Universidad Central "Marta Abreu" de Las Villas

Facultad de Matemática, Física y Computación

"Trabajo de Diploma"

"Sistema inteligente para la toma de decisiones en la Unión Nacional Eléctrica (UNE) utilizando un enfoque basado en casos"

Autora

Amanda Riverol Quesada

Tutores

Dra. María M. García Lorenzo

Msc. Nayi Sánchez Fleitas

Curso

2014-2015

Hago constar que el presente trabajo fue realizado en la Universidad Central “Marta Abreu” de Las Villas como parte de la culminación de los estudios de la especialidad de Ciencia de la Computación, autorizando a que el mismo sea utilizado por la institución, para los fines que estime conveniente, tanto de forma parcial como total y que además no podrá ser presentado en eventos ni publicado sin la autorización de la Universidad.

_____

Firma del autor

Los abajo firmantes, certificamos que el presente trabajo ha sido realizado según acuerdos de la dirección de nuestro centro y el mismo cumple con los requisitos que debe tener un trabajo de esta envergadura referido a la temática señalada.

_____ _____

Firma del tutor

Firma del jefe del Seminario de Inteligencia Artificial

Aléjate de aquellos que intentan menospreciar tus ambiciones. Gente pequeña siempre lo hace, pero los verdaderamente magníficos te hacen sentir que, tú también, puedes ser magnífico.
Mark Twain

A mis padres
A Damny

A dios por permitirme llegar

A mis padres, por apoyarme cada día sin perder el aliento

A Maida, por su cariño obsesivo todos estos años

A Damny, por todo su amor y compañía incondicional

A Lisvandy, por ser el primero y el último de mis agradecimientos

A Ernesto, por brindarme su amistad y su ayuda

A Daniela por ser la hermana que todos necesitan

A Lázaro, Neysi, Dianela y Jennifer, mis amigos por alegrarme y estar siempre pendientes

A mis compañeros de aula, porque de una forma u otra formaron parte de estos cinco años

A mis tutoras, por su entrega y apoyo

A mis profesores Morell, Yailen, Liset, Mabel, Beatriz, Marilyn y Zenaida de quienes he aprendido mucho y a los cuales considero grandes personas

A todos, gracias.

Resumen

En los últimos años el ahorro de energía eléctrica se ha convertido en una de las principales tareas del país. Desde el punto de vista social, la energía es un factor importante en el desarrollo de las fuerzas productivas y en la elevación del nivel de vida de la población. Como parte del proceso de informatización, la empresa de Tecnología de la Información y la Automática, perteneciente a la UNE, desarrolla el Sistema de Gestión de Redes que constituye la base de información del Sistema de Información Geográfica (SIG). El SIG cuenta aproximadamente con 265 opciones de búsqueda que facilitan la información para tomar algún tipo de decisión en varias partes de la empresa. Información que satisface las necesidades principales aunque no las necesarias para su empleo óptimo, partiendo de esta necesidad la empresa plantea la necesidad de obtención de consultas en tiempo real utilizando un enfoque basado en casos. El presente trabajo diseña un sistema basado en casos tipo solucionador de problemas utilizando como base de casos las 265 consultas estáticas registradas las cuales se describen en ocho rasgos predictores del tipo de datos nominal, conjunto, ontologías y tres rasgos objetivos. La base de caso está organizada jerárquicamente en tres niveles lo cual favorece los procesos de acceso, recuperación y aprendizaje del razonador. La implementación del sistema inteligente de consultas en tiempo real para la UNE (SICUNE) cumple los requerimientos solicitados por el usuario con un 97 % de exactitud en las respuestas esperadas.

ABSTRACT

In recent years, saving electricity has become one of the main tasks of the country. From the social point of view, energy is an important factor in the development of productive forces and raising the population's standard of living. As part of the computerization process, the Enterprise of Information Technology and Automation, belonging to UNE, developed the Network Management System which is the basis of information of the Geographic Information System (GIS). GIS has approximately 265 search options that facilitate information to make some decisions in various parts of the company; information that meet the primary needs but not the necessary ones for optimum use, starting from this need the company raises the necessity to obtain real-time queries using a case-based approach. This work designed a case-based problem-solver type system using as a basis the 265 static queries registered, which are described in eight predicting features of nominal data types, set, ontologies and three objective features. The base case is organized hierarchically in three levels which favors the processes of access, retrieval and reasoner's learning. The implementation of the real-time queries' intelligent system for UNE (SICUNE) fulfills the requirements requested by the user with 97% accuracy in the expected answers.

Tabla de contenidos

INTRODUCCIÓN

En los últimos años el ahorro de energía eléctrica se ha convertido en una de las principales tareas del país, para lo que se han utilizado los más diversos adelantos de la ciencia y la técnica. Desde el punto de vista social, la energía es un factor importante en el desarrollo de las fuerzas productivas y en la elevación del nivel de vida de la población. La infraestructura eléctrica del país no ha estado al margen de los problemas presentados en los últimos tiempos, por lo que el consumo debe ser planificado racionalmente; lo cual se traduce en la disminución de interrupciones, fallas y pérdidas eléctricas(Alfredo Castro).

Como parte del proceso de informatización, la empresa de Tecnología de la Información y la Automática (ATI), perteneciente a la UNE, desarrolla el Sistema de Gestión de Redes (SIGERE) para automatizar el proceso de transmisión y distribución de la energía eléctrica, que abarca la informatización de las redes y tiene su génesis en el Sistema de Gestión Empresarial (SIGE).

SIGERE constituye la base de información del Sistema de Información Geográfica (SIG) desarrollado como tecnología para el manejo de información espacial, territorial y geográfica, logrando la producción y almacenamiento de gran cantidad de datos geográficos para estudios analíticos (Massó, 2014).

El SIG brinda la información de sus elementos a través de la conexión con la base de datos del SIGERE, esto permite al usuario obtener una mayor información de la red mediante consultas al sistema. La primera versión del SIG se presenta en la UNE en febrero del 2002 y contó aproximadamente con 265 opciones de búsqueda que facilitan la información para tomar algún tipo de decisión en varias partes de la empresa.

Esta información satisface las necesidades principales aunque no las necesarias para su empleo óptimo, dado a que las redes eléctricas son heterogéneas y muy variables según la localidad donde se encuentran.

Si se desarrolla para cada problema que surge, una consulta estática, con el tiempo la base de datos se abarrotaría de estas. Esto puede provocar que la base de datos se vuelva cada vez más grande y almacene a su vez muchas consultas poco usadas. Para darle solución a esto el sistema debe ser capaz de generar consultas inteligentes en tiempo real, donde se aproveche el conocimiento

obtenido de las realizadas anteriormente. Esto evidencia una problemática que aún no se aborda de manera completa y justifica el siguiente planteamiento de investigación:

Cómo desarrollar consultas automáticas que involucren tipos de datos conjuntos, ontologías y cadenas, obtenidos de consultas previamente hechas al SIG, que mejoren la calidad y los tiempos de respuesta.

Partiendo del problema científico se formula el **objetivo general** del trabajo:

Desarrollar un sistema inteligente para la toma de decisiones en la UNE utilizando un enfoque basado en casos.

Para dar cumplimiento al objetivo general, se desglosan los **objetivos específicos:**

1. Definir los rasgos que caracterizan al caso.
2. Analizar como determinar la similitud entre casos.
3. Construir la base de casos.
4. Diseñar e implementar el razonador basado en casos.
5. Evaluar el sistema desarrollado.

Para lo que se plantea las siguientes **preguntas de investigación:**

1. ¿Resultan igualmente importante todos los rasgos del espacio de representación del caso?
2. ¿Existe una función de semejanza que resulte factible implementar para analizar la similitud entre casos o es necesario proponer una nueva función de semejanza?
3. ¿Cuál organización favorecerá el acceso y recuperación de casos, considerando una toma de decisiones en tiempo real?
4. ¿Con qué valor de *k* trabajar para evaluar la respuesta de los *k* casos recuperados por el sistema?

La **novedad científica** principal que aporta esta investigación, radica en la creación de consultas inteligente para SQL utilizando Sistemas RBC y el manejo de ontologías en el mismo.

La implementación de este sistema como soporte al SIG para las UNE tiene un **valor práctico e impacto económico**, pues permitirá reducir gastos de combustible y de tiempo en la UNE, al brindar respuestas adecuadas a cualquier tipo de consultas, ya sea de representación de las redes o de fallas de equipamiento. Su empleo contribuirá a facilitar la toma de decisiones al realizar un análisis de riesgo o dar solución a un determinado problema en la red.

Su **relevancia social** esta evidenciada en que la población se ve altamente beneficiadas con la aplicación, pues permitirá la reducción de los tiempos de respuesta a las quejas y disminuirá el tiempo de avería en las áreas claves del país.

Este trabajo es viable pues con el desarrollo de la tecnología de la información y de la Inteligencia Artificial (IA), es viable la creación de un sistema inteligente para la toma de decisiones, para la realización del mismo será utilizada la base de datos del SIGERE (que nutre de información al SIG) gestionada mediante SQL Server 2008 R2, que tiene más de una década de creación y contiene información detallada de todos los elementos asociados a las redes eléctricas, además se cuenta con especialistas en la materia de gestión y administración de redes, pertenecientes a ATI Santi Spíritus y con los recursos necesarios para la elaboración del mismo.

Para la presentación de esta investigación, este Trabajo de Diploma se estructuró de la forma siguiente. Un Capítulo 1, que contiene el marco teórico-referencial que sustentó la investigación originaria. Un Capítulo 2, en el que se resume y explica todo el proceso de recuperación y adaptación del sistema basado en caso que soporta el SICUNE. En el Capítulo 3 se presenta el sistema SICUNE, en el que se aplican los resultados teóricos de la investigación. Este sistema permite la generación de consultas automáticas. Este documento culmina con las conclusiones, recomendaciones, referencias bibliográficas y los anexos.

Sistemas basados en casos

CAPÍTULO 1. SISTEMAS BASADOS EN CASOS.

Los Sistemas Expertos o Sistemas Basados en Conocimiento (SBC) surgen en la década del 70, dado que los métodos de solución de problemas generales eran insuficientes por ser necesario conocimiento específico sobre el problema, en lugar de conocimiento general aplicable a muchos dominios (Méndiz, 1992). Los SBC son un campo de la Inteligencia Artificial que se ocupa del estudio de la adquisición del conocimiento, su representación y la generación de inferencias sobre ese conocimiento (Malagón, 2003).

Estos constituyen un modelo computacional formado por tres componentes básicas: la base de conocimientos, la máquina de Inferencia (MI) y la interfaz con el usuario (Elkan, 1996) donde:

- La base de conocimientos almacena el conocimiento necesario para resolver los problemas del dominio de aplicación, atendiendo a una forma de representación del conocimiento (FRC).
- MI es un procedimiento basado en un esquema de razonamiento o método de solución de problemas (MSP) que utiliza el conocimiento para resolver los problemas de ese dominio.

Los SBC simulan las cadenas de razonamiento que realizaría un experto para resolver un problema de su dominio a través de una heurística. El conocimiento general es usado por la heurística para guiar la búsqueda, permitiendo encontrar rápidamente una solución a un problema sin tener que realizar un análisis detallado de una situación particular (Expósito Gallardo, 2008).

El componente principal de un SBC es su base de conocimiento, que se compone de un conjunto de representaciones de ciertos hechos reales en lenguaje natural o formal; sobre la misma se pueden hacer varias acciones como llegar a conclusiones, proceso llamado inferencia; o modificarla, llamado aprendizaje, para adaptarla a otros planteamientos.

Para la representación de este conocimiento se utilizan varios esquemas clásicos como: lógica, reglas, redes asociativas, redes semánticas, probabilidades, pesos, o los propios ejemplos de un dominio de aplicación. Estos dan lugar a distintos tipos de sistemas basados en el conocimiento como se puede observar en la Tabla 1. Entre los sistemas se encuentran los Sistemas Basados en Reglas (SBR)(Moya-Rodríguez et al., 2012), los Sistemas Basados en Probabilidades (SBP)(Páez et al., 2011), los Sistemas Expertos Conexionistas o Redes Expertas (RNA) (H and Q, 2001)y los

Sistemas Basados en Casos (SBCa) o Sistemas de Razonamiento Basado en Casos (RBC)(Richter and Weber, 2013), entre otros.

Aunque un comportamiento inteligente no se obtiene con la representación por sí sola, es necesario organizarlo de una forma efectiva, de manera que permita que se recupere el conocimiento adecuado en el momento preciso. Por lo que se puede decir que el éxito del SBC depende tanto de los procesos que manejan el conocimiento como el conocimiento en sí mismo (Díaz, 2002).

Tabla 1: Tipos de sistemas basados en el conocimiento.

Tipo	FRC	MSP	Fuente de Conocimiento
SBR	Reglas de producción	Usualmente backward o forward	Expertos
SBP	Probabilidades o frecuencia	Teorema de Bayes y otras técnicas de inferencia estadísticas	Ejemplos
RNA	Pesos y alguna otra FRC	Calculo de los niveles de activación de las neuronas	Ejemplos
RBC	Casos	Razonamiento basado en caso (búsqueda de semejanza y adaptación de la solución)	Ejemplos

Como se puede observar en la Tabla 2, muchas veces en la práctica el proceso de obtención del conocimiento es muy complejo y lleno de incertidumbre, o puede causar un cuello de botella al dificultarse su obtención, lo que frena el desarrollo de los SBC.

Los sistemas RBC surgen como un paliativo al proceso de la ingeniería de conocimiento y se apoya en la premisa de que problemas parecidos tendrán soluciones semejantes. Tomando este principio como base, la idea consiste en, dado un problema a resolver, recuperar de una memoria de ejemplos resueltos, las experiencias pasadas más parecidas, para que las soluciones recuperadas anteriormente sirvan de guía para encontrar las nuevas soluciones (Rosa, 2009).

En los últimos años, los sistemas RBC han experimentado un rápido crecimiento desde su nacimiento en Estados Unidos. Lo que sólo parecía interesante para un área de investigación muy reducida, se ha convertido en una materia de amplio interés, multidisciplinario y con gran auge comercial (Laura Lozano, 2008).

Estos sistemas necesitan una colección de experiencias, llamadas casos, almacenadas en una base de casos (BC), donde cada caso se compone generalmente de una descripción del problema y la solución que se aplicó (Díaz, 2002).

Tabla 2: Comparación entre los sistemas basados en el conocimiento respecto a la obtención de su conocimiento.

Tipo	Forma de obtención de conocimiento	Complejidad
SBR	Extracción del conocimiento desde varias fuentes (entrevistas a expertos, consulta de ICT y análisis de casos). Formulación de reglas. Codificación en dependencia del lenguaje seleccionado. Refinamiento de las reglas.	Este proceso puede ser muy complejo y lleno de incertidumbre.
SBP	Coleccionar muestras y realizar un procesamiento estadístico que produzca las probabilidades o frecuencias.	Este proceso puede ser también arduo por la posible carencia de fuentes que permitan estimar tales probabilidades con suficiente fiabilidad
RNA	Selección de ejemplos. Diseño de la topología de la red. Entrenamiento de la red para hallar el conjunto de pesos.	El entrenamiento puede requerir sólo pocas semanas y por ello puede ser menos complejo el proceso sin embargo, la definición de la topología de la red puede ser compleja.
RBC	La selección de un conjunto de ejemplos o casos resueltos y su organización en la base de casos	Reduce la envergadura del proceso sustancialmente.

Watson (Watson, 1997) propone clasificar estos sistemas por el tipo de tarea que desempeñan en:

1. Clasificación: La solución almacenada en el caso es la clase a la que pertenece la situación descrita en el problema. Suele ser útil para problemas con información incompleta. La clase del caso recuperado se utilizada como solución. En esta categoría entran los sistemas de diagnosis, predicción, control de procesos y evaluación.
2. Síntesis: La solución para el nuevo problema debe ser construida en función de los casos recuperados, por lo general después de un proceso de adaptación. Aquí entran los sistemas de diseño, planificación y configuración.

1.1 Arquitectura de los sistemas basados en conocimiento.

Como se observa en la ***Figura 1*** los SBC cuentan con tres componentes principales: una interfaz de usuario, un MI y una base de conocimiento (Cordero Morales et al., 2013) donde:

- **Base de conocimiento** almacena el conocimiento.

- **MI** es la máquina de razonamiento del sistema, la cual compara el problema insertado con los que están almacenados en la base de conocimiento y como resultado infiere una respuesta con el mayor grado de semejanza a la que se busca.
- La **interfaz de usuario** permite la comunicación entre el sistema y el usuario, dando la posibilidad de interactuar con la base de conocimiento, plantear nuevos problemas y consultar los resultados inferidos.

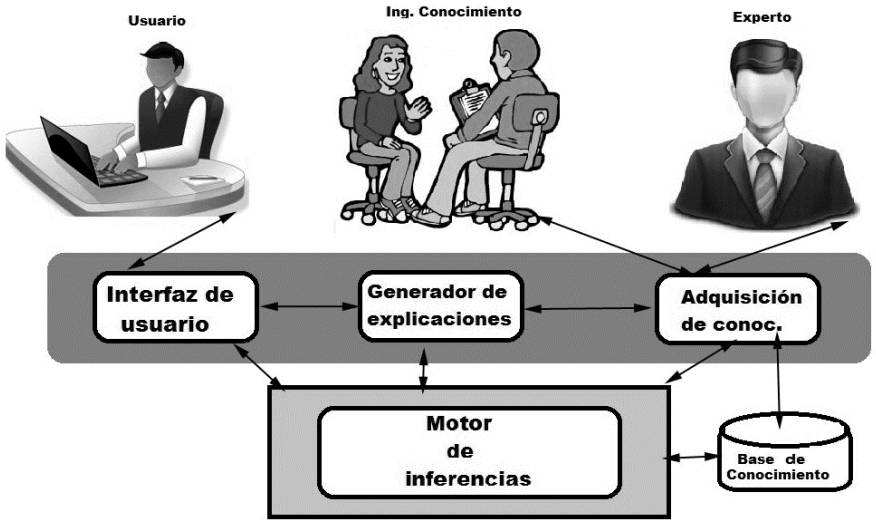


Figura 1: Arquitectura de los sistemas basados en conocimiento.

Los sistemas RBC basan su arquitectura en la de los SBC. De ahí que un sistema que implemente RBC tenga como motor de inferencia un razonador basado en casos del tipo interpretativo o solucionado de problemas y una base de casos. A continuación se le dedicará un sub-epígrafe a cada una de las partes que lo conforman. Se hará énfasis en las características de cada una y la utilidad dentro del sistema.

1.2 La base de casos.

Un caso contiene información útil en un contexto concreto, el problema es identificar los atributos que caracterizan al contexto y detectar cuándo dos contextos son similares. Un caso es una experiencia que enseña algo, pero en un dominio puede haber experiencias que no aporten nueva información al sistema, lo cual plantea el problema de identificar cuando dos casos son superficialmente distintos. Además, lo que el caso enseña es relativo a determinados objetivos y por lo tanto, se ha de tener en cuenta que una determinada solución puede fallar, en los RBC interesa no solo guardar las soluciones que funcionan sino también aquellas que fallaron, ya que

ambas contienen información útil que permitirá repetir las soluciones exitosas, y evitar la repetición de las fallidas. Los casos pueden ser de diferentes tamaños, y se van almacenando en una memoria denominada Base de Casos (BC) (Laguía, 2003).

La Rosa (Rosa, 2009) plantea que un caso se puede ver como una tupla de la forma

Caso = {*Problema*, *Solución*, *Efectos*}, donde:

Problema: Es la descripción de un problema, ya sea la situación a interpretar, el problema de planificación a resolver o el artefacto a diseñar. Esta es la parte del caso que se utiliza para determinar su similitud con el problema a resolver.

Solución: Es la solución al problema o los pasos necesarios para obtener dicha solución. Además del artefacto construido, el plan aplicado o la interpretación asignada. En la solución de un caso se puede guardar información adicional, sobre todo con el objetivo de facilitar futuras adaptaciones. Información acerca del proceso que llevó a la obtención de la solución, qué alternativas se consideraron, cuáles se eligieron y cuáles se descartaron, y el motivo de la elección o el rechazo de dicho caso.

La Solución será reusada cuando el problema sea una situación similar, por lo que se debe tener especial cuidado en la información que se almacena (Sankar, 2004).

Efectos: Describe la situación después de aplicar la solución al problema.

Este esquema es válido en:

- Situaciones sencillas como las de clasificación, donde el problema es un conjunto de variables atributo-valor y la solución la clase a la que pertenece el ejemplo.
- Situaciones complejas, un caso puede representar un estado inicial en un modelo. La solución es una serie de pasos aplicados basados en la experiencia pasada. Los efectos almacenan cual fue el resultado de aplicar la solución encontrada.

El modelo sugerido por La Rosa es una estructura general, por lo que se pueden encontrar casos compuestos solo por la descripción del problema y la solución aplicada. También se pueden encontrar casos en los que no se hace una distinción entre sus partes, sino que cada caso está descrito por un conjunto de atributos los cuales pueden ser representados por dominios diferentes y se pueden diferenciar también por su importancia en la inferencia. Una consulta estaría

compuesta solo por una parte del conjunto que conforma el problema. La solución estaría dada por aquellos atributos que no estén en la consulta (Giménez, 2006).

En (Azán et al., 2014) distinguen dos grandes tipos de representaciones para los casos:

Representaciones planas: Define una serie de atributos con un conjunto de posibles valores de tipos simples, cadenas de caracteres, números o símbolos, donde no se define relación alguna entre los atributos ni entre sus valores.

Representaciones estructuradas: Define la lista de atributos y valores asociados, pero a diferencia de las representaciones planas los valores pueden ser objetos que a su vez tienen atributos. En las listas se definen relaciones entre los atributos y/o entre los valores. Para esto se utiliza cálculos de predicados, redes semánticas, lenguajes basados en marcos, lenguajes orientados a objetos y lógicas descriptivas, entre otros.

Guardar las experiencias en forma de casos tiene como ventaja sobre otros sistemas (Giménez, 2006):

- A los expertos les resulta más fácil proporcionar ejemplos, que no crear reglas de comportamiento.
- Se puede obtener un nuevo caso cada vez que se resuelve un nuevo problema, se puede indicar su resultado, explicarlo y clasificarlo como fracaso o éxito, pues un caso clasificado como fracaso es también información útil.
- Es posible comparar los casos y adaptarlos de manera efectiva.
- Los casos mantienen su vigencia durante bastante tiempo, es decir, los problemas tienden a repetirse.

Independientemente de la representación de los casos estos se almacenan en una BC, donde la organización de la memoria y su acceso sería la parte central del razonamiento basado en casos. La respuesta de un caso problema se puede deducir de otros casos de la BC, pero si se necesita un gran costo computacional para llegar a esta respuesta, entonces puede interesar almacenarlo en la BC para disminuir la complejidad en futuras problemáticas, así como, si la respuesta del caso es fácilmente deducible a partir de otros casos de la BC y no aporta nada nuevo, entonces no debe guardarse, disminuyendo en complejidad espacial, intentando mantener un equilibrio entre complejidad temporal y espacial.

Como se ha dicho anteriormente, la base de casos es el principal activo de los Sistemas RBC y estaría formada por todas las experiencias representadas en forma de casos, ya sean casos en que la solución fue la correcta o casos incorrectos. Los entornos en los que se puede crear una BC son muy variados y al mismo tiempo pueden ser muy complejos.

En la Figura 2, se puede ver el esquema de representación de una BC compuesta de *n* casos. Cada caso se compone de una descripción y una solución. La descripción está compuesta por una serie de atributos denominados rasgos. Se puede observar que en esta BC, el último atributo del caso corresponde con la solución de los atributos planteados en la descripción del caso.

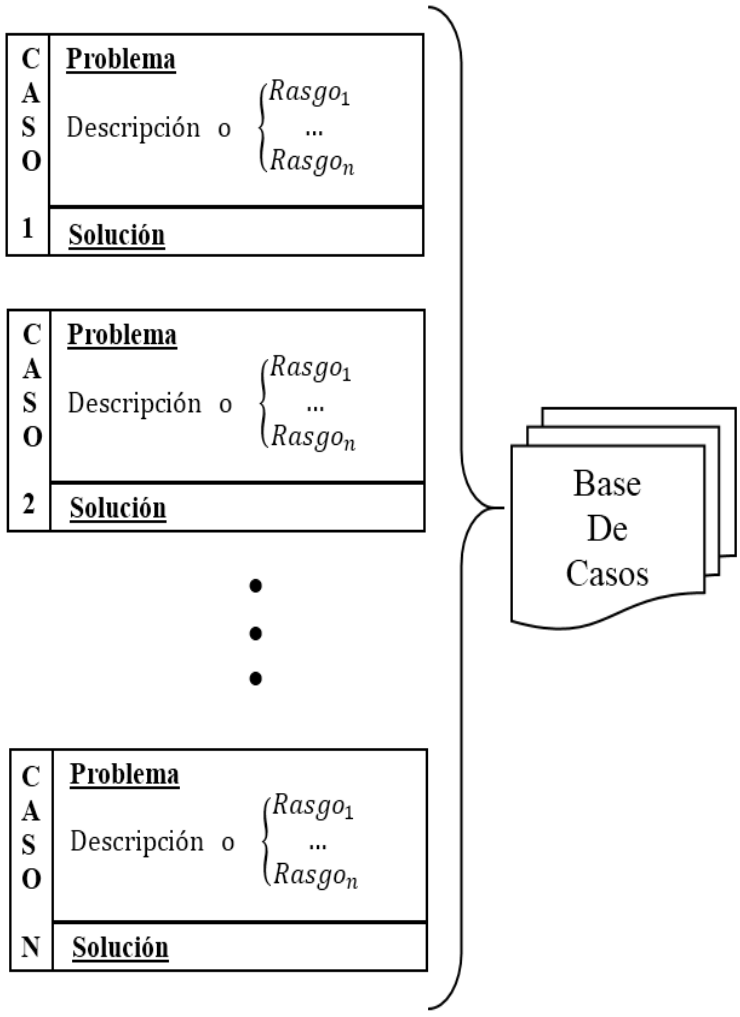


Figura 2: Esquema de representación de una base de casos.

1.2.1 **Organización de la base de casos.**

Una vez que se ha decidido que información se incluye en cada caso y como representarla, se organiza el conjunto de casos de forma que se facilite el acceso. Esto se realiza a través de la *indexación* de la BC, proceso que requiere: asignar los índices de los casos y organizar dichos índices.

Los índices deben ser predictivos, esto es, que permitan identificar las situaciones en las que los casos pueden aportar información útil. En (Kolodner and Leake, 1996) se sugiere que para determinar los índices de los casos hay que considerar la tarea que se quiere resolver y elegir como índices los conjuntos de características que describen que un caso será útil para resolver dicha tarea. Esta idea trata de que los índices distingan los casos entre sí con respecto a algún objetivo, para obtener proporciones conceptuales útiles de la BC.

Los índices se pueden obtener usando la técnica de adquisición de conocimiento que consiste en entrevistar a un experto que identifica cuales son las características críticas. Otra opción consiste en utilizar técnicas inductivas que ayudan a identificar cuáles son las características más discriminantes. En (Lopez and Plazas, 1997) los autores se refieren a la organización de la BC como un problema abierto, en el que una buena indexación no es suficiente.

Un esquema de organización muy usado consiste en construir un árbol de decisión, de forma que sean los valores de los índices los que permitan situar el caso en una parte u otra de la estructura. Los arboles de decisión inducidos mediante el algoritmo ID3 o alguna de sus variantes (kolodner, 1993). El problema básico de los árboles de decisión es que solo comparan la igualdad de características y no permiten manejar adecuadamente las consultas incompletas.

De mayor difusión son los arboles k-d, una estructura de datos que resulta de generalizar los árboles de decisión. La estructura está pensada para dado un de puntos repartidos en un espacio métrico *k*-dimensional, recuperar eficientemente los *k* vecinos más próximos a un punto dado. El árbol *k-d* se induce, mediante un algoritmo sofisticado, a partir del conjunto de puntos -casos- inicial, de forma que en cada hoja del árbol haya como máximo un cierto número *n* de puntos. El problema geométrico consiste en dividir un espacio k-dimensional en volúmenes homogéneamente (Belén, 2002).

Las redes de activación son una adaptación de las redes neuronales donde la BC se representa como una red de nodos interconectados. Esta aproximación se ha utilizado en CREEK (Aamodt, 1991). A su vez en el sistema PROTOS (Bareiss, 1988) se utiliza una organización jerárquica alternativa para los casos. Cada caso está asociado con una categoría y los índices pueden apuntar tanto a un caso como a una categoría. El sistema PARIS (Bateman, 1990) propone representar los casos en distintos niveles de abstracción. Los casos abstractos ubicados en los distintos niveles de abstracción pueden utilizarse como índices jerárquicos para aquellos casos que contienen el mismo tipo de información pero aun nivel de abstracción menor. Se puede construir una jerarquía. Los nodos hojas contienen los casos concretos.

1.3 Motor de Inferencia

El RBC es un ciclo de las 4R, ver Figura 3, que incluye las siguientes etapas:

1. Entrada de un nuevo problema a resolver, éste es enviado al Modulo Recuperador, el cual realiza una búsqueda en la Base de Casos y recupera (***Retrieve***) problemas o casos similares.
2. Estos son enviados al Modulo Adaptador, con el fin de dar la solución de la forma más óptima al problema reutilizando (***Reuse***) las soluciones propuesta en los casos recuperados.
3. Una vez hallada la solución, de ser necesario se revisa (***Revise***) la solución propuesta y se almacena (***Retain***) junto con la descripción del problema en la Base de Casos, constituyendo un nuevo caso.

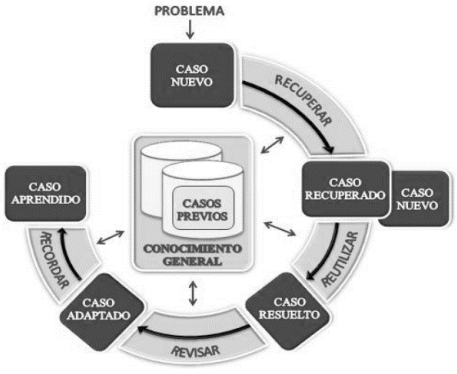


Figura 3: Ciclo Razonador Basado en Casos[1].

[1] DASIEL CORDERO, Y. R., YOANNY TORRES 2013. Sistema de Razonamiento Basado en Casos para la identificación de riesgos de software. Ingeniería y Gestión de software – Inteligencia artificial, 7, 222-239.

En la práctica cada uno de estos pasos puede ser implementado de diversas formas; por ejemplo, en la recuperación de casos se usan algoritmos de mayor semejanza, árboles de decisión o memorias asociativas conexionistas; la representación de los casos puede realizarse en forma de documentos textos, registros de bases de datos, redes semánticas u otros modelos, así como para realizar la selección de los casos semejantes se usa una función de semejanza.

1.3.1 **Recuperación.**

El proceso de recuperación se encarga de extraer de la base de casos el caso o los casos más parecidos a la situación actual. Centra su objetivo en encontrar una medida de similitud efectiva que permita una buena recuperación de casos. Dependiendo de la representación de los casos y del número de casos recuperados, puede tener las siguientes fases (Rosa, 2009):

Identificación de características: Por lo general los casos no están representados como los problemas, por lo que hay que identificar en el problema nuevo los atributos que se van a utilizar para la recuperación.

Comparación: Compara el nuevo problema con los anteriores. Esta comparación se hace equiparando las características que forman los índices de los casos. Es su ausencia la comparación se hace con todos los casos en la base de casos o un subconjunto de ellos dependiendo de la organización utilizada.

Ranking: Se le asigna a cada caso un valor en función de la medida de similitud, para determinar cuál de todos los casos se parece más al problema actual y que grado de semejanza tienen el resto de casos, por si se utilizan varias alternativas.

Selección: Se escogen los *n* mejores casos del ranking según se necesiten en el proceso de reutilización.

Esta etapa es muy importante dentro del ciclo RBC, pues si el caso recuperado no es el adecuado, no podrá resolverse el problema correctamente y el sistema conducirá a cometer un error. Por lo que se desarrollan algoritmos y técnicas con el objetivo de recuperar los casos más adecuados de la BC. Para esto se debe partir de identificar cual sería el caso más similar y luego establecer la similitud entre casos, pues la mayoría de los algoritmos de recuperación y selección están basados en estos aspectos fundamentalmente.

1.3.2 **Cálculo de la similitud entre casos.**

Las funciones de similitud dan como resultado un valor entre 0 y 1 que expresa cuan semejante es el problema a cada uno de los casos contenidos en la Base de Casos. Para su trabajo la función de similitud utiliza una función de comparación por rasgos, esta función compara el valor de un rasgo del problema con el valor de ese rasgo en un caso y obtiene un valor (generalmente entre 0 y 1) que expresa el grado de similaridad entre ambos valores.

Dependiendo del tipo de información que se represente, existen distintas definiciones de similitud, disimilitud o distancias (Bonillo, 2003). Considerando que un caso viene descrito por un conjunto de atributos, la aproximación más básica para el cálculo de la similitud consiste en contabilizar los valores iguales en los atributos comunes de los casos a comparar.

Esta comparación se basa en una similitud local que determina la similitud entre valores de un mismo atributo y una similitud global, que combina los resultados de ciertas similitudes locales a todos los atributos de los casos a comparar. La Tabla 3 muestra algunas de las similitudes globales más utilizadas (Althoff et al., 1995), estas no son sensibles a la naturaleza de los atributos por el contrario, las similitudes locales varían según el tipo de datos de los atributos.

Tabla 3: Similitudes Global.

$\frac{1}{p}\sum_{i=1}^{p} sim_i(a_i, b_i)$	**Bloques**	$\frac{1}{p}\left[\sum_{i=1}^{p} [sim_i(a_i, b_i)]^r\right]^{\frac{1}{r}}$	**Minkowski**
$\sum_{i=1}^{p} w_i * sim_i(a_i, b_i)$	Bloques Ponderados	$\left[\sum_{i=1}^{p} w_i [sim_i(a_i, b_i)]^r\right]^{\frac{1}{r}}$	Minkowski Ponderados
$\frac{1}{p}\left[\sum_{i=1}^{p} [sim_i(a_i, b_i)]^2\right]^{\frac{1}{2}}$	Euclidiana	$\max_{i=1}^{p} [w_i * sim_i(a_i, b_i)]$	Máximo
$\frac{\sum_{i=1}^{p} sim_i(a_i, b_i)}{p}$	Media Aritmética	$\min_{i=1}^{p} [w_i * sim_i(a_i, b_i)]$	Mínimo
Donde: $\sum_{i=1}^{p} w_i = 1$			

El valor asignado a un atributo de evaluación para indicar su importancia relativa con respecto a los otros atributos se le denomina peso. El peso de un atributo puede ponerse por criterio de expertos incrementando o decreciendo el rango de los atributos o de forma calculada. Mientras

mayor sea el peso de un atributo, mayor es su importancia. En el caso de *n* atributos, un conjunto de pesos está definido por:

$$w = (w_1, w_2, \ldots, w_i, \ldots, w_n), w_i \geq 0, y\ \Sigma\ w_j = 1.$$

Como se puede observar en el Anexo 1 existen diferentes procedimientos para ponderar atributos, como son: proporción, comparaciones pareadas, proceso analítico jerárquico y SMART. Estos métodos varían en su grado de dificultad, supuestos teóricos y rigor.

La similitud local, calcula la similitud entre los valores del mismo atributo en dos casos distintos, las mismas se determinan según el tipo de atributo a comparar, ya sean numéricos, simbólicos, o algunos más complejos.

Los rasgos simbólicos o numéricos pueden ser univaluados o multivaluados, en Tabla 4 se muestran ejemplos de criterios de comparación local los cuales puedes ser booleanos o reales. (Althoff et al., 1995).

Tabla 4: Funciones de distancia local para datos numéricos y nominales.

	univaluados	**multivaluados**	
simbólicos	$\mathrm{sim(a,b)} = \begin{cases} 1, & \mathrm{a} != \mathrm{b} \\ 0, & \mathrm{a} = \mathrm{b} \end{cases}$	$\frac{\mathrm{card(a \cap b)}}{\mathrm{card(a \cup b)}}$	$\frac{\mathrm{card(a \cap b)}}{\mathrm{card(O)}}$
		$\frac{\mathrm{card(a \cap b)}}{\min\{\mathrm{card(a)}, \mathrm{card(b)}\}}$	$\frac{\mathrm{card(a \cap b)}}{\max\{\mathrm{card(a)}, \mathrm{card(b)}\}}$
numéricos	$\mathrm{sim(a,b)} = \frac{\lvert \mathrm{a} - \mathrm{b} \rvert}{\mathrm{long\ I}}$	$\frac{\mathrm{long(a \cap b)}}{\mathrm{long(a \cup b)}}$	$1 - \frac{\lvert \mathrm{a_c} - \mathrm{b_c} \rvert}{\mathrm{long(O)}}$
		$\frac{\mathrm{long(a \cap b)}}{\min\{\mathrm{long(a)}, \mathrm{long(b)}\}}$	$\frac{\mathrm{long(a \cap b)}}{\max\{\mathrm{long(a)}, \mathrm{long(b)}\}}$

Siendo: O el conjunto de valores posibles, $long(I)$ la longitud del intervalo I, y a_c el punto central del intervalo a.

Existen disímiles problemáticas donde aparecen rasgos que toman diferentes valores de forma simultánea, los cuales pudieran representarse de forma natural mediante conjuntos. Según (Devlin, 1993): un conjunto es cualquier colección C de objetos determinados y bien distinto x de la percepción o del pensamiento (elementos de C), reunidos en un todo.

De acuerdo con esta definición un conjunto queda definido si es posible describir completamente sus elementos. El procedimiento más sencillo de descripción es nombrar cada uno de sus elementos

entre llaves. Es decir un conjunto A formado por la colección de elementos a, b y c se define como $A = \{a, b, c\}$ (Devlin, 1993).

En la Tabla 5 se muestra una serie medidas de distancia, adaptadas a datos de tipo conjunto. Todas estas distancias están basadas en las operaciones básicas entre conjuntos. Se han realizado trabajos para extender algoritmos de aprendizaje automático sobre este tipo de datos (González, 2010), debido a que favorecen una modelación del problema más natural.

Las distancias propuestas se basan en operaciones clásicas entre conjuntos como la intersección ($Y \cap X$), la unión ($Y \cup X$), la diferencia simétrica ($Y \Delta X$) y el cardinal ($|X|$). Las distancias anteriores tienen como argumento dos conjuntos y toman valores de salida en el intervalo [0,1]. En caso de que los dos conjuntos sean vacíos, las funciones de distancia se indefinen, por lo que se toma valor cero ya que son iguales. La complejidad temporal de estas funciones de distancia pertenecen a un orden de *O*(*n*), donde *n* es igual al número de elementos del dominio del atributo.

Tabla 5. Funciones de distancia local para datos tipo conjunto.

Jaccard:	**Czkanowsky-Dice**
$\boldsymbol{\gamma_{set}(X,Y) = \frac{\lvert X \cap Y \rvert}{\lvert X \cup Y \rvert}}$	$\gamma_{set}(X,Y) = 1 - \frac{\lvert X \Delta Y \rvert}{\lvert X \rvert + \lvert Y \rvert}$
Sokal-Sneath:	**Cosine**:
$\boldsymbol{\gamma_{set}(X,Y) = \frac{\lvert X \cap Y \rvert}{\lvert X \cup Y \rvert + \lvert X \Delta Y \rvert}}$	$\gamma_{set}(X,Y) = \frac{\lvert X \cap Y \rvert}{\sqrt{\lvert X \rvert * \lvert Y \rvert}}$
Braun-Blanquet:	**Simpson**:
$\boldsymbol{\gamma_{set}(X,Y) = \frac{\lvert X \cap Y \rvert}{máx\{\lvert X \rvert * \lvert Y \rvert\}}}$	$\gamma_{set}(X,Y) = \frac{\lvert X \cap Y \rvert}{mín\{\lvert X \rvert * \lvert Y \rvert\}}$
Kulczynski:	
$\boldsymbol{\gamma_{set}(X,Y) = 1 - \frac{\lvert X \cap Y \rvert * (\lvert X \rvert + \lvert Y \rvert)}{2 * \lvert X \rvert * \lvert Y \rvert}}$	

Un estudio comparativos sobre estas distancias utilizadas en la comparación de conjuntos arrojaron que la de mejor desempeño de las siete es *Jaccard* (González, 2010).

En varias problemáticas un rasgo tipo cadena se contempla de formas diferentes, se puede considerar cada cadena como una secuencia ininterrumpida de caracteres o como un conjunto de sub cadenas delimitadas por caracteres especiales como espacios en blanco, comas y puntos; en esta última se considera una cadena como un conjunto de tokens (Amón, 2010).

El valor de un atributo puede aparecer representado de formas diferentes. Por ejemplo, el nombre "Jorge Eduardo Rodríguez López" también puede aparecer como "Rodríguez López Jorge Eduardo", "Jorge E Rodríguez López" o "Jorge Eduardo Rodríguez L."; con errores de escritura, como "Jorje Eduardo Rodrigues Lópes"; con información adicional, como "PhD Jorge Eduardo Rodríguez López", entre otras.

La distancia entre dos cadenas está definida para $d \in [0,1]$, donde 0 indica que ambas cadenas son idénticas y 1 que no tienen ni un solo carácter en común.

Actualmente existen diversas propuestas, clasificadas en dos categorías: basadas en caracteres y basadas en palabras completas o tokens. Las primeras consideran cada cadena como una secuencia ininterrumpida de caracteres; las segundas como un conjunto de subcadenas delimitadas por caracteres especiales como espacios en blanco, comas y puntos; y calculan la distancia entre cada pareja de tokens mediante alguna función basada en caracteres (Baldizzoni, 2013).

Funciones de distancia basadas en caracteres

Distancia de brecha afín: La distancia de brecha afín ofrece una solución al penalizar la inserción/eliminación de *k* caracteres consecutivos (brecha) con bajo costo, mediante una función afín $p(k) = g + h * (k - 1)$, donde g es el costo de iniciar una brecha, k el costo de extenderla un carácter, y $h + g$ (Gotoh, 1982). Se suele utilizar cuando hay abreviaciones en las cadenas o cuando hay un gran volumen de datos y además existen prefijos/sufijos sin valor semántico.

Distancia de Levenshtein: La distancia entre dos cadenas de texto *A* y *B* se basa en el conjunto mínimo de operaciones de edición necesarias para transformar *A* en *B* (o viceversa). Las operaciones de edición permitidas son eliminación, inserción y substitución de un carácter (Ramírez and López, 2006). Esto se puede utilizar cuando existen errores de ortografía dado que la distancia de edición y otras funciones de distancia tienden a fallar identificando cadenas equivalentes que han sido demasiado truncadas.

$$\textbf{Levenshtein}(\sigma_1, \sigma_2) = \text{Matriz}\,(|\sigma_1|, |\sigma_2|) \qquad \text{Matriz (i,j)=Min} \begin{cases} \text{Matriz (i-1,j) + 1 (eliminación)} \\ \text{Matriz(i,j-1) + 1 (inserción)} \\ \text{Matriz(i-1,j-1) + Cij(substitución)} \end{cases}$$

Distancia Smith-Waterman: La distancia Smith-Waterman entre dos cadenas *A* y *B* es la máxima distancia entre una pareja (*A′*,*B′*), sobre todas las posibles, tal que *A′* es subcadena de *A* y *B′* es subcadena de *B*. Tal problema se conoce como alineamiento local. (Smith and Waterman, 1981). Esto lo hace adecuado para identificar cadenas equivalentes con prefijos/sufijos que, al no tener valor semántico, cuando existe una o más tokens que no se encuentran en alguna de las dos cadenas o cuando existen espacios en blanco inútiles.

$$E(i,j) = max\begin{cases} E(i,j-1) - G_{ext} \\ H(i,j-1) - G_{init} \end{cases}$$

$$F(i,j) = max\begin{cases} F(i-1,j) - G_{ext} \\ H(i-1,j) - G_{init} \end{cases}$$

$$H(i,j) = \begin{cases} 0 \\ E(i,j) \\ F(i,j) \\ H(i-1,j-1) - W(q_i, d_j) \end{cases}$$

Distancia de Jaro:

Jaro (Jaro, 1976) desarrolló una función de distancia que define la trasposición de dos caracteres como la única operación de edición permitida. Los caracteres no necesitan ser adyacentes, sino que puede estar alejados cierta distancia que depende de la longitud de ambas cadenas.

$$\mathbf{Jaro}(\boldsymbol{\sigma_1}, \boldsymbol{\sigma_2}) = \frac{1}{3}\left(\frac{c}{|\sigma_1|} + \frac{c}{|\sigma_2|} + \frac{c - t/2}{c}\right)$$

Winkler (Winkler, 2000) propone una variante que asigna puntajes de distancia mayores a cadenas que comparten algún prefijo, un modelo basado en distribuciones Gaussianas. La distancia de Jaro puede ser calculada en $O(n)$ (Amón, 2010).

Distancia de *q-grams*

Un *q-gram*, también llamado *n-gram*, es una subcadena de longitud *q (W Yancey, 2006).* El principio tras esta función de distancia es que, cuando dos cadenas son muy distancia, tienen muchos *q-grams* en común. Se utiliza cuando hay múltiples problemas de distancia o cuando los tokens están desordenados.

Funciones de distancia basadas en tokens

Distancia de Monge-Elkan**:** Dadas dos cadenas *A* y *B*, sean $\alpha_1 \ldots . \alpha_n$ y $\beta_1 \ldots . \beta_n$ sus tokens respectivamente. Para cada token α_1 existe algún β_1 de mínima distancia. Entonces la distancia de Monge-Elkan entre *A* y *B* es la distancia máxima promedio entre una pareja $\alpha_1\ y\ \beta_1$ (Elkan, 1996)

Gelbukh (Gelbukh et al., 2009) presenta un modelo basado en la media aritmética generalizada,

$$\textbf{Monge} - \textbf{Elkan}(\sigma_1, \sigma_2) = \frac{1}{K} * \sum_{i=1}^{K} \min\{dis(\alpha_i \beta_j)\} \; donde \; j = 1 \ldots |\sigma_2|$$

en lugar del promedio, el cual supera al modelo original sobre varios conjuntos de datos. El algoritmo para la distancia de Monge-Elkan tiene un orden computacional $\boldsymbol{O(nm)}$ (Gelbukh et. al., 2009).

distancia coseno TF-IDF: Dadas dos cadenas *A* y *B*, sean $\alpha_1 \ldots . \alpha_n$ y $\beta_1 \ldots . \beta_n$ sus tokens respectivamente, que pueden verse como dos vectores $\boldsymbol{V_A}$ y $\boldsymbol{V_B}$ de *K* y *L* componentes. Cohen (Cohen, 1997) propone una función que mide la distancia entre *A* y *B* como el coseno del ángulo que forman sus respectivos vectores. La distancia coseno TF-IDF no es eficaz bajo la presencia de variaciones a nivel de caracteres, como errores ortográficos o variaciones en el orden de los tokens.

1.3.3 **Reutilizar y Adaptar**

Se consideran los casos recuperados para elaborar una solución inicial. Si se intenta resolver un problema, normalmente se toma como punto de partida la solución del problema recuperado, parte de ella o una combinación de las soluciones recuperadas.

A esta solución inicial se le suelen hacer adaptaciones sencillas, que se puede denominar adaptaciones de sentido común, antes de pasar a la fase de adaptación propiamente dicha.

Si se trata de un problema de interpretación se suele agrupar los casos recuperados de acuerdo con la interpretación que hacen. A partir de los grupos creados, se asigna una interpretación inicial de la situación. Puede ocurrir que exista una interpretación inicial y no sea necesario este paso.

Para adaptar el caso recuperado es combinado junto con el nuevo caso a través del reúso de información, y mediante mecanismos de similitud que definen la cercanía o no del caso recuperado con el nuevo, se propone una solución al caso. Esta solución puede no ser la correcta para el nuevo problema siendo necesario adaptar de algún modo esa solución.

Hay dos pasos importantes involucrados en la adaptación (Aamodt and Plaza, 1994):

1. Descifrar las necesidades que van a ser adaptadas del caso.
2. Hacer la adaptación.

1.3.3.1 Identificar qué hay que adaptar

Cuando se comienza a adaptar surgen preguntas como: ¿Qué se sustituye?, ¿Cómo afecta a la solución un cambio en la descripción?

Según (Díaz, 2002) para identificar qué debe ser corregido se pueden usar varios métodos:

- Diferencias entre las especificaciones del problema recuperado y el nuevo.
- Usar una lista de comprobaciones que se deben realizar.
- Detectar inconsistencias entre la solución recuperada y los objetivos y restricciones del problema nuevo.
- Prever los efectos que va a tener la solución, por ejemplo utilizando modelos, casos o simulación.
- Llevar a cabo la solución y analizar el resultado.

1.3.3.2 Tipos de adaptación.

Existen varios tipos de adaptación (Laguía, 2003):

Sin adaptación: Un gran número de sistemas RBC no realizan adaptación. No es necesario que el sistema la realice para que resulte útil "*repartamos el trabajo entre máquinas y humanos de forma que cada uno haga lo que mejor sabe hacer*"

Métodos basados en sustitución: Se sustituye algunos valores o elementos de la solución antigua por otros adecuados al problema nuevo, Algunos de estos métodos son:

- Re-instanciación
- Ajuste de parámetros
- Búsqueda local
- Búsqueda en la memoria
- Búsqueda especializada
- Sustitución basada en casos

Métodos basados en transformación: se realizan modificaciones en la solución antigua como son:

- Transformaciones de sentido común
- Reparación guiada por un modelo

Adaptación y reparación de propósito especial: son métodos específicos del dominio que realizan sustituciones, transformaciones estructurales o reparaciones que intentar corregir los fallos que se producen al ejecutar la solución

- Repetición derivacional: repetir los pasos que se han usado al derivar la solución antigua para obtener la solución al problema nuevo. Es lo que se hace al resolver un problema de estadística siguiendo los mismos pasos que se utilizaron antes en otro problema igual pero con valores distintos.

1.3.4 **Revisar**

Después de adaptar una posible solución, esta puede ser correcta o no, si es correcta la nueva solución será almacenada en el sistema, pero en caso contrario se pasa a la tercera fase del ciclo que consiste en revisar esa solución incorrecta.

La revisión comprende básicamente dos fases:

- Evaluar la solución. Donde se decide si la solución dada es la correcta al problema planteado.
- Reparar los fallos. Si no es correcta la solución se detectan los fallos y se corrigen.

La revisión de casos es controlada en la mayoría de los sistemas por un operador humano, o se basan en reglas elaboradas por el criterio experto, es decir, las regalas establecen pautas que utilizan los expertos para controlar esta revisión, o normas que deben cumplir las soluciones.

En (Giménez, 2006)se presenta una propuesta de adaptación y evaluación de los casos. Esto se realiza mediante el método RCR (Reaplication cost and relevance) que se basa en la observación de la adaptación que el operador realiza sobre el caso. De esta forma se crean guías de ayuda al operador en el momento de la adaptación, así como su revisión para comprobar que la adaptación se ha realizado correctamente. Estas guías también son utilizadas para predecir la dificultad de la adaptación, para ello se da la información necesaria con los costos de adaptación de los casos mediante estimaciones de la diferencia entre los problemas actuales y los casos de adaptación. Una vez el operador confirma la adaptación esta será insertada en la base de casos.

1.3.5 **Retener o Aprender un nuevo caso**

Después que la etapa de revisar confirma que la adaptación es correcta esta será retenida en la BC. La BC se enriquece con las soluciones de nuevos problemas. La forma de estructuración de la BC

y las políticas de retención del sistema facilitarán el buen funcionamiento. Por esto, se debe decidir correctamente qué casos se pueden aprender. La eficiencia del sistema se ve afectada cuando el número de casos crece excesivamente, por lo que es importante evitar incluir casos que no aporten información nueva al sistema.

La retención se vincula a la estructura de la BC, dado que la complejidad de la misma afecta este proceso. Por ejemplo, si la organización es lineal, basta con añadir un nuevo elemento a la lista, pero si a su vez esta se induce a partir de los casos, será necesario redefinirla periódicamente.

Una propuesta a esto es dada por (Tsatsoulis, 1989) y se basa en aprendizaje sobre qué casos mantener. Un ejemplo de estos es el sistema llamado IDIOTS (Intelligent System for Design of Telecommunications Systems). Para ello, se diseñó un sistema dinámico basado en una metodología que proporciona la organización de la BC conceptualmente relacionada. De esta forma, los casos se almacenan en varios niveles de abstracción y al mismo tiempo el conocimiento se organiza para una rápida recuperación (Ahn and Medin., 1992).

1.4 Principales Aplicaciones de RBC

El Razonamiento Basado en Casos es aplicable a una amplia gama de situaciones reales, abarcando desde el conocimiento de situaciones sencillas, en las cuales los casos proporcionan directamente el conocimiento requerido hasta las situaciones importantes en las cuales la construcción de soluciones es compleja.

Esto tiene varias ventajas, pues permite proponer soluciones rápidas a los problemas, razonar en situaciones que no son bien entendidas, evaluar soluciones cuando los métodos algorítmicos no están disponibles, evitar repetición de problemas, y enfocarse en las partes importantes de una situación.

Otras ventajas serían que facilitan la adquisición de conocimiento. Como tecnología para el desarrollo de sistemas basados en el conocimiento facilitan el aprendizaje. Cada vez que se soluciona un nuevo problema este puede ser incorporado como un nuevo caso a la BC permitiéndoles aprender por simple acumulación de casos. Son razonadores muy eficientes, suele ser menos costoso modificar una solución previa que construir una nueva solución desde cero. Los casos pueden proporcionar también "información negativa", alertando sobre posibles fallos. Se facilita el mantenimiento de la base de caso, dado que los usuarios pueden añadir nuevos casos sin ayuda de los expertos.

Janet Kolodner (kolodner, 1993) describe una serie de características que permiten identificar aquellos dominios en los que la aproximación basada en casos tiene probabilidades de éxito. Las aplicaciones de los sistemas RBC se pueden clasificar en dos grandes grupos:

1. Clasificación e interpretación de situaciones

2. Resolución de problemas

En general las aplicaciones industriales están dentro del primer grupo, donde las técnicas a utilizar están mejor definidas, la representación de los casos es más sencilla y existen herramientas comerciales que soportan el desarrollo. Los sistemas de resolución de problemas, por el contrario son más habituales en el ámbito académico ya que necesitan representaciones más complicadas y más esfuerzo de adquisición y formalización de conocimiento.

En el ámbito de los sistemas RBC de clasificación e interpretación destacan los siguientes dominios de aplicación:

- Servicios de atención al cliente (help-desk). si el producto sobre el que se presta soporte es sofisticado, entonces es necesario que la persona que resuelve las dudas de los clientes sea un experto, pero claro, a una empresa le resulta muy costoso dedicar personal cualificado a una tarea repetitiva, ya que los problemas de los clientes tienden a repetirse (Giménez, 2006). De esta forma el experto sólo será consultado cuando el cliente planteé un problema que no haya sido resuelto previamente. Naturalmente cada vez que los expertos resuelven un nuevo problema, éste pasa a formar parte de la base de casos.
- Diagnóstico y resolución de fallos. Sirven como una "memoria corporativa" a través de la cual distintos expertos comparten el resultado de su actividad diaria (Cordero Morales et al., 2013, Bregón et al., 2005).
- Predicción y valoración. En dominios difíciles de formalizar la casuística es una fuente valiosa de predicciones, resultan aptos para las aproximaciones basadas en casos(Rosa, 2009).
- Enseñanza: Un sistema de este tipo suele incluir, además de un conjunto de ejemplos de valor pedagógico, un entorno simulado donde el alumno realiza algún tipo de interacción, de forma que en base a dicha interacción, el sistema sea capaz de sugerir ejemplos que aporten información relevante a la tarea que en ese momento se esté realizando(Sánchez; et al., 2009).

- En sector de generación de energía existe implementado un SIG que representan un adelanto digital. El SIG está integrado al SIGERE, que cuenta con una vasta información alfanumérica y en el cual se pueden realizar una gran variedad de consultas geográficas, pero presenta la dificultad de que la gama de consultas es limitada y la información geográfica que brinda un levantamiento de este tipo de cierta manera está subutilizada por el sistema, y a veces los técnicos solicitan información que el sistema no es capaz de dar. Por lo que el SIG necesita la capacidad de recuperar información de forma automática en tiempo real.

1.5 Conclusiones parciales del capítulo.

Los sistemas RBC constituyen una alternativa factible para construir un SBC donde el conocimiento del problema se presenta a partir de soluciones a experiencias pasadas.

Los casos se describen en función de rasgos los cuales son relevantes al problema a solucionar y se representan por diferentes dominios, lo cual determina el criterio de comparación a emplear. Resulta conveniente el manejo de similitudes globales ponderadas cuando todos los atributos no son igualmente importantes en la solución del problema

Una organización jerárquica de la base de casos favorece los procesos de acceso y recuperación de casos más similares y debe ser tenido en cuenta en el aprendizaje continuo de casos resueltos.

La presencia de múltiples rasgos objetivos puede derivar en la implementación de varias técnicas de adaptación.

Desarrollo de un sistema inteligente utilizando un enfoque basado en casos

2. CAPITULO 2. DESARROLLO DE UN SISTEMA INTELIGENTE UTILIZANDO UN ENFOQUE BASADO EN CASOS

Partiendo de la problemática presentada a la empresa ATI de la UNE de obtener consultas en tiempo real utilizando un enfoque basado en casos. El presente trabajo desarrolla un sistema inteligente para el desarrollo de consultas dinámicas aprovechando la experiencia acumulada en forma de consultas estáticas en el SIGERE y manejando atributos de diferente índole.

2.1 Adquisición del conocimiento para la construcción de la Base de casos

SIGERE está integrado por todos los equipos, instalaciones, infraestructura y acciones que forman la red de transmisión y distribución. El sistema debe recoger datos técnicos, económicos y de gestión que faciliten la dirección, operación, explotación y planificación de las redes. El sistema está orientado al cliente, esto permite reducir costos operativos y mejorar la calidad de suministro (Alfredo Castro). Como se observa en la Figura 4 SIGERE se encuentra dividido en 5 subsistemas que agrupan diversos módulos según su funcionalidad, facilitando la interacción con el medio, que en el caso de los sistemas de esta magnitud es muy activa (Fernández et al., 2008).

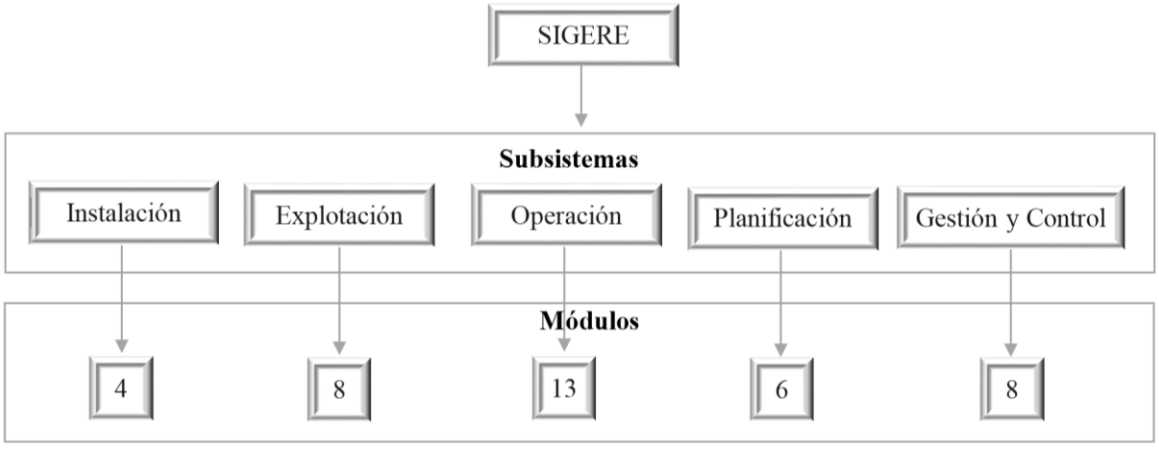


Figura 4: Estructura del Sistema SIGERE

El SIG forma parte del SIGERE y brinda la información relativa a la parte operativa y a la planificación permitiendo de forma rápida:

- Localizar las quejas de interrupciones de la población.

- Localizar una instalación fallida.
- Organizar el recorrido de los carros de forma eficiente, disminuyendo el consumo de combustible.
- Ver en el mapa las zonas con defectos de instalaciones, circuitos con conductores inadecuados o con mal aislamiento, postes en mal estado, entre otros.
- Permite localizar las instalaciones con parámetros diferentes.
- Visualizar en el mapa, los voltajes de los clientes y detectar problemas de voltaje en los mismos.
- Hacer un estudio de fallas de equipamiento por zonas rurales.
- Tener la ubicación de transformadores sobrecargados y tomar acciones.
- Observar los circuitos de alumbrado con problemas y las luminarias o bombillas fundidas y realizar acciones.

Así como planificar:

- Optimización del uso de las redes.
- Expansión óptima de las redes de distribución al conocer en detalles las características de las redes existentes y su ubicación en el mapa.
- En determinadas escalas, permite dibujar el croquis de los nuevos proyectos con la exactitud necesaria, lo que disminuye los costos totales del proyecto.

A pesar de estas funcionalidades han sido numerosas las recomendaciones hechas al SIGERE y en especial al SIG por diferentes tipos de usuarios encargados de la gestión de la cartografía digital. Entre las observaciones hechas al SIG se encuentran limitaciones para:

- Identificar localizaciones geográficas donde son necesarias mejoras en los circuitos y acciones de mantenimientos.
- Dar respuestas a algunas peticiones geográficas que requieren los especialistas de las empresas eléctricas provinciales.
- Satisfacer las expectativas de los especialistas en la recuperación de los mapas temáticos.

Como se explicó anteriormente para alcanzar niveles óptimos en la toma de decisiones se requiere almacenar todas las consultas estáticas requeridas incrementándose la base de datos que almacena esto con un agravante adicional que puede llevar a consultas no existentes en las ya almacenadas.

Además de un proceso engorroso de consultas estáticas, pues una consulta simple del usuario se convierte para el sistema en dos consultas complejas como se puede observar en el ejemplo siguiente:

Consulta del usuario:

```
SELECT CodigoInst, Conexion, Seccionalizador, Circuito, Codigo, Capacidad
      FROM Punto, BancoTransformadores, ConexionBanco,
       Capacidades, Transformadores
      WHERE BancoTransformadores.Circuito = VALOR
```

Consultas del sistema para dar respuesta al usuario:

- Consulta al SIGERE:

```
SELECT Punto.CodigoInst, ConexionBanco.Conexion,
      BancoTransformadores.Seccionalizador,
      BancoTransformadores.Circuito, Punto.Codigo, Capacidades.Capacidad
      FROM BancoTransformadores INNER JOIN Punto ON
       BancoTransformadores.Codigo = Punto.CodigoInst INNER JOIN
       ConexionBanco ON BancoTransformadores.Conexion =
 ConexionBanco.IdConexion INNER JOIN Transformadores ON
 BancoTransformadores.Codigo = Transformadores.Codigo INNER JOIN
 Capacidades ON Transformadores.Id_Capacidades =
 Capacidades.Id_Capacidades
       WHERE (BancoTransformadores.Circuito=VALOR)
```

- Consulta al SIG:

```
 SELECT poste
 FROM Elemento
       WHERE poste = Postes_M.CODIGO_POSTE
```

El problema es cómo lograr consultas inteligentes en tiempo real partiendo de las consultas almacenadas y teniendo en cuenta la diversidad de rasgos predictores a manejar

Base de casos

La BC contiene la descripción de consultas estáticas realizadas previamente en forma de casos. Cada consulta se compone por 11 rasgos fundamentales derivados de la ingeniería del conocimiento realizada, de los cuales 8 son predictores y 3 objetivos. La ***Figura 5*** muestra la estructura de los mismos.

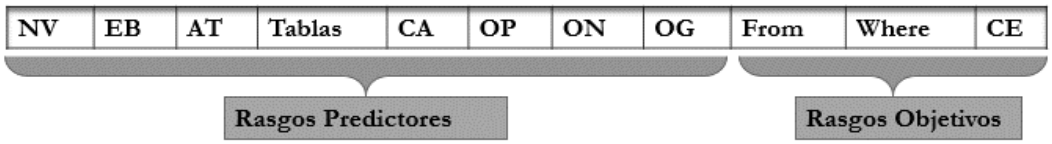


Figura 5: Estructura de un caso.

Un caso almacena como rasgos predictores:

- ***NV***: almacena el nivel de voltaje del circuito, es de tipo simbólico y univaluado, entre los valores que puede tomar se encuentran: Secundario, Primario, Sub-transmisión, Transmisión, entre otros.
- ***EB***: almacena el elemento base de la red que se va a consultar, es de tipo simbólico y univaluado y toma valores como: Postes, Bancos de transformadores, Bancos de capacitores, Grupos generadores, Desconectivos, o cualquier elemento de la red.
- ***AT*** y ***Tablas***: almacenan lo que el usuario desea conocer y las tablas del SIGERE donde se encuentra esta información, respectivamente. Como el usuario puede consultar más de un elemento en la misma consulta y estos a su vez encontrarse en diferentes tablas, estos rasgos son de tipo conjunto y toman como valor $\mathrm{AT} = \{\mathrm{AT}_1, \mathrm{AT}_2, \ldots \mathrm{AT}_n\}$, donde AT_i es un atributo y $Tablas = \{T_1, T_2, \ldots T_n\}$, donde T_i recibe una tabla que almacena algún elemento de AT.
- ***AC***: contiene el atributo por el cual el usuario va a restringir la consulta, de tipo simbólico y univaluado, y toma como valor cualquier atributo de las tablas consultadas.
- ***OP***: almacena el operador de restricción, de tipo simbólico y univaluado, y puede tomar valores como $<, >, \leq, \geq, =, like$, entre otros.
- **OG** y ***OE***: son las ontologías de restricciones, generales y espaciales respectivamente, basadas en lógica descriptiva. Las ontologías definen conocimiento de dominio a nivel genérico que se pueden aplicar a distintas situaciones. Estas definen los términos tratados en el sistema y las relaciones básicas para la compresión del área del conocimiento, así como las reglas para poder combinar los términos que definen las extensiones de este tipo de vocabulario (Belén, 2002). *OG* y *OE* establecen relaciones entre los elemento de la red y entre sus conexiones. Un ejemplo de *OG* sería:

  $T \cap TP \cap TMonofásicos \neg SSecundaria$, el cual expresa que el elemento es un trasformador primario monofásico sin salida secundaria. Como se puede observar, a medida que la ontología se hace más profunda el elemento a consultar cumple con requisitos que pueden ser flexibles al usuario; para que un elemento sea similar a otro no necesita estar al mismo nivel en la ontología, pero sí haber atravesado por la misma rama, por tanto el grado de importancia de cada nivel va disminuyendo a medida que se profundiza en el árbol. Un elemento en el cual $OG = T \cap TP \cap TMonofásicos$ es muy similar al ejemplo planteado anteriormente, aunque

carece de un nivel de profundidad, está por la misma rama. Otro caso sería $BT \cap T \cap TPot \cap TCto \neg Ssecundaria$, que aunque tiene elementos en común con los ejemplos anteriores no sigue la misma rama.

OE funciona muy similar, pero la relación es espacial, un ejemplo que hace referencia a la ubicación de un elemento sería $P \cap Prov \cap Muncp$ el cual expresa que un elemento pertenece al país, a una provincia y a un municipio.

Por la naturaleza de estos rasgos y siendo consecuente con el estado del arte, las ontologías se trabajaron como cadenas, estableciéndose la similitud entre las subcadenas.

Y como rasgos objetivos:

- ***From*:** De tipo cadena y devuelve el From de la consulta al SIGERE.
- ***Where**:* De tipo cadena y devuelve el Where de la consulta al SIGERE.
- **CE:** De tipo cadena y devuelve la consulta la SIG.

Para organizar la base de casos se utiliza una estructura jerárquica, lo cual favorece al sistema el proceso de acceso y recuperación a los ejemplos más similares a la consulta en tiempo real.

Como se observa en la Figura 6, la estructura jerárquica fue diseñada de la siguiente forma: *NV*, nodo raíz; en segundo y tercer nivel *EB* y *OP*, respectivamente, por ser estos elementos los más discriminativos; quedando en los nodos hojas, sub-conjuntos de casos que representan aquellos ejemplos que el valor de *NV, EB, OP* coinciden.

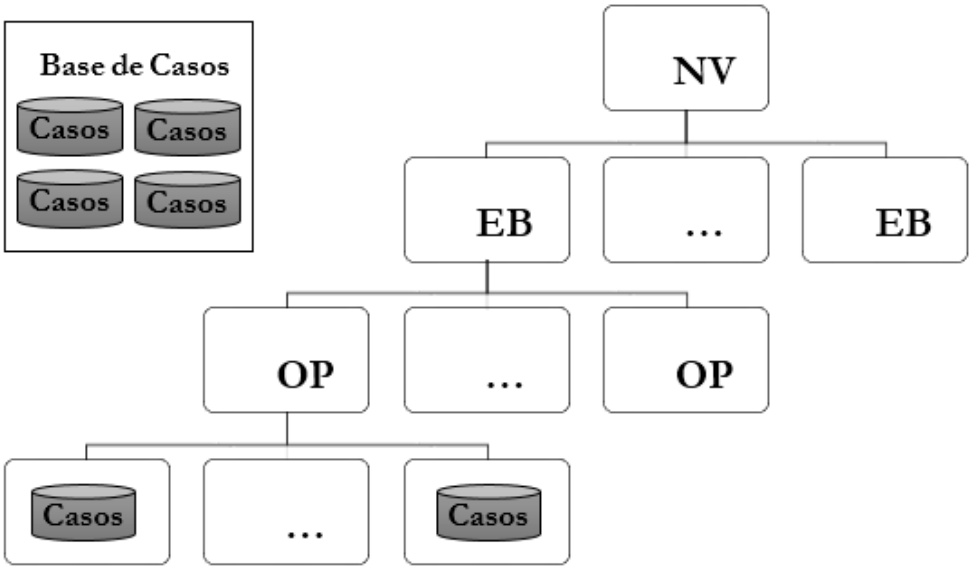


Figura 6: Estructura de la Base de Casos del SICUNE.

2.2 Motor de inferencia

El **motor de inferencia** es la máquina de razonamiento del sistema, el cual compara el problema insertado con los que están almacenados en la base de casos, como resultado infiere una respuesta con el mayor grado de semejanza a la que se busca, adaptando los casos más similares recuperados. A continuación se explicara cómo funciona cada parte del ciclo en el sistema propuesto.

2.2.1 **Módulo recuperador**

El módulo de recuperación se encarga de extraer de la base de casos el caso o los casos más parecidos a la situación actual, utilizando una medida de similitud efectiva que permita una buena recuperación de casos. Esta recuperación se basa en una distancia local que determina la distancia entre valores de un mismo rasgo y una similitud global, que combina los resultados de ciertas distancias locales a todos los rasgos de los casos a comparar.

La similitud global es el resultado de la sumatoria ponderada de las distancias entre el valor de cada rasgo en un caso y el valor que toma el mismo en el caso problema. Esta similitud está determinada por la Ecuación 1. Las distancias son ponderadas teniendo en cuenta el criterio de experto, con un peso w_i, para hacer que el caso problema se acerque más a un caso en particular; w_i es definido debido que los rasgos no tienen la misma importancia, es decir, mientras mayor sea w_i mayor será la importancia del rasgo.

$$\mathrm{SimGlobal}(X,Y) = \sum_{i=0}^{m} w_i * d_i(x_j, y_j)/n \quad \text{donde} \quad \sum w_i = 1 \tag{1}$$

La distancia local $d_i(x_i, y_i)$ está determinada por el tipo de datos de x_i, y_i. En el caso expuesto se cuenta con tres tipos de datos para los cuales se utilizan distintas medidas de distancia que se describen a continuación.

Los rasgos *NV*, *EB*, *AC* y *OP* son tipo simbólico univaluado, la distancia utilizada es de tipo booleana, es decir toma valor 0 o 1 según una condición. La distancia seleccionada es la mostrada en la ecuación 2. Se puede observar que toma valor 1 si el valor x_i es distinto al de y_i o valor 0 en caso contrario (Althoff et al., 1995).

$$d_i(x_j, y_j) = \begin{cases} 0 \text{ Si } x_j == y_j \\ 1 \text{ Si } x_j \neq y_j \end{cases} \tag{2}$$

Los rasgos *AT* y *Tablas* son de tipo conjunto. En (González, 2010) se analizan propuestas para este tipo de distancia. Como resultado de las pruebas realizadas al sistema se implementa la distancia Jaccard (Ecuación 3) que se basa en las operaciones entre conjunto unión e intercepción.

$$d_i(x_j, y_j) = \frac{|x_j \cap y_j|}{|x_j \cup y_j|} \tag{3}$$

Los rasgos OG y OE representan las ontologías general y espacial ver

Anexo **2**, respectivamente. Al ser tratadas como cadenas cada uno de sus componentes serían sub-cadenas, prestando mayor interés al orden que aparecen estas subcadenas sobre la cantidad de cadenas similares. Se implementaron Brecha Afín (Gotoh, 1982), Edición (Ramírez and López, 2006), Smith-Waterman (Smith and Waterman, 1981), Jaro (Jaro, 1976), JaroWinkler (Winkler, 2000) y basado en una muestra de control y una prueba de campo se selecciona la distancia de *JaroWinkler*, ver Ecuación 4. En la ecuación l es la longitud de prefijo común de las cadenas hasta un máximo de 4 caracteres; p es un factor de escala constante que no debe ser mayor que 0.25, de lo contrario la distancia puede dar valores mayores que 1 (0.1 es el valor estándar que se utiliza), b_t es el umbral que se define (se recomienda 0.7), d_{jaro} es la distancia de Jaro entre las cadenas x_j, y_j, la cual se define por la ecuación 5 (Winkler, 2000).

$$d_i(x_j, y_j) = \begin{cases} d_{jaro} & si \quad d_{jaro} < b_t \\ d_{jaro} + \left(l * p * \left(1 - |y_j|\right)\right) & e.o.c \end{cases} \tag{4}$$

$$d_{jaro}(x_j, y_j) = \frac{1}{3}\left(\frac{c}{|x_j|} + \frac{c}{|y_j|} + \frac{c - t/2}{c}\right) \tag{5}$$

El recuperador basado en el Algoritmo 1 obtiene los k casos más cercanos a la consulta solicitada por el usuario, tomando para el cálculo de la distancia la Ecuación 1. Resultante de la evaluación del sistema el k seleccionado por defecto es 3. El conjunto de casos obtenidos por el Algoritmo 1 constituye la entrada del módulo adaptador.

Entrada: P; problema a resolver y $BC' = \{bc_1,\ bc_2,\ \ldots,\ bc_n\}$; sub-conjunto de casos obtenidos de la sub-base que corresponde al caso P.

Salida: Conjunto de K casos más similares.

1. $\forall\ bc_j \in BC'$se cacula la $SimGlobal(bc_i, P)$. Donde:

$$SimGlobal(bc_j, P) = \sum_{i=0}^{8} w_i * d_i(bc_{ji}, P_i)/n$$

$$d_i(bc_{ji}, P_i) = \begin{cases} d_i(bc_{ji}, P_i) = \begin{cases} 0 \text{ Si } bc_{ji} == P_i \\ 1 \text{ Si } bc_{ji} \neq P_i \end{cases} & i = 1,2,5,6 \\ d_i(bc_{ji}, P_i) = \dfrac{|bc_{ji} \cap P_i|}{|bc_{ji} \cup P_i|} & i = 3,4 \\ d_i(bc_{ji}, P_i) = \dfrac{1}{3}\left(\dfrac{c}{|bc_{ji}|} + \dfrac{c}{|P_i|} + \dfrac{c - t/2}{c}\right) & i = 7,8 \end{cases}$$

2. Se devuelven los K casos más similares con un K=3.

Algoritmo 1: Algoritmo recuperador de los k casos más similares.

2.2.2 **Módulo adaptador**

El modulo sigue una analogía transformacional, como se puede observar en la **Error! Reference source not found.** parte de transformar la solución de un problema previo en la solución del nuevo problema a resolver. Se considera que existe un *T-espacio* en el cual la *Solución Conocida* puede ser Transformada, usando *T-operadores*, hasta convertirla en la solución de un nuevo problema.

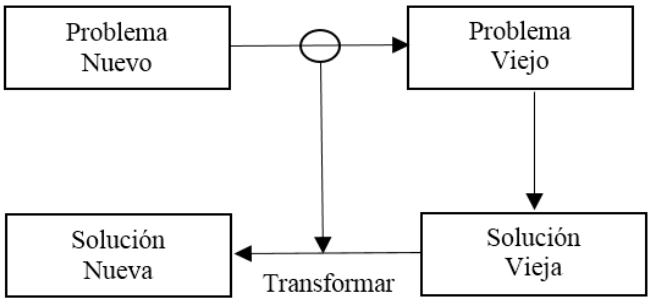


Figura 7: Analogía transformacional.

Basados en la analogía transformacional se desarrolla el ***Algoritmo 2***, donde se propone una solución inicial a partir de los *k* casos recuperados, que es evaluada y adaptada para que esta sea lo más factible posible a las condiciones del nuevo caso. Esto es necesario, dado que normalmente las consultas que se presentan no son idénticas a las ya almacenadas en casos anteriores y surge la necesidad de adaptar una solución existente para que se ajuste al nuevo caso.

Entrada: $CC = \{bc_1, bc_2, \dots, bc_k\}$ subconjunto de casos obtenidos del módulo recuperador.

Salida: From, Where, OE; Rasgos objetivos.

3. Obtener *Solución Inicial*
4. Si *Revisar* =Válida
   Entonces devolver Solución Inicial
   sino
   a. *Adaptar*
   b. *Retener*

Algoritmo 2: Algoritmo para obtener la solución al problema basado en la analogía transformacional.

A continuación se explicara cómo funciona la obtención de una solución inicial, la revisión de esa solución y finalmente la adaptación y retención de la nueva propuesta.

2.2.2.1 Propuesta de una Solución Inicial

En la elaboración de la solución inicial se consideraron todos los casos recuperados. Como la solución da respuesta a un problema, se tomó como punto de partida una combinación de las soluciones recuperadas siguiendo el Algoritmo 2 donde a partir de un subconjunto de casos recuperados se obtiene una solución inicial en función de los tres rasgos objetivos definidos.

Entrada: $CC = \{bc_1, bc_2, \dots, bc_k\}$ subconjunto de casos obtenidos del módulo recuperador.

Salida: From, Where, OE; Rasgos objetivos.

1. Where= MAJORITY ($bc_{i\,WHERE}$) donde i=1…k.
2. Se obtiene CC' donde $CC' \subseteq CC$ y $\forall\, bc_i \in CC'$ bc_{iFROM} contiene Where.Tabla()
3. k=| CC'|
4. From= MAJORITY ($bc_{i\,FROM}$) donde i=1…k.
5. CE = MAJORITY ($bc_{i\,CE}$) donde i=1…k.

Algoritmo 3: Solución inicial

En el Algoritmo 3 se comienza con la búsqueda del valor del rasgo objetivo *Where* utilizando el operador *MAJORITY* (Baldizzoni, 2013). A partir del valor obtenido para el rasgo Where se hace una selección de casos para transformar el conjunto de k casos en un subconjunto CC'; donde cada $bc_i \in CC'$, satisface que su rasgo objetivo *From* contiene la tabla consultada en el *Where*. Finalmente aplicando nuevamente *MAJORITY* se obtiene los restantes rasgos objetivos.

2.2.2.2 Revisión de la Solución Inicial.

La solución inicial se considera válida si cumple las siguientes restricciones.

1. Si el valor del rasgo *OP* es una sub-cadena en *Where*, entonces la solución inicial **válida**.
2. Si el valor del rasgo *AC* es una sub-cadena en *Where*, entonces la solución inicial **válida**.
3. Si los elementos contenidos en el atributo *Tabla* (Rasgo tipo conjunto) son sub-cadenas en *From*, entonces la solución inicial **válida**.
4. Los restantes casos constituyen adaptaciones **no válidas**.

Para comprender estas restricciones se parte que el rasgo Where junto con el From forma una consulta de SQL que satisface la consulta problema. A continuación se explicara las restricciones impuestas a la solución inicial mediante ejemplos.

La restricción 1 comprueba que el operador del caso problema se encuentre en el rasgo objetivo Where, pues de no ser así, la solución inicial nos devolvería consultas opuestas a las que necesita el problema un ejemplo de esto sería:

Rasgo	**Valor**
NV	Primario
EB	Banco de transformadores
AT	CodigoInst, Conexion, Seccionalizador, Circuito, Codigo, Capacidad
Tablas	Punto, BancoTransformadores, ConexionBanco, Capacidades, Transformadores
AC	Capacidad
OP	>
OG	BT∩T∩TPot∩Tcto
OE	P∩Prov

La consulta solicita los datos de los Banco de transformadores que su capacidad excede a un valor. Dado esta consulta, una solución inicial para el rasgo Where sería:

(Capacidades.Capacidad<Valor)

Esta solución inicial no contiene el OP solicitado, como se puede observar, ese valor contrapone la consulta. La solución inicial nos devuelve los menores a ese valor, por lo cual se considera no válida para el problema.

Si la solución inicial estuviera dada por un valor que contuviera el OP solicitado, por ejemplo:

(*BancoTransformadores*.Circuito >Valor)

Fuera válida para la restricción 1, pero al no contener el AC, la solución inicial nos está devolviendo resultados restringidos por un atributo que no está relacionado con la capacidad de los Banco de transformadores, por lo cual a pesar de cumplir la restricción 1 se considera solución no válida por incumplir la restricción 2.

La restricción 3 se estableció dado que las tablas declaradas en la consulta solicitada contienen los atributos a devolver. Las tablas deben estar relacionadas todas en el rasgo objetivo From para poder devolver al usuario una respuesta correcta. A partir del ejemplo anterior una solución inicial para el From seria:

BancoTransformadores INNER JOIN Punto ON BancoTransformadores.Codigo = Punto.CodigoInst INNER JOIN Transformadores ON BancoTransformadores.Codigo = Transformadores.Codigo INNER JOIN Capacidades ON Transformadores.Id_Capacidades = Capacidades.Id_Capacidades

Donde las tablas relacionadas son: BancoTransformadores que almacena de los datos solicitados en *AT* Circuito y Seccionalizador, Punto que almacena Codigo y CodigoInst, Transformadores que no almacena nada pedido pero no afecta la consulta y Capacidades que almacena Capacidad. Como se puede observar falta de los elementos de *AT* Conexión que se encuentra en la tabla ConexionBanco que no fue relacionada en la solución inicial por lo que no se considera valida por ser una solución incompleta.

2.2.2.3 Adaptación en función de la revisión.

Se llega a este paso cuando la adaptación no es válida, para ello resultan necesario operadores de sustitución, sustituyendo algunos valores o elementos de la solución antigua por otros adecuados al nuevo problema, basados en las siguientes reglas.

1. Si *OP* no es el operador del *Where*, entonces reemplazar operador del *Where* por P.OP.
2. Si *AC* no es el Atributo del *Where*, entonces reemplazar Atributo del *Where* por P.AC.
3. Si $\exists X_i \in (\text{Tablas}); X_i \notin From$, entonces adicionar X_i al *From*.

La solución a la consulta realizada resultante a la adaptación inicial o procedente de la revisión de la adaptación se incorpora a la base de casos como ejemplo resuelto aprendido.

2.3 Conclusiones parciales

Se diseña un sistema basado en casos tipo solucionador de problemas utilizando como base de casos inicial las 265 consultas estáticas registradas en SIGERE las cuales se describen con por ocho rasgos predictores del a tipos de datos nominal, conjunto, ontologías y tres rasgos objetivos.

La base de casos responde a una organización jerárquica de tres niveles lo cual favorece los procesos de acceso, recuperación y aprendizaje de casos.

La adaptación combina transformaciones y reglas obtenidas del trabajo con el experto que favorecen la obtención de soluciones validas

La etapa de retención de casos se encuentra en fase preliminar dado que el tamaño actual de la base de casos no presupone reediciones de los casos pues aún resulta de tamaño medio.

Implementación y verificación de SICUNE

3. CAPÍTULO 3: IMPLEMENTACIÓN Y VERIFICACIÓN DE SICUNE

El SICUNE es un Sistema Inteligente de Consulta para la UNE que tiene como objetivo ayudar a la toma de decisiones en la UNE utilizando un enfoque basado en casos. Este sistema se desarrolla como parte del SIG y permite establecer una consulta entre el SIG y el SIGERE utilizando analogía transformacional sobre consultas previamente hechas, recuperadas por un razonador basado en casos, para dar respuesta a interrogantes del usuario. Durante este proceso el sistema utiliza las facilidades del API Jxl de Java para manipular la base de casos.

El sistema soporta el proceso de recuperación y adaptación con eficiencia y eficacia. Utiliza diferentes medidas de distancia para establecer la cercanía entre casos, permitiendo al usuario la selección de las mismas. Permite establecer un grado de importancia a los rasgos, a través de la asignación de pesos. SICUNE posibilita solucionar múltiples problemas cargados de un repositorio y almacenar su respuesta. Fue desarrollado completamente en Java, por lo que es multiplataforma.

3.1 Diseño del SICUNE

Como se puede observar en la Figura 8 se muestra un diagrama que modela la arquitectura en tiempo de ejecución del sistema y la configuración de los elementos de hardware. En el mismo se puede apreciar la relación entre la BC almacenada en Excel, la aplicación programada en lenguaje java que contiene una interfaz desarrollada en el IDE NetBeans 7.3 y el usuario.

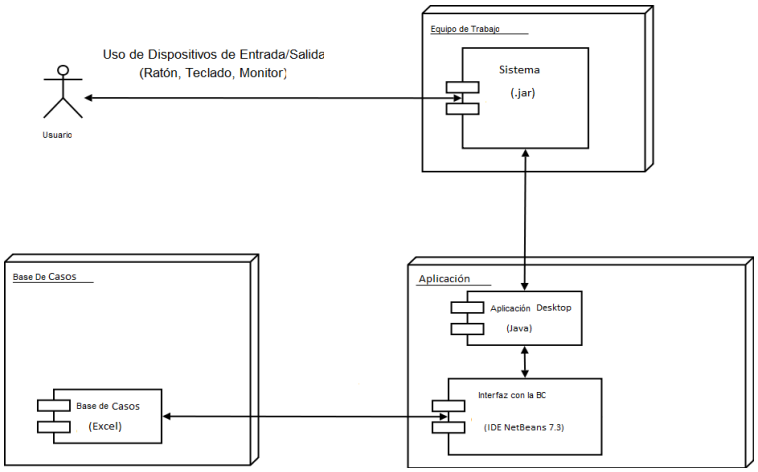


Figura 8: Diagrama de despliegue.

La Figura 9 describe los elementos físicos del sistema y sus relaciones. Se observan los elementos de software que influyeron en la fabricación de la aplicación como son los componentes visuales Consulta y Configuración; los cuales tributan al componente visual Recuperador que interactúa con el componente Adaptador y con la biblioteca jxl.jar; la cual sirve de vínculo con el fichero de la BC. El componente visual Base de Caso interactúa con el jxl.jar que le permite interactuar con el fichero de la BC y envía datos a el componente visual Consulta.

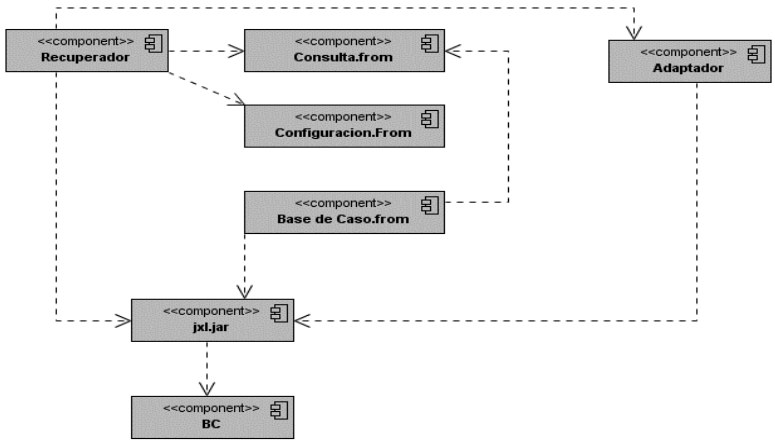


Figura 9: Diagrama de Componente.

Para facilitar la organización, la aplicación se estructuró en cinco paquetes (Figura 10) como mecanismo utilizado para agrupar elementos. Los paquetes **Set** y **String** almacenan todos los elementos relacionados con el trabajo con conjuntos y cadenas respectivamente. Los paquetes **Structure** y **Useful** almacenan todos los elementos relacionados con el acceso y manipulación de la base de casos respectivamente. Todos estos paquetes tributan a un paquete **RBC Visual** que almacena los elementos encargados de establecer el vínculo entre la interfaz y la aplicación.

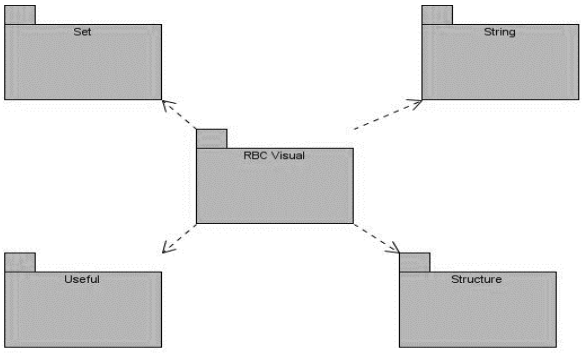


Figura 10: Diagrama de paquete.

En la Figura 11 se pueden visualizar las relaciones entre las clases del sistema, divididas según el paquete donde están almacenadas. Entre las cuales se encuentra:

- *TreeStructure* y *NodeStructure* :contienen los métodos que permiten el acceso jerárquico a la base de casos; pertenecientes al paquete *Structure*.
- *ReadExcel, WriteExcel* y *calcSimilarity* :contienen los métodos parala manipulación de la base de casos y el cálculo de la similitud entre casos; pertenecientes al paquete *USEFUL*.
- *Set, Sets, DistancesSet y WriteExcel* :contienen los métodos para el trabajo con los rasgos tipo conjunto y el cálculo de la similitud entre ellos; pertenecientes al paquete *SET*.
- *DistancesString*: contienen los métodos para el trabajo con los rasgos tipo cadena y el cálculo de la similitud entre ellos; pertenecientes al paquete *STRING*.
- *Visual*: maneja la conexión entre el razonador y la interfaz visual; perteneciente al paquete *RBC Visual*.

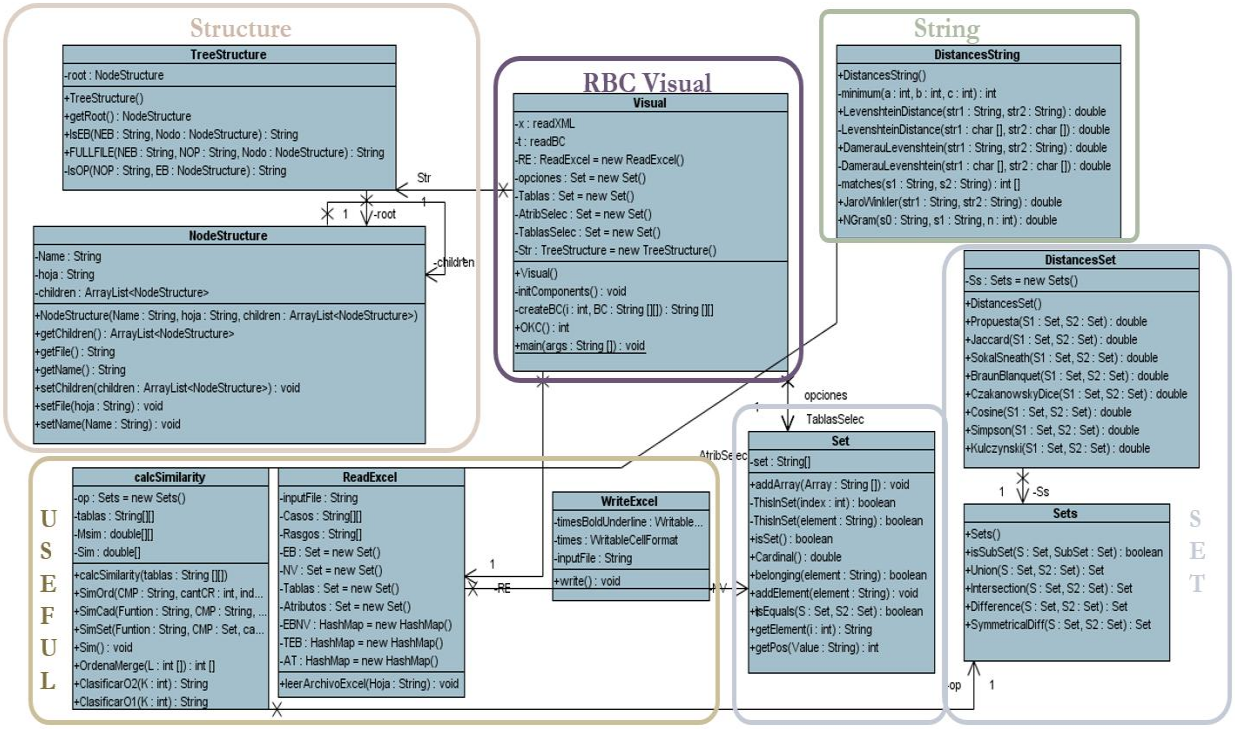


Figura 11: Diagrama de clase.

3.2 Interfaz de usuarios de SICUNE para la toma de decisiones en la UNE.

El sistema cuenta con 6 funcionalidades:

- Configuración de medidas de distancia para los rasgos.

- Asignación de peso a los rasgos.
- Consulta al sistema.
- Observación de la Base de Casos en su totalidad o la sub-base en la que se está recuperando.
- Cargar un conjunto de casos a consultar.
- Almacenar la respuesta al conjunto de datos de los casos cargado.

3.2.1 Opciones de configuración

El sistema permite realizar configuraciones de las distancias y pesos relativos a cada rasgo, por tanto al querer realizar una configuración se puede acceder a la opción 1 para configurar las distancias o a la opción 2 para configurar los pesos (ver Figura 12).

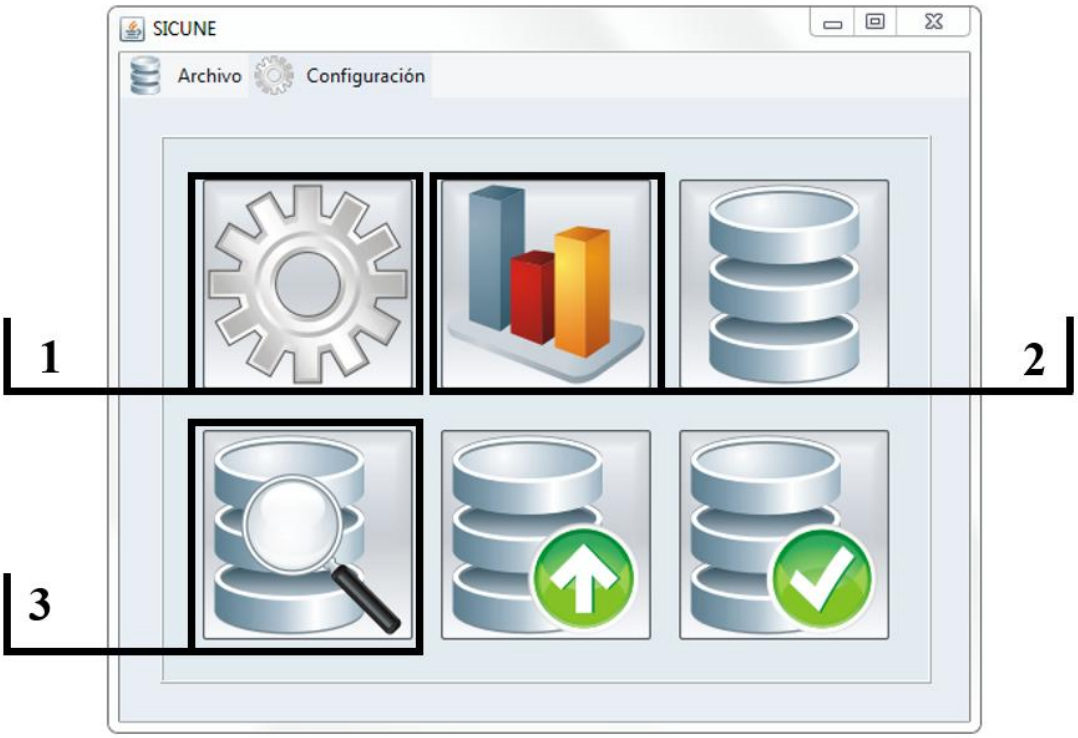


Figura 12: Opciones de Configuración.

Como se observa en la Figura 13 la opción 1 cuenta con tres secciones donde:

- La sección *Rasgos Simbólicos* (1.1) permite seleccionar la distancia a utilizar para los rasgos NV, EB, AT, OP; por defecto tiene la opción *Univaluada* determinada por la ecuación 2.
- La sección *Rasgos Conjuntos* (1.2) permite seleccionar entre las mencionadas en la Tabla 5, la distancia a utilizar para los rasgos AT y Tablas; por defecto tiene la opción *Jaccard* determinada por la ecuación 3.

- La sección *Rasgos Ontologías* (1.3) permite seleccionar entre:
    - Edición (Ramírez and López, 2006)
    - Smith-Waterman (Smith and Waterman, 1981)
    - N-gram *(W Yancey, 2006)*
    - JaroWinkler (Winkler, 2000)

  La distancia a utilizar para los rasgos OE y OG, por defecto tiene la opción *JaroWinkler* determinada por la ecuación 4.

Esta opción para facilitar el trabajo del usuario a la hora de establecer la configuración cuenta con:

- La funcionalidad 1.4 que permite restablecer la configuración por defecto.
- La funcionalidad 1.5 que permite salvar la nueva configuración.

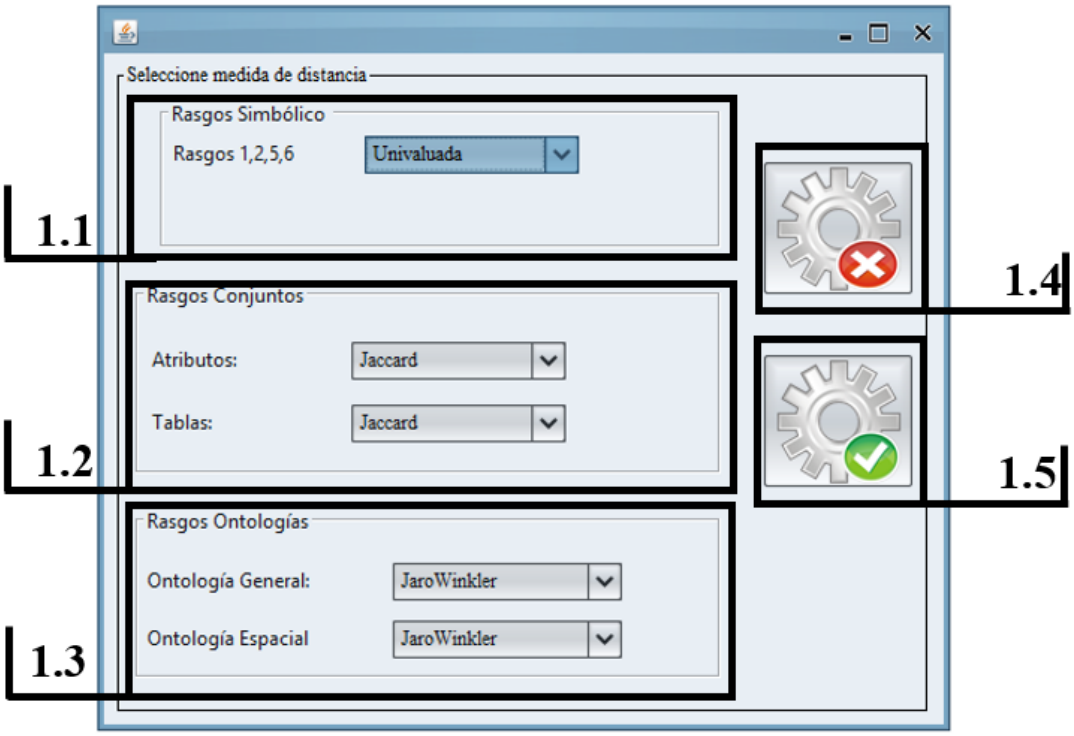


Figura 13: Ventana asociada a la opción de Configurar distancias.

Como se observa en Figura 14 para configurar los pesos, opción a la cual se accede a través de la opción 2 de la interfaz visual (Figura 12*: Opciones de Configuración.*), cuenta con la sección Asignación *de los pesos* (2.1) que permite asignar un peso a cada rasgo.

Esta opción para facilitar el trabajo del usuario a la hora de establecer la configuración cuenta con:

- La funcionalidad 2.2 que permite restablecer la configuración por defecto.
- La funcionalidad 2.3 que permite salvar la nueva configuración.

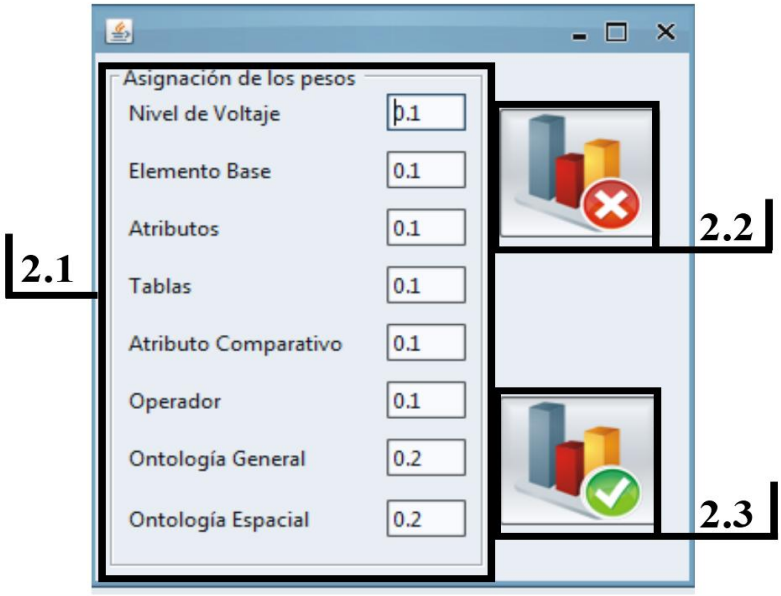


Figura 14: Ventana asociada a la opción de Configurar pesos.

3.2.2 **Opciones de consultas**

Con el objetivo de minimizar el trabajo y los errores del usuario, se diseña una ventana que permite realizar la consulta de forma sencilla, a esta se accede desde la opción 3 de la interfaz visual, Figura 12.

Para realizar una consulta el usuario tiene que introducir los valores de los rasgos predictores. En algunos casos para facilitar el proceso se hace de forma de selección como son: NV, EB y operador. Los rasgos Tablas, AT y AC se hacen a través de selección condicionada o inserción. Los rasgos ontológicos OG y OE se hacen mediante la inserción.

Como se observa en la Figura 15 la opción 3 cuenta con cuatro secciones donde:

- La sección *Seleccione nivel de voltaje y elemento base* (3.1): Permite seleccionar el Nivel de Voltaje (NV) de las opciones que aparecen en el combobox de la izquierda, y después de seleccionado el NV, el combobox de la derecha se actualiza con los Elementos Bases (EB) pertenecientes al NV seleccionado, permitiendo seleccionar el EB que se desea consultar en el combobox de la derecha.
- La sección *Seleccione Tablas y Atributos a consultar* (3.2): Permite seleccionar las tablas y sus atributos a consultar de la siguiente manera:

- Se selecciona una tabla de la sección *seleccionar tablas* que se encuentra a la izquierda y después de seleccionada, la sección *seleccionar atributos* a la derecha, se actualiza con los atributos correspondientes a la tabla seleccionada escogiendo los Atributos a consultar de la misma. La sección *seleccionar atributo a comparar* (3.3): Permite seleccionar el atributo y el operador por el cual se restringirá la consulta.
- La sección *ontología* (3.4): Permite introducir la ontología espacial y general que restringen el elemento a consultar.

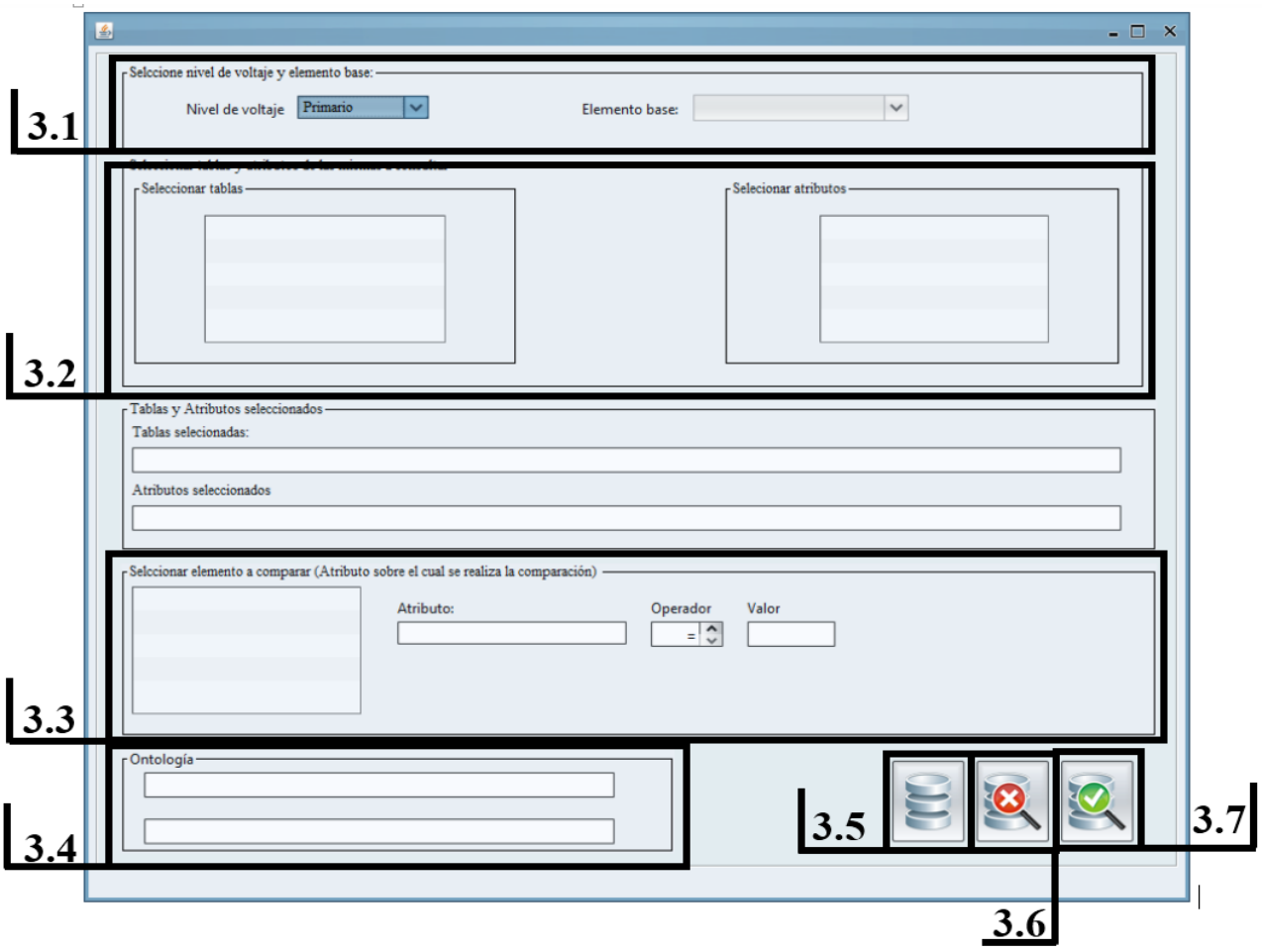


Figura 15: Ventana de Consulta.

Esta opción para facilitar el trabajo del usuario a la hora de realizar la consulta cuenta con:

- La funcionalidad 3.5 permite observar la sub-base en la cual se establece la recuperación.
- La funcionalidad 3.6 permite terminar la consulta.
- La funcionalidad 3.7 permite realizar la consulta y observar la respuesta del sistema.

3.2.3 Funcionamiento del sistema para obtener el resultado de una consulta

Dada la siguiente consulta:

Rasgo	Valor
NV	Primario
EB	Banco de transformadores
AT	CodigoInst, Conexion, Seccionalizador, Circuito, Codigo, Capacidad
Tablas	Punto, BancoTransformadores, ConexionBanco, Capacidades, Transformadores
AC	Capacidad
OP	>
OG	BT∩T∩TPot∩Tcto∩Ttrif
OE	P∩Prov
From	¿?
Where	¿?
CONSE	¿?

El sistema recupera los *K* casos más similares utilizando el Algoritmo 1; los cuales sirven de entrada al Algoritmo 2***Algoritmo 1***, a través del cual se obtiene una solución a la consulta. Como se puede observar en la Figura 16 la solución se compone de los rasgos From, Where, CONSE; los cuales permitirán desarrollar una nueva consulta SQL para el SIG que permita devolverle al usuario los datos que desea conocer de la red.

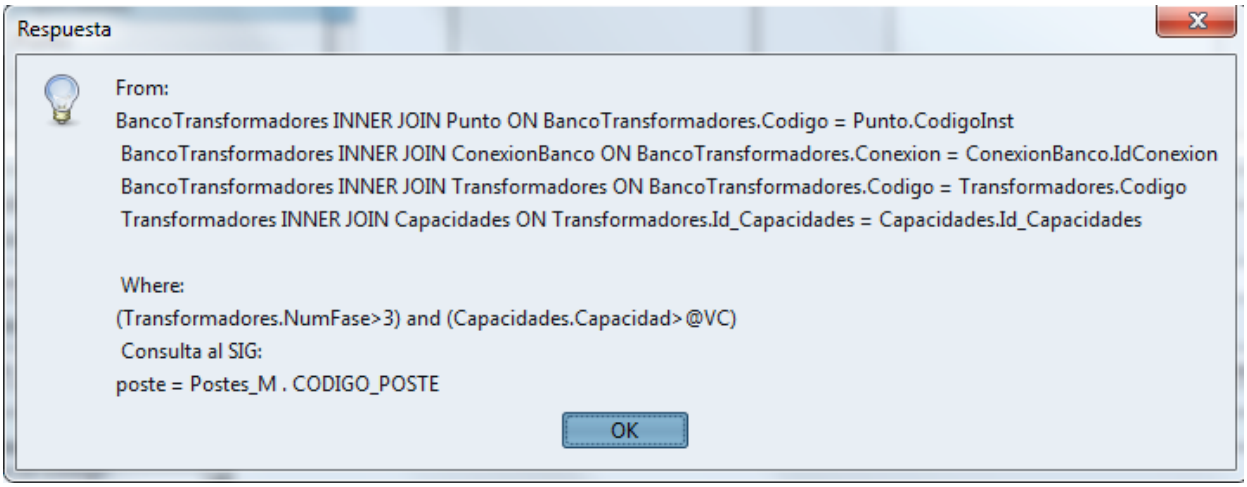


Figura 16: Solución al problema.

3.3 Evaluación

La evaluación incluye la verificación y validación del sistema. Se debe verificar que el sistema se construyó correctamente y que efectivamente es el producto que satisface los requerimientos.

La verificación se realizó mediante criterio de expertos, probando el sistema y comparando lo que recuperaba con lo que los expertos definían como casos más similares, de igual forma se realizó con los resultados ofrecidos y se transformaron aspectos del adaptador para lograr los resultados deseados por los expertos.

En la validación del sistema propuesto se utilizó un conjunto de casos suministrados por la empresa ATI. Se realizó una validación cruzada dejando uno fuera (Leave One Out Crossvalidation), en la cual se separan los datos de forma que para cada iteración se tiene un caso de prueba y el resto de entrenamiento. Sus ventajas están dadas porque la estimación del error es mucho más estable y es mucho mayor el error en la base de prueba que en toda la base de casos. Sus desventajas son que la programación se vuelve mucho más complicada y el tiempo de ejecución puede ser muy alto, pues se debe generar el modelo n veces. Dando como resultados una precisión de 97%.

3.4 Conclusiones Parciales

Se implementa un sistema inteligente de consultas en tiempo real para la UNE (SICUNE) basado en consultas estáticas del SIGERE, portable con una interfaz amigable y que cumple los requerimientos solicitados por el usuario con opciones que favorecen continuar las pruebas del mismo.

La validación del sistema se realizó con casos de prueba obteniéndose un 97 % de exactitud en las respuestas esperadas.

SICUNE constituye un módulo del SIGERE y para su integración al mismo se deberá continuar con la evaluación que presuponga nuevas reglas en el proceso de revisión de casos.

CONCLUSIONES

Se implementa un sistema inteligente de consultas en tiempo real para la UNE (SICUNE) basado en consultas estáticas del SIGERE, portable con una interfaz amigable y que cumple los requerimientos solicitados por el usuario con opciones que favorecen continuar las pruebas del mismos.

- Los casos almacenan las consultas estáticas registradas en SIGERE las cuales se describen con por ocho rasgos predictores del a tipos de datos nominal, conjunto, ontologías y tres rasgos objetivos tipo cadena.
- La similitud entre dos casos se determinó por la suma ponderada de la distancia de sus rasgos. La forma de calcular la distancia entre rasgos se realizó en función de la naturaleza de estos. Para los rasgos de tipo nominal se calculó una distancia booleana, la distancia entre los rasgos de tipo conjunto se determinó mediante Jaccard y las ontologías se trataron como cadenas utilizando la distancia Jaro Winkler. La evaluación evidencia la factibilidad de la similitud propuesta.
- La base de casos responde a una organización jerárquica de tres niveles lo cual favorece los procesos de acceso, recuperación y aprendizaje de casos.
- La validación del sistema se realizó con casos de prueba obteniéndose un 97 % de exactitud en las respuestas esperadas correspondiéndose con todas las reglas de adaptación y revisión propuesta El 3% de error viene dado porque aún es insuficiente los casos almacenados, lo cual se encuentra en vías de solución por parte de ATI.

RECOMENDACIONES

- Continuar el perfeccionamiento de la fase de adaptación, haciendo énfasis en las reglas que se refieren a la inclusión de tablas faltantes en el rasgo objetivo *From*.
- Mejorar la etapa de retención de casos sobre todo cuando crezca el número de casos representativos en la Base de Casos.

REFERENCIAS

AAMODT, A. 1991. *knowledge-intensive, intergrated approach to problem solving and sustained learning.* Ph.D, University of Trondheim,.

AAMODT, A. & PLAZA, E. 1994. Case-based Reasoning: Foundational Issues, Methodological variations, and System approaches. *Artificial Intelligence communication,* 7**,** 39-59.

AHN, W. & MEDIN., D. L. 1992. A two-stage model of category construction. *Cognitive Science,* 16**,** 81-121.

ALFREDO CASTRO , B. M. Sistema de Información Geográfico para el control y operación de las Redes Eléctricas de Sancti Spíritus. .

ALTHOFF, K., AURIOL E., BARLETTA R. & MANAGO M. 1995 *A Review of Industrial Case-Based Reasoning Tools,* AI Intelligence, Oxford UK.

AMÓN, I. 2010. *Guía metodológica para la selección de técnicas de depuración de datos.* Maestría, universidad nacional de colombia.

AZÁN, Y., BRAVO, L., ROSALES, W., TRUJILLO, D., GARCÍA, E. & PIMENTEL, A. 2014. Solución basada en el Razonamiento Basado en Casos para el apoyo a las auditorías informáticas a bases de datos. *Revista Cubana de Ciencias Informáticas,* 8**,** 52-58.

BALDIZZONI, E. 2013. *Un proceso de transformación de datos para proyectos de explotación de información* Licenciado en sistemas Universidad Nacional de Lanus.

BAREISS, E. 1988. *PROTOS: A unified approach to concept representation, classification and learning* Ph.D, University of Texas.

BATEMAN, J. 1990. Upper modeling: organizing knowledge for natural language processing. *5th International Workshop on Natural Language Generation*.

BELÉN, M. 2002. *Una aproximación ontológica al desarrollo de sistemas de razonamiento basado en casos.* Doctor en Ciencias Doctor en Ciencias, Universidad Complutense de Madrid.

BONILLO, M. L. 2003. *Razonamiento Basado en Casos aplicado a Problemas de Clasificación.* Doctor Tesis Doctoral.

BREGÓN, A., SIMÓN, A., ALONSO, C., RODRÍGUEZ, J. J., PULIDO, B. & MORO, I. 2005. Un sistema de razonamiento basado en casos para la clasicación de fallos en sistemas dinámicos. *III Taller Nacional de Minería de Datos y Aprendizaje, TAMIDA2005* 203-211.

COHEN, D. 1997. Recursive Hashing Functions for n-Grams. *ACM Transactions on Information Systems,* 15**,** 291-320.

CORDERO MORALES, D., RUIZ CONSTANTEN, Y. & TORRES RUBIO, Y. 2013. Sistema de Razonamiento Basado en Casos para la identificación de riesgos de software. *Revista Cubana de Ciencias Informáticas,* 7**,** 222-239.

DEVLIN, K. J. 1993. The Joy of Sets. *Springer-Verlag***,** 1-7.

DÍAZ, M. 2002. *Una oproximación ontológica al desarrollo de sistemas de razonamiento basado en casos* Phd Doctorado, Complutense de Madrid.

ELKAN, M. Y. 1996. The Field Matching Problem: Algorithms and Applications. *Second International Conference on Knowledge Discovery and Data Mining***,** 267-260.

EXPÓSITO GALLARDO 2008. Aplicaciones de la Inteligencia Artificial en la medicina: Perspectivas y problemas. *ACIMED,* 17**,** 2-9.

FERNÁNDEZ, R., LIZANO, N., CABRERA, Y., PONCE, R., MACHÍN, J., PLASENCIA, M., DÍAZ, M. E. & MARCOS, A. 2008. Propuesta de nueva visión del sistema de gestion de redes. *APLICACIÓN DE LA COMPUT ACIÓN,* XXIX.

GELBUKH, A., JIMÉNEZ, S., BECERRA, C. & GONZÁLEZ, F. 2009. "Generalized Mongue-Elkan Method for Approximate Text String Comparison". *10th International Conference on Computational Linguistics and Intelligent Text Processing***,** 559-570.

GIMÉNEZ, M. 2006. *Estudio para la implementación de un sistema de razonamiento basado en casos.* Proyecto final de carrera, Universidad Rovira Virgli.

GONZÁLEZ, M. 2010. *Extensión de algoritmos representativos del aprendizaje automático al trabajo con datos tipo conjunto* MSc., UCLV.

GOTOH, O. 1982. An Improved Algorithm for Matching Biological Sequences. *Journal of Molecular Biology,* 162**,** 705-708.

H, R. M. & Q, V. P. 2001. El tratamiento de la imformación financiera con redes neuronales artificiales 4**,** 67-62.

JARO, M. A. 1976. Unimatch: A Record Linkage System: User's Manual. *US Bureau of the Census*.

KOLODNER, J. 1993. Case-Based Reasoning. *Morgan Kaufmann Publishers, San Mateo, California*.

KOLODNER, J. & LEAKE, D. 1996. A TutorialIntroducction to Case-Based Reasoning. *Case-Based Reasoning: experiences, lessons,and future directions***,** 31-65.

LAGUÍA, M. 2003. *Razonamiento Basado en Casos aplicado a Problemas de Clasificación.*

LAURA LOZANO, J. F. 2008. Razonamiento Basado en Casos: Una Visión General 59.

LOPEZ & PLAZAS 1997. Case-Base Reasonig : An Overview. *IA communications,* 10.

MOYA-RODRÍGUEZ, LAUREANO, J., BECERRA-FERREIRO, MARÍA, A., CHAGOYÉN-MÉNDEZ & A., C. 2012. Utilización de Sistemas Basados en Reglas y en Casos para diseñar transmisiones por tornillo sinfín. *Ingeniería Mecánica,* 15**,** 01-09.

PÁEZ, L. O. M., LOZANO, M. R. & DAVILA, J. A. R. 2011. Descripción general de la Inferencia Bayesiana y sus aplicaciones en los procesos de gestión *La simulación al servicio de la academia*.

RAMÍREZ, F. & LÓPEZ, E. Spelling Error Patterns in Spanish for Word Processing Applications. Fifth international conference on Language Resources and Evaluation, LREC 2006, 2006.

RICHTER, M. M. & WEBER, R. O. 2013. *Case-Based Reasoning,* Springer Berlin Heidelberg.

ROSA, T. D. L. 2009. *Razonamiento Basado en Casos Aplicado a la Planificación Heurística.* Php DOCTORAL, CARLOS III DE MADRID.

SÁNCHEZ;, N. M., LORENZO;, M. M. G., VALDIVIA;, Z. Z. G. & LORENZO;, G. F. 2009. El paradigma del razonamiento basado en casos en el ámbito de los sistemas de enseñanza aprendizaje inteligentes.

SANKAR, K. 2004. "Foundations of Soft Case-Based Reasoning". *Wiley Series On Intelligent Systems.*

SMITH, T. F. & WATERMAN, M. S. 1981. Identification of Common Molecular Subsequences. *Journal of Molecular Biology,* 147**,** 195-197.

TSATSOULIS, C. 1989. *Case-Based Design And Learning In Telecommunications.* University of Kansas.

W YANCEY 2006. Evaluating String Comparator Performance for Record Linkage. *Fifth Australasian Conference on Data mining and Analytics***,** 21-23.

WATSON, I. 1997. Applying Case-Based Reasoning: Techniques for Enterprise systems. *Morgan Kaufman Publisher*.

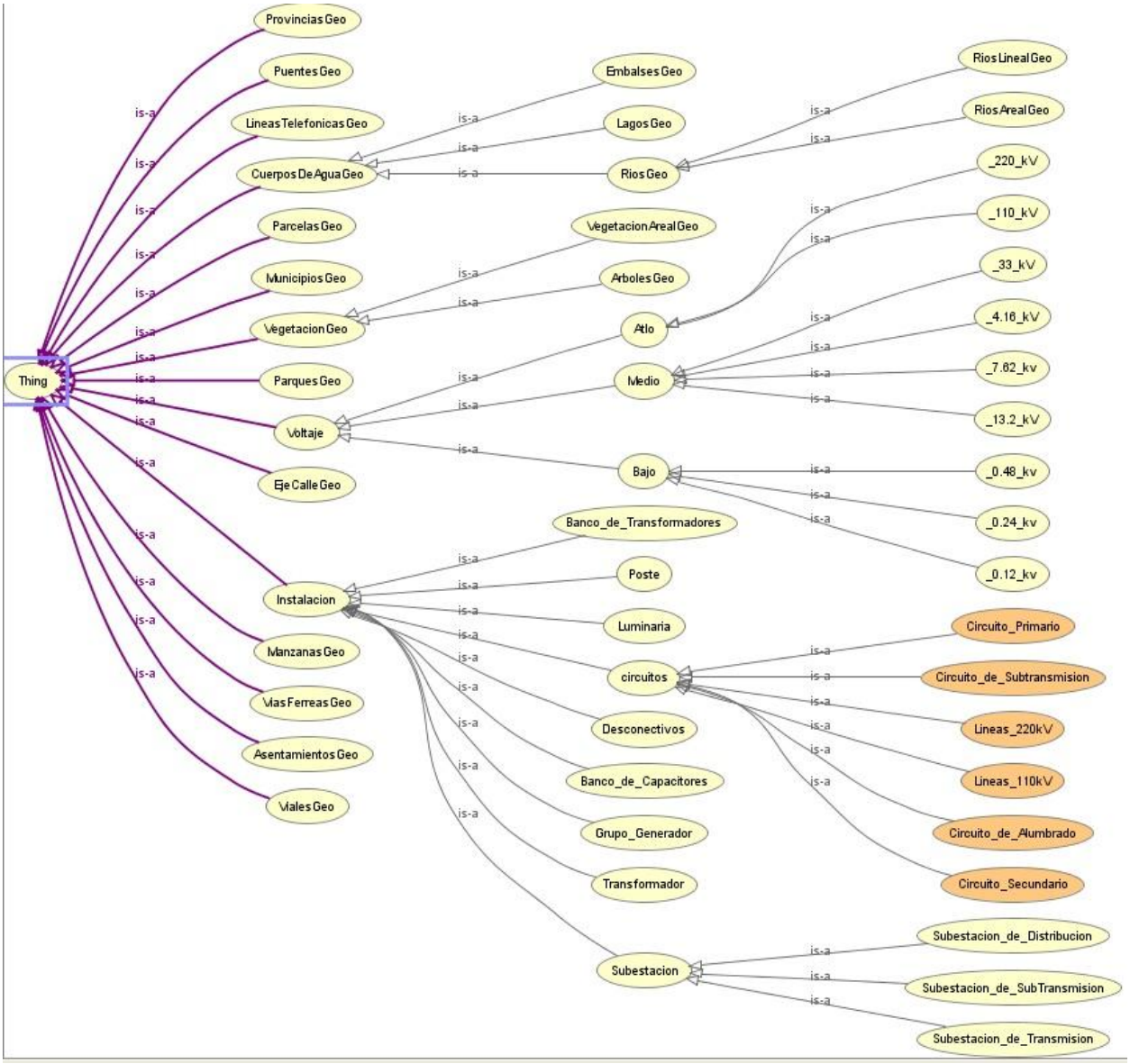
WINKLER, W. E. 2000. "Using the EM Algorithm for Weight Computation in the Fellegi-Sunter Model of Record Linkage". *Proceedings of the Section on Survey Research Methods***,** 667-671.

ANEXOS

Anexo 1: Formas de calcular w_i

Suma	Se ordenan los atributos en orden de preferencia. $$w_i = \frac{n - r_i + 1}{\sum_{k=1}^{n}(n - r_k + 1)}$$ • w_j es el peso normalizado para él $i - ésimo$ atributo. • n es el número de atributos • r es la posición que ocupa el atributo
Recíprocos	Se ordenan los atributos en orden de preferencia. $$w_i = \frac{1/r_i}{\sum_{k=1}^{n} 1/r_k}$$ • w_j es el peso normalizado para él $i - ésimo$ atributo. • n es el número de atributos • r es la posición que ocupa el atributo
Proporción	• Asignar un valor de 100 al atributo más importante y valores menores a los atributos menos importantes. • Obtener el peso de cada atributo dividiendo el valor del atributo menos importante entre el valor de cada uno de los atributos. • Normalizar los pesos dividiéndolos entre el peso total.
Proceso analítico jerárquico	1. Definir la jerarquía de los atributos más importantes del problema. 2. Realizar comparaciones pareadas de los elementos de decisión, valorar la preferencia entre cada par de atributos, utilizando una escala de 1 a 9 y llenar la matriz asumiendo que es recíproca • Desarrollar una matriz de comparación en cada nivel de la jerarquía. 3. Cálculo de los pesos de los atributos • Sumar los valores de cada columna de la matriz • Generar la matriz normalizada, dividiendo cada elemento de la matriz entre el total de su columna. • Calcular el promedio de los elementos de cada renglón de la matriz normalizada. • Estimar la proporción de consistencia 4. Determinar la consistencia • determinar el vector de la suma ponderada, multiplicar el peso del primer atributo por la primera columna de la matriz, el peso del segundo atributo por la segunda columna y así, sucesivamente. • sumar los renglones • dividir el vector de la suma de pesos entre los pesos de los atributos determinados previamente • calcular el valor de lambda promediando el vector de consistencia • calcular el índice de consistencia

Anexo 2: Esquema que representa los conceptos y taxonomía de la ontología propuesta.



Anexo 3: Relaciones de objetos que conforman la ontología.