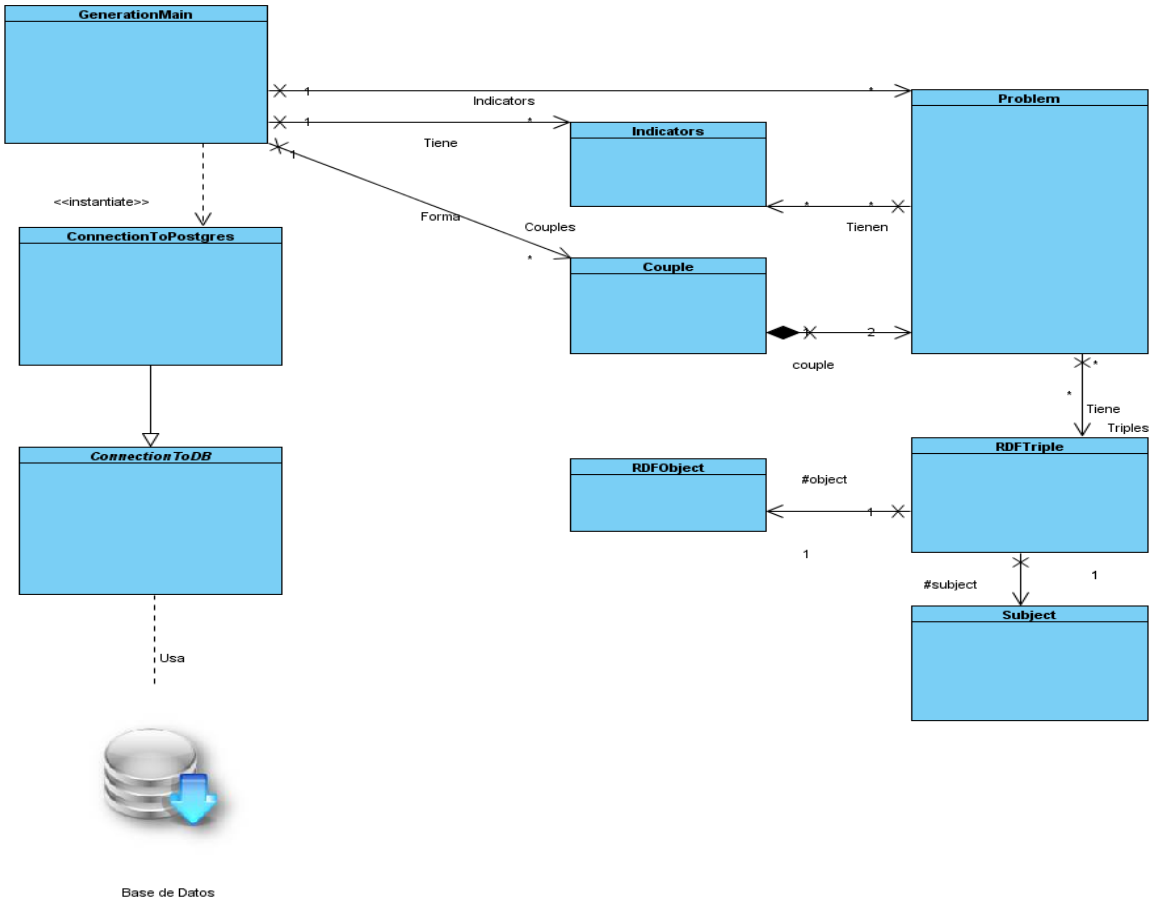


Figura 2.8 Diagrama de clases del diseño. Fuente: Elaboración propia.

2.6.2. Diagramas de secuencia

Un caso de uso contempla una secuencia de acciones, y si se considera el interior del sistema se puede apreciar como los objetos que integran al mismo interactúan

mediante mensajes llevando a cabo el caso de uso. El modelo del diseño provee un artefacto para apreciar esta interacción, denominado diagrama de secuencia.

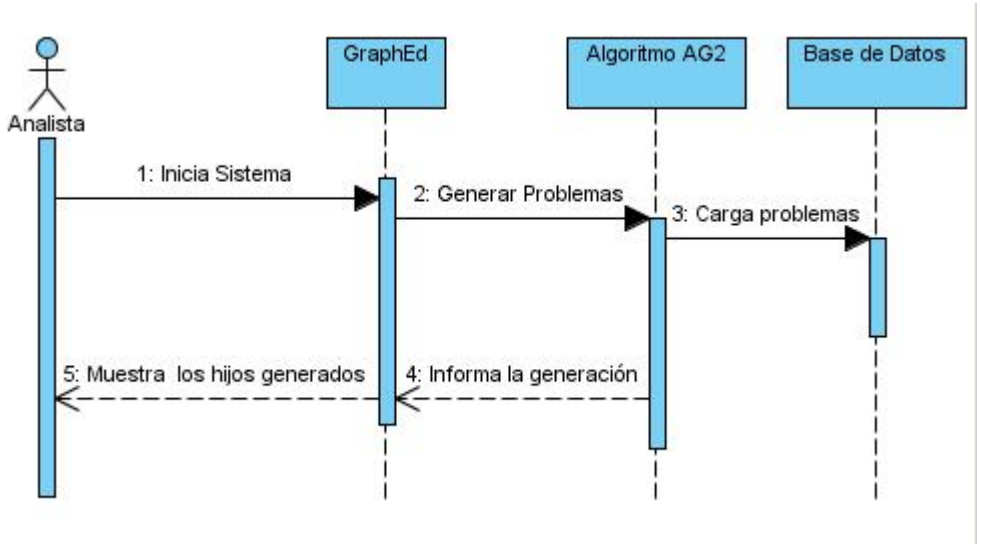


Fig. 2.9. Diagrama de secuencia para caso de uso Generar nuevos problemas decisionales. Fuente: Elaboración propia.

2.6.3 Diseño de la Base de Datos

El diseño de la base de datos permite definir de antemano las tablas que estarán incluidas en esta, así como las relaciones que se establecen entre ellas. Una correcta normalización de la base de datos se puede lograr en esta etapa, cuando el desarrollador se enfoca solamente en el diseño lógico, y no tiene en cuenta aún la implementación de la persistencia. El diseño de la base de datos se realiza haciendo uso de un diagrama de clases persistentes. Se entiende por clase persistente, aquella que representa datos que mantienen su valor en el espacio y el tiempo. Las relaciones que existen entre estas y las clases no persistentes, definen los accesos físicos que se realizarán a la base de datos desde la aplicación (Wynekoop and Russo, 1997).

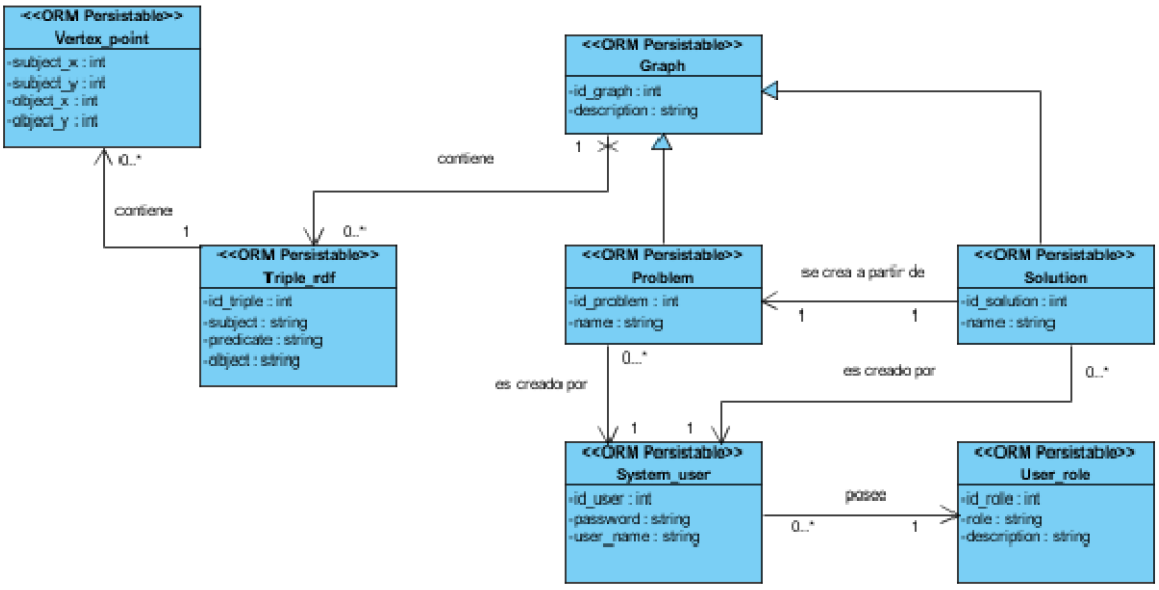


Fig. 2. 10 Diagrama de Clases Persistentes. Fuente: tomado de:(Velázquez, 2009)

2.7 Modelo de Implementación

En la actividad de implementación se contempla el sistema desde la perspectiva de sus componentes, es decir, ficheros de código fuente, scripts, fichero de código binario, ejecutables etc. Aunque la mayor parte de la arquitectura del sistema es capturada durante el modelo del diseño el objetivo principal de la implementación es desarrollar la arquitectura y el sistema como un todo (Wynekoop and Russo, 1997)

2.7.1 Diagrama de componentes

Los diagramas de componentes es uno de los artefactos que integran el modelo de implementación. Este está conformado por los componentes que intervienen en el sistema. Estos componentes empaquetan clases del diseño, además de representar documentos, librerías de código, tablas de base de datos etc. Cada componentes en el modelo de diseño se mapea al modelo de implementación. La fig. Muestra la dependencia de trazas desde los componentes de implementación hasta las clases del diseño. Además se muestra la dependencia de compilación entre los componentes de la implementación.

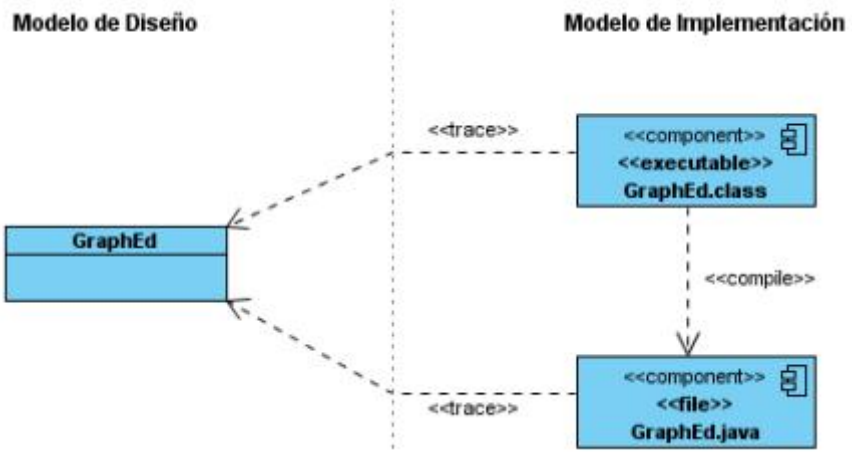


Fig.2.11. Dependencia de trazas entre componentes y clases del diseño, y dependencia de compilación entre componentes. Fuente: Elaboración Propia.

A continuación se muestra un diagrama de componentes como ejemplo de cómo cada clase del diseño se convirtió en un componente de la implementación.

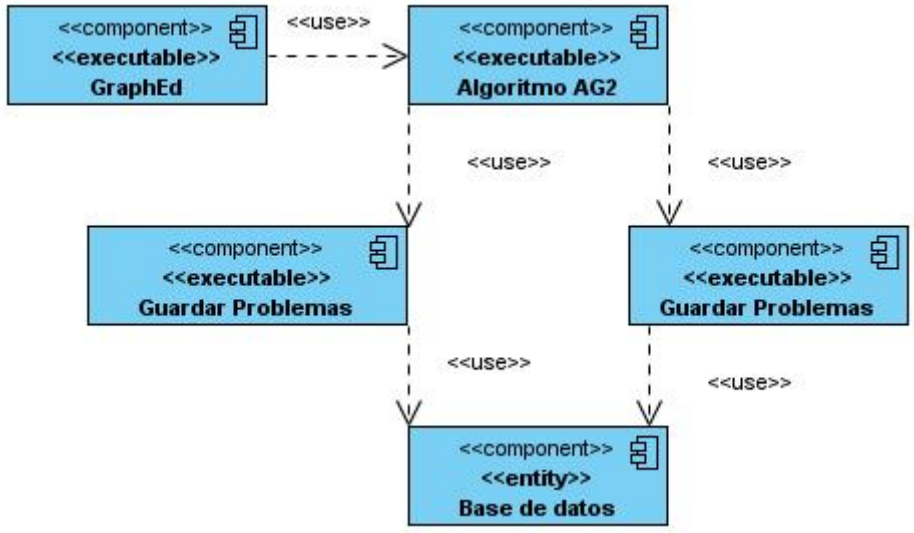


Fig. 2.12. Diagrama de componentes para el caso de uso Generar nuevos problemas Decisionales. Fuente: Elaboración propia.

2.7.2 Diagrama de Despliegue

Los diagramas de despliegue muestran la configuración de elementos de procesamiento en tiempo de ejecución y los componentes, procesos, y objetos de software que viven en ellos.

Fig. 2. 12 Diagrama de despliegue. Fuente: Elaboración propia.

La herramienta cliente se implantará en cualquier computadora cliente y podrá ser usada si posee conexión con el servidor que aloja el SGBD PostgreSQL.

2.8 Prueba de la Herramienta Implementada

En la prueba de la herramienta implementada se describe como son probados los aspectos específicos del sistema. Se tomó el caso de uso del sistema Generar nuevos Problema para realizar las pruebas correspondientes, debido a su relevancia en el funcionamiento de la herramienta. Para probar este caso de uso fue necesario definir dos problemas decisionales.

Caso de prueba: Generar nuevos problemas decisionales.

Entrada:

- Se desea cargar de la BD los problemas decisionales almacenados hasta el momento.
- Se desea ejecutar el AG2 para generar con este nuevos problemas decisionales válidos
- Se desea almacenar los nuevos problemas generados por el AG2.

Resultados esperados:

1- Se cargan todos los problemas de la BD
2- Se ejecuta el AG2 obteniendo una serie de nuevos problemas decisionales.
3- Se guardan en la BD estos nuevos problemas.

Procedimiento de prueba: Generar nuevos problemas decisionales.

1. Seleccione la opción “Generate new problems” en el menú Data de la ventana principal del sistema

2- Espere los resultados del AG2.

4- Compruebe gráficamente los nuevos problemas seleccionando la opción Load problem del menú Data. Se abre la ventana de cargar problemas.
5- Seleccione el nuevo problema generado por el AG2 que desee ver por la fecha actual.
6- Oprima el botón Load. Se mostrará gráficamente el nuevo problema.

La prueba fue realizada satisfactoriamente obteniendo los resultados esperados.

2.9 Valoración de Sostenibilidad según su impacto Social, Económico, Tecnológico y Ambiental.

Para desarrollar un software es importante definir desde las primeras etapas de su desarrollo el impacto social, económico, tecnológico y ambiental que este tendrá. A continuación se expondrá el impacto de la herramienta propuesta como solución en cada una de las dimensiones mencionadas anteriormente.

2.9.1 Administrativa

En la dimensión administrativa se valora si la solución planteada ahorra recursos, se tienen presente los gastos implicados para su desarrollo, el impacto en la calidad de la producción y los servicios, así como otros aspectos que garanticen la sostenibilidad administrativa de la solución.

El producto informático (PI) resultante de la presente investigación pretende ser una herramienta para la construcción de bases de conocimiento de problemas decisionales en el contexto de la toma de decisiones estratégico-logística con enfoque de CS. Para la implementación del mismo se utilizarán herramientas Open Source por lo que el costo de la aplicación se verá reducido en este aspecto. Con el PI se podrá introducir, analizar, guardar y reutilizar información que anteriormente podía haber sido tratadas a través de papel. Por esta razón con la aplicación del PI se ahorrará tinta y papel además del tiempo a la hora de procesar la información. El producto tendrá gran impacto en la calidad de los servicios y producción de la empresa u organismo que lo

implante en la medida que utilice su potencialidad para generar nuevos conocimientos partiendo de los problemas de toma de decisiones ya conocidos. En este sentido el producto puede ser utilizado para diversas situaciones y los nuevos conocimientos pueden ser almacenados en una base de datos para su posterior consulta. Esto provee la posibilidad de analizar una situación que no se ha dado antes y tener la solución disponible, ahorrando considerablemente el tiempo y estandarizando además los tipos de problemas y las soluciones dadas a los mismos, lo que permitirá un aumento de la calidad gerencial en cuanto a eficacia y eficiencia.

La inmensa mayoría de las decisiones tomadas en una empresa tienen que ver directa o indirectamente con el dinero. El uso del producto traerá ingresos en la medida que se utilice para generar nuevos problemas de toma de decisiones y sus soluciones, no solo que consideren recursos monetarios directamente sino cualquier otra decisión que involucrará a largo plazo ahorro de recursos. Uno de los mayores problemas a los que se han enfrentado las empresas a lo largo del tiempo es el de la gestión y uso de toda la información que se genera.

Es el caso de que en un puesto determinado de trabajo, muchas veces en la administración, existen personas “claves” que a lo largo del tiempo han acumulado conocimiento que no se ha plasmado en documentos o en procedimientos sino que se encuentra en su intelecto. Cuando esas personas salen de la empresa se crea un caos pues los que se suceden tienen que invertir tiempo y dinero para recuperar todo el conocimiento perdido. Esto se traduce directamente en recursos, viajes de capacitación, postgrados. Es decir que el software no solo será de gran ayuda para la asistencia a la toma de decisiones sino que permitirá que la empresa deje “huellas” de las decisiones tomadas y asistirá a la toma de nuevas decisiones sin que antes hubiesen ocurrido partiendo de situaciones similares que si han ocurrido lo que se traducirá en una correcta gestión de la información que contribuirá a que el problema antes descrito no se produzca o se produzca en menor medida. Para la elaboración del producto, todo software y herramienta que se utilizó es libre. Por lo que se clasifica al PI como sostenible en esta dimensión.

2.9.2 Socio-Humanística

El PI trata de dar solución a la compleja situación de la gestión estratégica empresarial, proveyendo un estándar común para solucionar variadas situaciones que requieren toma de decisiones, que se pueden encontrar en el ámbito empresarial. Según las soluciones de problemas empresariales son generalmente similares esto puede resultar en la generalización del sistema. El PI pretende que la generación de los nuevos problemas sea gestionada a través de una interfaz gráfica agradable y con un tiempo de respuesta lo más rápido posible de acuerdo a la complejidad del algoritmo concebido, permitiendo una mejor satisfacción para el usuario. Con el PI no se genera ni se disminuyen las fuentes de empleo, lo que si facilita el trabajo del personal existente. El PI trabajará con datos y situaciones reales por lo que se demanda del usuario responsabilidad y honestidad para que el resultado sea el más óptimo para la empresa u organización. En la medida que el producto sea eficiente al generar nuevos problemas empresariales será aceptado en este ámbito. Se buscará que el PI tenga una interfaz visual agradable e intuitiva, que cuente con manuales de usuario y ayudas pertinentes para cada situación, se implementarán ejemplos para su uso. En el ámbito científico el PI se llevó a la práctica en las investigaciones de una Tesis de maestría y será pionero en Cuba en su tipo como sistema de generación de bases de conocimientos para la asistencia a la toma de decisiones con enfoque a CS. Por lo que en este aspecto el sistema es sostenible.

2.9.3 Ambiental

En cualquier empresa u organismo de este país se trabaja con el papel, recurso derivado de los árboles. Cualquier esfuerzo que se realice a favor de reducir el empleo de papel es loable. El PI resultante apuesta por reducir el uso de este recurso. Cuenta con una base de datos que permitirá almacenar los problemas que ya han sido modelados y los nuevos generados por el AG creando así una BC logrando tener en soporte digital información sin necesidad de tenerla en papel. Al ahorrar papel se ahorrará también recursos utilizados en la impresión de información como tinta, cinta y otros. Con el transcurso del tiempo y el uso frecuente de la herramienta, se podrá contar con una BC almacenados en una BD que posee un gran número de problemas

modelados. Esto significará ahorro de tiempo en el análisis y búsqueda de soluciones a los nuevos problemas que se presenten. Lo que demuestra que el PI es sostenible.

2.9.4 Tecnológica

El PI tiene ciertos requisitos debidos a los cuales un usuario sin experiencia pudiera tener problemas a la hora de implantarlo. Por esta razón se proveerá al PI de una buena guía de instalación por la cual el usuario final podrá instalar la aplicación.

La aplicación podrá ser ejecutada desde una máquina cliente con solo tener instalado el ambiente de ejecución de Java. De esta forma se garantiza que el sistema pueda ser utilizado en la gran mayoría de las organizaciones nacionales. El gestor de base de datos que se ha de utilizar es de libre distribución así como las librerías de código usadas para desarrollar el PI. Se garantiza también que el sistema se puede usar en múltiples Sistemas Operativos debido a que está desarrollado en el lenguaje de programación Java, el cual posee esta característica. Debido a la existencia de una base de datos para almacenar los problemas generados por el AG, el mayor factor de riesgo es el estado de las redes y la velocidad de la máquina. En caso de fallar la conexión a la base de datos podría haber pérdida de información y no se podría guardar los nuevos problemas obtenidos. En caso de la velocidad de la máquina se necesita una máquina con al menos 1024 MB de memoria RAM.

El PI es resultado de la integración de varias tecnologías que actualmente están en constante desarrollo y mejoras. Es perfectamente adaptable a las mejoras que se le puedan hacer a las tecnologías usadas, mejorando también así las prestaciones del mismo. En el campo de estudio de los sistemas de asistencia a la toma de decisiones hay mucho que investigar todavía. Muchos son los esfuerzos que se están realizando en Cuba y el mundo para desarrollar métodos, técnicas y algoritmos en esta rama de la Inteligencia Artificial. El PI que se valora es un aporte más a este campo. Es novedoso el uso de los AG y de una ontología desarrollada por Concepto de la filosofía utilizado actualmente en las ciencias informáticas como artefacto de ingeniería para describir ciertos dominios a través de conceptos y asunciones explicitas asociadas a dichos conceptos.(Sáez Mosquera, 2008). Por lo que se concluye que el PI obtenido es sostenible en esta dimensión también.

De acuerdo con lo analizado en las diferentes dimensiones por las que se evaluó la sostenibilidad del sistema se puede arribar a la conclusión que el PI es sostenible.

2.10 Conclusiones Parciales

- El software utilizado para la modelación en las etapas de desarrollo de la herramienta resultó ser muy eficiente y queda disponible para su utilización en sistemas similares.

- La capacidad generativa de los AG demostraron ser un elemento muy útil en la generación de nuevos conocimientos a partir de los ya existentes en la BC.

- Después de una serie de pruebas y comparaciones se opta por implementar el AG2 sobre el framework para la asistencia decisional.

-Con las pruebas realizadas a la herramienta implementada, se arribó a la conclusión de que la misma posee un buen rendimiento en su funcionalidad.

-Sobre la base del análisis realizado cuando se valoró la sostenibilidad del software según su impacto económico, social, tecnológico y ambiental se concluye que el mismo es sostenible en el contexto que se utilice.

Conclusiones Generales

- Se logró un amplio levantamiento de los fundamentos teóricos que rigen la gestión de conocimiento mediante ontologías y su aplicación en las ciencias informáticas. Así como también la generación de nuevos individuos usando técnicas de Inteligencia Artificial como los AG.
- Se implementaron Algoritmos Genéticos que en el contexto definido por la ontología SCOR permiten generar y validar nuevos conocimientos a partir de las formulaciones de problemas decisionales y para la creación de una BC que permitirá la asistencia a la toma de decisiones con enfoque de CS. Cumpliéndose así el objetivo de esta investigación.
- La herramienta informática implementada es sostenible y tiene condiciones para evolucionar y brindar mejores servicios para el apoyo a la toma de decisiones con un enfoque estratégico en el ámbito de las Cadenas de Suministro.
- Los actuales desarrollos en la modelación, representación e intercambio de información y conocimiento impulsados por la revolución en las tecnologías digitales, permite tratar cada eslabón de una Cadena de Suministro como fuentes de conocimientos sobre una base semántica común.
- Se comprobó la eficiencia de los AG para la generación de nuevos individuos logrando un excelente resultado en cuando a los nuevos conocimientos generados.

Recomendaciones

Sobre la base de la investigación realizada y la experiencia acumulada durante el proceso de realización de este trabajo, se presentan a continuación una serie de recomendaciones para una posterior mejora y construcción de nuevas versiones:

- Aplicar AG para también cruzar las soluciones de los problemas almacenados y así obtener posibles soluciones para los problemas que puedan ser generados a partir del cruzamiento de dichos problemas.
- Implementar un método para convertir a los hijos inválidos por no tener todas sus triplas o relaciones unidas en un mismo grafo en válidos uniendo todas sus relaciones en un solo grafo aumentando la cantidad de hijos válidos generados.
- Tener en cuenta al formar nuevas relaciones para los hijos las inferencias sobre las relaciones de los padres para así formar hijos con más calidad y solidez.

Glosario de Términos

Base de Conocimientos: Compuesto de datos, reglas, procedimientos y relaciones que deben tomarse en cuenta para generar valor u obtener un resultado.

Base de datos: Una base de datos de consta de una colección de tablas que contienen datos y otros objetos, como vistas, índices, procedimientos almacenados y desencadenadores, que se definen para poder llevar a cabo distintas operaciones con datos. Los datos almacenados en una base de datos suelen estar relacionados con un tema o un proceso determinados como, por ejemplo, la información de inventario para el almacén de una fábrica.

Cadena de Suministros: Compleja serie de procesos de intercambio o flujo de materiales y de información que se establece tanto dentro de cada organización o empresa como fuera de ella, con sus respectivos proveedores y clientes.

Desktop: Se refiere a escritorio en español, usada como adjetivo de aplicaciones, indica una aplicación que consiste en un fichero que puede ser ejecutado dependiendo sólo de él mismo.

Framework: En el desarrollo de software, es una estructura de soporte definida mediante la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros software para ayudar a desarrollar y unir los diferentes componentes de un proyecto.

Hardware: Componentes electrónicos, tarjetas, periféricos y equipo que conforman un sistema de computación; se distinguen de los programas (software) porque éstos le indican a los componentes mencionados lo que deben hacer.

Referencias Bibliográficas

ABAD, M. T. (2003) Ontologías. España, Inteligencia Artificial – FIB-UPC.

ANDREA BELLANDI, B. F., VALERIO GROSSI, ANDREA ROMEI (2007) Ontology-driven association rules extraction: a case of study. *Contexts and Ontologies Representation and Reasoning.* Roskilde University, Denmark.

CATER-STEEL, A. & AL-HAKIM, L. (2008) *Information Systems Research Methods, Epistemology, and Applications*, Information Science Reference.

CECCARONI, L. (2001) ONTOWEDSS - An Ontology Based Environmental Decision-Support System for the management of wastewater treatment plants. *Inteligencia Artificial.* Cataluña. España, Universidad Politecnica de Cataluña.

CESPÓN CASTRO, R., SÁEZ MOSQUERA, I., HERNÁNDEZ PÉREZ, G. & KNUDSEN GONZÁLEZ, J. A. Sistemas de Gestión Logística.

CHRIS WELTY, J. W. M. (2006) Towards Knowledge Acquisition from Information Extraction.

D. MARTINEZ, M. T., J. MIRA (2002) Knowledge base development. 7.

ENCYCLOPEDIA, P. M. (2005) Definition of: knowledge base.

ESCHENBACH, C. & GRUNINGER, M. (Eds.) (2008) *Formal Ontology in Information Systems*, IOS Press.

FENSEL, D. (2001) *Ontologies: Silver Bullet for Knowledge Management and Electronic Commerce*, Springer.

FERRER, J. (2007) BASES DE CONOCIMIENTO.

FIRESTONE, J. M. (1998) Basic Concepts of Knowledge Management.

GRACIELA ELISA BARCHINI, M. M. Á., DIANA PALLIOTTO, SUSANA HERRERA Y PAOLA BUDÁN (2006) Ontologías en los sistemas de información. Conocimiento. .

GRIFFITHS, A. D., HARRISON, M. D. & DEARDEN, A. M. (1999) Case-based reasoning system for knowledge mediation. IN JOHNSON, M. A. S. C. (Ed.) *Human - Computer Interaction.* Edinburgh, UK, IOS Press on behalf of the International Federation for Information Processing (IFIP).

GUARINO, N. & WELTY, C. (2000) Ontological analysis of taxonomic relationships. IN PUBLISHED IN, L., A., AND STOREY, V., EDS., (Ed.) *Proceedings of ER-2000: The International Conference on Conceptual Modeling.October, 2000. Springer-Verlag LNCS.* Berlin.

JOHN MYLOPOULOS, E. E. L. (1984) An overview of knowledge representation.

MARTIN HEPP, P. D. L., ALDO DE MOOR,YORK SURE (2008) *Ontology Management.Semantic Web, Semantic Web Services, and Business Applications*.

MARTÍNEZ DELGADO, E., ACEVEDO SUARÉZ, J., LAUZARDO RICO, L. & BERMÚDEZ LAO, R. (2003) Modelo de ayuda a la toma de decisiones y el análisis integral de los sistemas de producción y distribución. IN MARX GÓMEZ, J. & RAFAEL, E. A. (Eds.) *MT´2003.* La Habana, Cuba, AMSE.

MEAGHER, K. & WAIT, A. (2004) Decision making within organization.

MITCHELL, M. (1999) An Introduction to Genetic Algorithms.

NORVIG, S. J. R. A. P. (1995) Artificial Intelligence

A Modern Approach.

O'REILLY, U. (2002) Genetic Programing Series.

PÉREZ, R. S. (2007) Metodología para medir la calidad de software. Holguín, Oscar Lucero Moya.

SÁEZ MOSQUERA, I. (2008) Procedimientos y arquitectura de apoyo para la asistencia decisional en procesos estratégicos de gestión logística. *Departamento de Ingeniería Industrial.* Santa Clara, Universidad Central "Marta Abreu" de Las Villas.

VELÁZQUEZ, A. C. (2009) Sistema para la Gestión de Información en la Planeación

Estratégico Logística, con enfoque de Cadena de Suministros. Universidad de Holguín Oscar Lucero Moya, Armando Carracedo Velázquez.

WHITLEY, D. (1993) A Genetic Algorithm Tutorial. 38.

WIKIPEDIA (2009a) Base de Conocimiento.

WIKIPEDIA (2009b) Genetic fuzzy systems.

WYNEKOOP, J. L. & RUSSO, N. L. (1997) Studying systen development methodologies: an examination of research methods. *Info System,* 4, 47-65.

ZILLI, A., DAMIANI, E., CERAVOLO, P., CORALLO, A. & ELIA, G. (2008) *Semantic Knowledge Management: An Ontology-based Framework*, Information Science Reference.

Bibliografía

ANDERSON, J. D. (2003) Organization of knowledge. IN STURGES, J. F. P. (Ed.) *International Encyclopedia of Information and Library Science. 2nd.* London.

ANNE CREGAN, M. M., DENNY VRANDE, SEAN BECHHOFER (2006) Pushing the limits of OWL, Rules and Protege A simple example.

ARIZMENDI, R., NARANJO, G. & SORIANO, G. (2004) Knowledge management systems at consulting firms ISYE: 6231- Design of Human Integrated Systems.

ARNOTT, D. R. (1998) A Framework for Understanding Decision Support Systems Evolution. IN SYSTEMS, S. O. I. M. (Ed.). Monash University, Melbourne, Australia, Monash University.

B.AYERS, J. (Ed.) (2001) *Making Supply Chain Management Work: Design, Implementation, Partnerships, Technology, and Profits*, Auerbach Publications.

BAGCHI, P. K., HA, B. C., SKJOETT-LARSEN, T. & SOERENSEN, L. B. (2005) Supply chain integration: a European survey. *The International Journal of Logistics Management,* 16**,** 275 - 294.

BERNARD, R. (1993) Decision Science or Decision aid Science. *European Journal of Operational Research*.

BIRMINGHAM, W. P., D'AMBROSIO, J. G., DARR, T. & DURFEE, E. (2000) Coordinating Decision Making in Large Organization. IN MICHIGAN, T. U. O. (Ed.). Michigan, USA.

BOCKSKOPF, V., DIEKMANN, A., DRIEHAUS, S., EHLICH, M., ERBS, F., FRÄGER, C., GAUBISCH, T., GÖTZ, O., GUHL, R.-T., HARMS, M., HEß, T., HILFERT, S., KIEL, E., KIELE-DUNSCHE, M., KOHLMEIER, D., LÜCHTEFELD, K., LÜLSDORF, S., SCHARRENBACH, R., SIEVERS, R., STÖWER, M., TIELKER, U., TOEBERG, M., VOGT, J. P., WEBER, K.-H., WENDT, H.-J. & WITTE, S. (2008) *Drive Solutions. Mechatronics for Production and Logistics*, Springer.

COHEN, D. (Ed.) (1999) *Sistemas de Información para la Toma de Decisiones*.

CORDI, V., LOMBARDI, P., MARTELLI, M. & MASCARDI, V. (2005) An ontology-based similarity between sets of concepts. *WOA 2005.* Italy.

CHIA-HUNG LIN, J.-S. H., MARTIN DOERR (2008) Issues in an inference platform for generating deductive knowledge: a case study in cultural heritage digital libraries using the CIDOC CRM. *Springer-Verlag Berlin Heidelberg*.

DEVEDZIC, V. (2001) Knowledge Modeling - State of the Art. IN DEPARTMENT OF INFORMATION SYSTEMS, F.-S. O. B. A. (Ed.). Belgrade, Yugoslavia, University of Belgrade.

ENGLISH, J. & GEORGE, R. (2002) Database support for collaborative battle planning. *World Automation Congress, 2002. Proceedings of the 5th Biannual,* 14**,** 429-434.

FIDJELAND, M. K. (2006) Distributed Knowledge in CBR Knowledge Sharing and Reuse within the Semantic Web. *Department of Computer and Information Science.* Norwegian. Noruega, Norwegian University of Science and Technology.

FORMAN, E. & SELLY, M. A. (2000) Capítulo 1: Management Decision-Making Today. IN UNIVERSITY, G. W. (Ed.) *Decision by Objective.* Washinton. USA, George Washinton University.

HERBERT, S., B. DANTZIG, G., HOGARTH, R., PIOTT, C., RAIFFA, H., SCHELLING, T., SHEPSLE, K., THAIER, R., TVERSKY, A. & WINTER, S. (1958) Decision making and Problem solving. IN SCIENCES, N. A. O. (Ed.).

HERNÁNDEZ, A. G. (2004) Aprendizaje Automático: Algoritmos genéticos. IN ARTIFICIAL, M. ı. E. I. (Ed.). Mèxico, Universidad Veracruzana. Facultad de Física e Inteligencia Artificial.

HJØRLAND, B. (2006) Units or entities in knowledge organization (KO). What is being organized? IN CAPURRO, R. (Ed.). Denmark.

IBM CORPORATION (2005) Ontology definition Metamodel. *OMG, 22 August 2005.* IBM, Sandpiper Software.

JIMÉNEZ SÁNCHEZ, J. E. (2004) Los factores críticos de éxito de la Cadena de Suministro. IN TRANSPORTE, S. D. C. Y. (Ed.). Instituto Mexicano del Transporte.

JOANNA JÓZEFOWSKA, A. Ł., TOMASZ ŁUKASZEWSKI (2006) Towards Discovery of Frequent Patterns in Description Logics with Rules

KASHYAP, V., BUSSLER, C. & MORAN, M. (2008) *The Semantic Web: Semantics for Data and Services on the Web (Data-Centric Systems and Applications)*, Springer.

KOKKORAS, F., BASSILIADES, N. & VLAHAVAS, I. (2004) Modelling Information Extraction Wrappers with Conceptual Graphs. *Proc.(2nd Volume) 3rd Panhellenic Conference on Artificial Intelligence (SETN'04) 5-8 May 2004*.

KOLODNER, J. L. (1991) Improving Human Decision Making through Case-Based Decision Aiding. *AI Magazine,* 12**,** 17.

LIBICKI, M. C. & PFLEEGER, S. L. (2004) Collecting the Dots Problem Formulation and Solution Elements.

LOZANO TELLO, A. (2002) Métricas de idoneidad de ontologías. *Departamento de Inteligencia Artificial.* Madrid. España, Universidad Politécnica de Madrid.

LUMMUS, R. R., KRUMWIEDE, D. W. & VOKURKA, R. J. (2001) The relationship of logistics to supply chain management: developing a common industry definition. *Industrial Management & Data Systems,* 101**,** 426 - 432.

MASSACHUSSETS SCHOOL OF PROFESSIONAL PSYCHOLOGY, M. (2007) The Landscape of Decision-Making in Human Resources. IN PSYCHOLOGY, M. S. O. P. (Ed.) *OCP Resources.* Boston.

MENTZAS, G. (2000) Intelligent Process Support for Corporate Decision Making. IN ATHENS, N. T. U. O. (Ed.). Department of Electrical and Computer Engineering.

OVIEDO, U. D. (2008) Sistemas Basados en Reglas.

PARTRIDGE, C. (2002) The role of ontology in integrating semantically heterogeneous Databases. *Technical report. Padova, Italy 2002.* Padova, Italy.

PATEL-SCHNEIDER, P. & SIMÉON, J. (2002) The Yin/Yang Web: A Unified Model for XML Syntax and RDF Semantics. IN LABORATORIES, B. (Ed.). New York, Bell Labs.

REZA MONSEFI, R. G. (2006) Hiding Generalized Association Rules in RDF Databases.

ROBBINS, S. (1987) La toma de decisiones en la práctica. IN HISPANOAMERICANA, P.-H. (Ed.) *Administración: Teoría y práctica.* México, Prentice-Hall Hispanoamericana.

ROBINS, D. E. (2001) A Brief History of Decision-Making. IN (TEC), T. C. C. (Ed.).

ROMEIJN, H. E., GEUNES, J. & PARDALOS, P. M. (Eds.) (2002) *Supply Chain Management : Models, Applications, and Research Directions*, Kluwer Academic Pub.

SHU, T., CHEN, S., LAI, K. K., XIE, C. & WANG, S. (2006) A Study of Collaborative Planning, Forecasting and Replenishment Mechanism of Agile Virtual Enterprises. *Management of Innovation and Technology, 2006 IEEE International Conference on,* 2**,** 896-900.

SUBMISSION, W. C. M. (2004) SWRL: A Semantic Web Rule Language Combining OWL and RuleML.

SVEN, H. (2001) Decomposing relationship type by pivoting and schema equivalence. *Data & Knowledge Engineering,* 39**,** 75 - 99.

THOMAS EITER, G. I., AXEL POLLERES, ROMAN SCHINDLAUER, HANS TOMPITS (2006) Reasoning with Rules and Ontologies.

VENKATRAMAN, N. (2005) Businesses Intelligence: El conocimiento compartido. *Ibermática.*

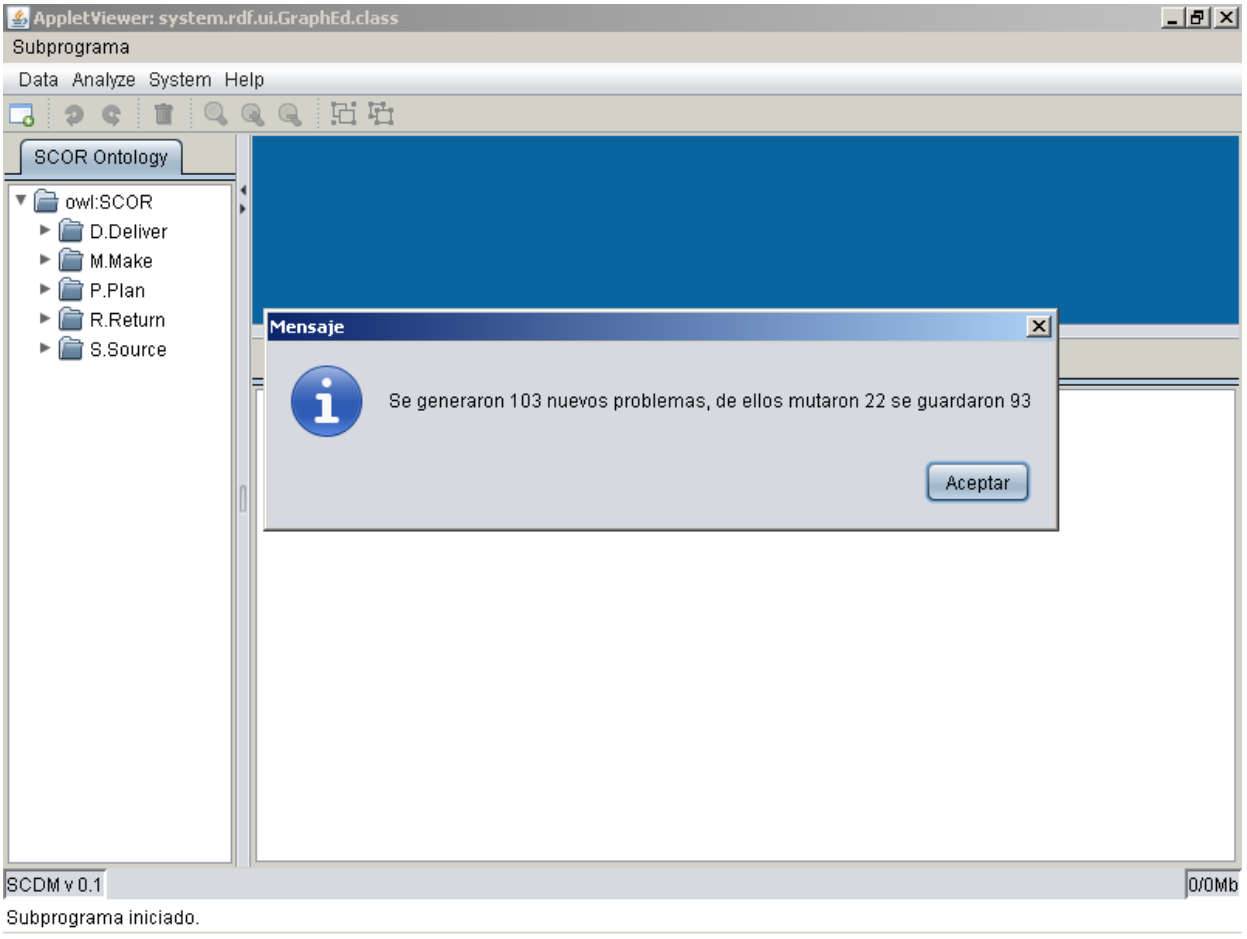
WACHE, H., VÖGELE, T., VISSER, U., STUCKENSCHMIDT, H., SCHUSTER, G., NEUMANN, H. & HÜBNER, S. (2001) Ontology-Based Information Integration: A Survey. IN BREMEN, C. F. C. T. U. O. (Ed.). Bremen, The BUSTER Project, Intelligent Systems Group.

WATSON, M. (2004) Practical Artificial Intelligence In Java.

WENYU, X. L. & LARA-ROSANO, F. (2000) Dynamic Knowledge Inference and Learning under Adaptive Fuzzy Petri Net Framework. *IEEE TRANSACTIONS ON SYSTEMS, MAN AND CYBERNETICS,* 30**,** 442-450.

YAN, B., FRANK, M., SZEKELY, P., NECHES, R. & LOPEZ, J. (2003) WebScripter: Grass-Roots Ontology Alignment via End-User Report Creation. *Springer-Verlag.* Berlin Heidelberg, Springer-Verlag.

Anexo A Imagen del sistema.



Anexo B Imagen de problema generado por el AG.