Universidad Central “Marta Abreu” de Las Villas.
Facultad Matemática, Física y Computación
Licenciatura en Ciencia de la Computación

TRABAJO DE DIPLOMA

Editor de Reglas de Negocio en lenguaje cercano al natural.

Autor: Nestor Ibargollin Pérez.

Tutores: M.Sc. María Elena Martínez del Busto.

M.Sc.Isel Moreno Montes de Oca.

Dra. Luisa Manuela González Glez.

Santa Clara
2009
“Año del 50 Aniversario del Triunfo de la Revolución”

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 Laboratorio

DEDICATORIA

A mis padres que siempre me han apoyado y por todo el sacrificio que han pasado para que no me falte nada en los estudios.
A mi país por darme la posibilidad de estudiar y a todos aquellos lectores que este trabajo les sirva como alguna base para aplicaciones futuras.

AGRADECIMIENTOS

A lo largo de estos años muchas personas han dejado una marca en mi vida como estudiante y les agradezco a todos por de alguna manera influir en mi formación personal y profesional. Esta tesis esta dedicada a todos ellos.

A mis compañeros de cuarto Michel, Alieski, Osmany y Yoan(lokiyo) que tantos momentos inolvidables pasamos juntos, con quienes me divertí y conocí muchos lugares.

A Arsenio por su gran amistad, siempre te recordaré.

A Yimmy por siempre estar ahí cuando lo necesité, a quien le debo mucho y por todo lo que me enseño. A Ernesto, por esos momentos tan divertidos que se pasan cuando se está a su lado. A Annay por manifestarme siempre su disposición en ayudarme.

A Bishoksan que era como mi hermano y que esta de vuelta en su país, nunca te olvidaré.

A todos mis compañeros de aula: Dayron, AnailYs, Erilis, Arazay, Yaise, Leandro, Yanier, Kevin, Yoanny, Yanet, Mavelin, Lisvani, Yunieski, Michel, Lester, Delvis, Sandi, Jose y muchos más que siempre nos mantuvimos unidos y nos ayudamos entre todos.

A Yania y Heidy por ser tan amables y buenas amigas a quienes quiero en demasía.

A Oscarito, Martín, Jose Carlos, Denia, Dennis y Denis por su maravillosa amistad, siempre estarán conmigo.

A mis padres Rosendo Ibargollin y Angela Pérez por todo el amor que siempre me han trasmitido y toda la confianza que siempre han mantenido en mí. También a mis hermanos y demás familiares que me han seguido de cerca en todo lo necesitado para la culminación de mis estudios.

A mi gran tutora María Elena Martínez no sólo por toda su ayuda en la tesis, que ha sido inmensa, sino por acogerme como un hijo más y estar ahí presente siempre que la necesité, también para su esposo Luis Felipe a quien admiro y respeto mucho, a su hijo Asiel con quien he compartido y he llegado a tener una gran amistad y a su hija Claudia, siendo todos para

mí como una familia más, los cuales ya han ocupado un lugar bien grande en mi corazón.
A todos los profesores que han hecho de mí un profesional bien preparado, gracias a ellos.
A los que cotidianamente de alguna manera me han apoyado, ayudado y han convivido estos últimos años en el local de Base de Datos en el CEI como son: Luisa, a quien estimo mucho, Isel, Marta B., Abel, Carlos García, Darien, Beatriz, Cuellar, Norma entre otros.
En fin a todos, aquellos que he mencionado y los que no, pero que siempre voy a recordar.

RESUMEN

Las reglas de negocio (RN) definen y controlan las políticas en el proceso de negocio en una organización. Por tal razón es importante que estas sean captadas en el lenguaje cercano al natural, logrando que el usuario del negocio pueda procesarlas con ayuda de una herramienta de cómputo. Existen diferentes formalismos con relación a expresar las regulaciones de un negocio y respecto a su representación en el tratamiento computacional. El desarrollo de herramientas basadas en tecnologías orientadas a RN puede clasificarse en tres tipos: herramientas independientes de la base de datos, herramientas basadas en servicios y sistemas basados en reglas.

En el presente trabajo se obtiene una herramienta independiente de la base de datos para la edición de RN. El software permite a los usuarios del negocio la manipulación de un conjunto de reglas previamente creado. Dicha herramienta facilita estructurar cada regla utilizando una forma de expresión semejante a la natural. Se escriben según patrones que permiten realizar validaciones de términos y atributos, los cuales han sido definidos como válidos en el modelo de hechos. Finalmente se obtiene un repositorio de reglas en formato XML que permite la integración con otras herramientas de procesamiento y generación automática de RN.

ABSTRACT

Business rules (BR) define and control policies in the business process in an organization. For this reason it is important that these are captured in the language close to natural, achieving the user to process them with a computation tool. There are different formalisms with respect to the regulations of a business and for their representation in the computational treatment. The development of tools based on technologies for RN can be classified into three types: independent tools of the database, tools based service and systems based on rules.

In this work it is obtained an independent tool of the database for editing RN. The software allows business users to handle a set of rules previously established. This tool makes it easy to structure each rule by using a form of expression similar to the natural. Patterns are written according to enable validation of terms and attributes, which have been defined as valid in the model acts. Finally it is get a rules repository in XML format that allows the integration with other tools of processing and automatic generation of RN.

ÍNDICE

INTRODUCCIÓN

El presente trabajo se enmarca dentro de un proyecto mayor de investigación acerca de la modelación de datos y el uso de las reglas de negocio (RN), siendo un objetivo fundamental crear mecanismos que posibiliten el acercamiento de los usuarios o especialistas del negocio al desarrollo de aplicaciones de software. Son estas personas las que tienen el conocimiento profundo y necesario acerca de cada uno de los procesos del negocio, o sea, los expertos en un área de producción o servicios determinada que necesita ser automatizada.

Precisamente son los especialistas del negocio las personas que realmente conocen acerca de sus propios procesos, quedando en manos de ingenieros y desarrolladores de software la posibilidad de lograr mejorar el negocio mediante los sistemas computacionales que respondan a sus necesidades, proporcionando sistemas adaptables y fáciles de modificar.

Resulta indispensable lograr disminuir las barreras de comunicación entre ambos: el especialista en computación y el experto del dominio. Se tiene entonces la necesidad de permitir a los usuarios del negocio establecer sus propias reglas y restricciones, de forma que los sistemas que dan soporte al negocio puedan adaptarse fácilmente a las nuevas necesidades, sin tener que esperar por la asistencia de especialistas de sistema.

En la actualidad es común que ocurran rápidos y constantes cambios en los ambientes de negocio. Estos cambios afectan tanto al negocio como a las aplicaciones que les dan soporte. Como resultado, las aplicaciones requieren renovaciones dinámicas y adaptaciones para cumplir con las necesidades reales del mismo. Una tendencia que aborda dicha problemática es la conocida por “enfoque de reglas de negocio” (Hendryx, 2003). La idea principal de este enfoque consiste en una técnica de dominio del negocio basada en reglas explícitas que son escritas y expresadas en lenguaje claro, cercano al natural y separadas de los datos de la aplicación.

En las últimas décadas las RN se han convertido en un instrumento muy utilizado en el desarrollo de bases de datos y aplicaciones flexibles, mediante ellas, los expertos del negocio pueden especificar políticas y condiciones que deberán verificarse en los sistemas de información que le sirven de soporte (Appelton, 1984). Las reglas deben ser definidas centralmente pudiendo ser reutilizadas, de fácil acceso, modificadas y

administradas por los desarrolladores, así como, por los usuarios del negocio (Bajec, 2000). Al lograrse estos objetivos se obtendrán mejoras en cuanto a la escalabilidad y mantenimiento de las aplicaciones (Moreno Montes de Oca, 2008).

Las reglas, existentes en cualquier organización, deben ser captadas en un lenguaje cercano a la forma en que la comunidad de usuarios expresan sus regulaciones. De acuerdo con el Grupo de Reglas de Negocio (BRG, acrónimo de Business Rules Group), una regla de negocio es "una sentencia que define o restringe algún aspecto del negocio, con la que se pretende imponer la estructura o controlar e influir en el comportamiento del mismo" (BRG, 2000). Una vez capturadas pueden no reflejar ideas precisas o incluso pueden resultar contradictorias entre sí, por tal motivo, se tiene la necesidad de dar tratamiento adicional al repositorio creado con ellas.

El repositorio de reglas necesita ofrecer ciertas facilidades:

- Mostrar las reglas en un ambiente amigable para usuarios del negocio.
- Permitir adicionar comentarios referentes a cada regla
- Posibilitar la modificación de reglas.
- Confirmar la aceptación de las reglas para ser añadidas al repositorio.
- Realizar chequeos de integridad al añadir reglas al repositorio.

Editores de reglas resultan objeto de trabajo para desarrolladores en el mundo del software.

Es común que en un enfoque de RN, la estructura de un sistema se visualice por medio del Modelo de Hechos, donde se modelan relaciones entre conceptos y se enfoca en la estandarización de la terminología del negocio para establecer un vocabulario de negocio, lo que es considerado un punto de partida crucial en la modelación basada en este enfoque (Ross, 1997, Nilsson, 1999, Appelton, 1984a, Bajec, 2006, Bajec, 2000, Barne, 1997, BRG, 2000, BRG, 2003, Ceri and Fraternale, 1997, Date, 2000, Hendryx, 2003, Hüsemann and Schäfer, 2006, Morgan, 2002, Ross, 2003, Ross, 2005, Ross and Lam, 2003, Struck, 1999, Gottesdiener, 1999, Youdeowei, 1997, Rosca et al., 1997, Herbst et al., 1994, Hay and Healy, 1997).

Las ontologías han tenido éxito en proveer un vocabulario para representar y comunicar conocimiento acerca de un tema y el conjunto de interrelaciones que existen entre los términos de ese vocabulario, o lo que es lo mismo, "un glosario de términos ontológicos"

(Gasevic et al., 2006), al proporcionar una comprensión compartida del conocimiento de un dominio que puede ser comunicada entre personas y sistemas heterogéneos (Moreno Montes de Oca, 2008).

El Protégé es considerado por la comunidad científica como una herramienta idónea para el manejo de ontologías. En este proyecto se utiliza una ontología para representar el Modelo de Hechos usando esta herramienta. Esto posibilita exportar las reglas en cualquiera de los siguientes formatos: RDF, OWL y XML, debiendo utilizar el más apropiado para procesamientos posteriores, lo cual justifica el planteamiento del *problema de investigación* siguiente:

Existen procesadores de RN que no permiten a los usuarios plantear las regulaciones del negocio en un lenguaje cercano al natural o lo hacen de forma muy rudimentaria.

El *objetivo general* de la investigación consiste en implementar una herramienta de software que facilite a los usuarios del negocio el manejo de las reglas en lenguaje cercano al natural, permitiendo validaciones sintácticas y semánticas.

Este objetivo general se desglosa en los siguientes *objetivos específicos*:

1. Realizar el estudio acerca de RN y su edición.
2. Seleccionar el formato apropiado para el almacenamiento inicial del conjunto de reglas.
3. Diseñar un modelo para la representación del repositorio de RN.
4. Diseñar el esquema que permita la validación sintáctica y semántica de las reglas del repositorio.

Las *preguntas de investigación* planteadas son:

¿Cuál puede ser el formato apropiado para la representación del repositorio de RN?

¿Cómo estructurar el repositorio de reglas para facilitar su posterior procesamiento a otros módulos o herramientas?

¿Qué validaciones sintácticas y semánticas se logran realizar a cada regla?

¿Cómo propiciar que los usuarios del negocio chequeen y validen las reglas disponibles en el repositorio temporal?

Para lograr los objetivos trazados y demostrar las hipótesis establecidas se acometieron las *tareas de investigación* siguientes:

1. Estudio del enfoque de RN como parte de las nuevas investigaciones en el desarrollo de Sistemas de Información (SI).
2. Análisis de posibles formatos para la representación de las RN capturadas por el editor.
3. Diseño de una estructura para el repositorio de reglas de negocio que facilite su posterior procesamiento.
4. Diseño del esquema de validación semántica y sintáctica para las reglas de negocio.
5. Diseño e implementación de una herramienta que permita a los usuarios del negocio el chequeo y validación de las reglas almacenadas en el repositorio.

El *valor práctico del trabajo* está dado por:
Disponer de una herramienta que facilita la validación de reglas de negocio en el lenguaje del usuario, posibilitando rechazar o confirmar reglas definidas dentro de un determinado problema. Se ofrece la posibilidad de hacer comentarios para una posterior edición de aquellas reglas que no sean totalmente inválidas y merezcan alguna modificación.
Herramientas con estas características facilitan a los usuarios del negocio realizar actualizaciones a sus sistemas de información según las nuevas condiciones. Por otra parte, para los desarrolladores representan una importante ayuda, pues la adaptabilidad de los sistemas es más ágil y sencilla.

La tesis está estructurada en tres capítulos. En el capítulo I se tratan conceptos sobre el trabajo con reglas de negocio, se enfatiza en el análisis de la necesidad del uso de la forma de representación informal para la interacción con los usuarios del negocio. En el capítulo II se caracterizan diferentes formatos para la representación del repositorio de reglas, abordando las ventajas del uso del XML. También son descritas las características a ser validadas mediante la herramienta que procesa el repositorio. En el Capítulo III se aplican los resultados teóricos de la investigación exponiendo el desarrollo de la herramienta de ayuda a la edición de reglas y el papel que ocupa dentro de la arquitectura

del editor de reglas de negocio, se explica el formato resultante para el repositorio. Este documento culmina con las conclusiones, recomendaciones, referencias bibliográficas y los anexos.

Capítulo 1: Reglas de negocio y su forma de representación

Este capítulo recoge aspectos generales sobre conceptos relacionados con el enfoque de reglas de negocio para ofrecer un contexto apropiado para el presente trabajo.

1.1 Antecedentes de las reglas de negocio

La primera aparición de la frase regla de negocio en el contexto de los SI fue en el año 1984, cuando Daniel Appelton escribió un artículo, llamado “Reglas de negocio: el enlace perdido” (Appelton, 1984). Appelton discutió problemas que eran causados por la falta de estandarización de términos de negocios. Su punto de vista consistía en que los analistas del negocio no podían proporcionar soluciones comunes si los usuarios usaban términos que variaban en significado de un departamento a otro dentro de una misma organización.

Luego de la aparición de este trabajo, muchos profesionales de la computación y de los SI comienzan a manejar la idea de que si las reglas de negocio podían ser capturadas en máquinas de procesamiento (como lo son las reglas en los sistemas expertos), entonces dichas máquinas podrían hacer cumplir estas reglas y asegurar que los procesos de negocio fuesen controlados y conducidos de acuerdo a los estándares del mismo, sus políticas y procedimientos (Struck, 1999). Este fue solo el comienzo. Después, los investigadores de los SI comenzaron a trabajar en las reglas de negocio desde varias perspectivas.

El enfoque de reglas de negocio está dirigido a proporcionar a las personas del negocio un control directo sobre el funcionamiento del mismo.

Las reglas de negocio siempre han sido controladas por el negocio, pero en las últimas décadas la administración de las reglas y el control de su cumplimiento ha sido delegado generalmente a los desarrolladores de software, los que han incorporado dichas reglas dentro de los diseños de los sistemas de computadoras desarrollados para los negocios. La incorporación de las reglas dentro de los sistemas significa que su interpretación se diluye en la lógica de los programas y ocurre usualmente que las reglas no son escritas ni por las personas del negocio (que delegan el control a los informáticos), ni por los informáticos (que están más interesados en la administración del software que en la administración de las políticas del negocio). Entonces, las reglas tienden a ser

"implicadas" por las prácticas aceptadas del negocio, y por lo general se pierden los detalles. Se deja al programador adivinar la regla y programar su cumplimiento dentro del sistema. En ocasiones, la tecnología disponible es forzada a manipular la regla, y se hacen compromisos por razones económicas o técnicas, dejando las reglas solo parcialmente implementadas, o dejándolas para que sean implementadas manualmente.

El resultado de las conexiones imprecisas, incompletas e informales entre el negocio y los desarrolladores de software durante el desarrollo del sistema consiste en que el SI a menudo falla en cumplir las necesidades reales del negocio. Aún peor, como las reglas se encuentran colocadas dentro de la estructura del SI y mezcladas con la lógica del control de los programas cualquier manipulación de las mismas es sumamente difícil, costosa y requiere de tiempo; además, solo pueden ser cambiadas por los programadores.

Estas circunstancias tienen como resultado una alta razón de fallo para los proyectos en satisfacer las necesidades reales del negocio, con costos excesivos asociados, y la necesidad de redesarrollar aplicaciones periódicamente para satisfacer las cambiantes necesidades del negocio. Producto de las limitaciones de la tecnología, estos defectos han sido aceptados como parte de los costos de las implementaciones de los sistemas.

Con el enfoque de reglas de negocio la tecnología está avanzando al punto en que estos costos no necesitan ser tolerados, y se devuelve el control a las personas del negocio.

1.2 Características y formas de expresión de las reglas de negocio

En este epígrafe se introduce al tema de reglas de negocio exponiendo lo que son, sus características, los beneficios de usarlas y formas de expresión.

1.2.1 Definir reglas de negocio

Una regla de negocio, básicamente, es una declaración compacta sobre un aspecto del negocio, usando un lenguaje simple, inequívoco, accesible a todas las partes interesadas: el dueño del negocio, el analista, el arquitecto técnico, y así sucesivamente (Morgan, 2002). Este autor enumera características deseables en las declaraciones de reglas. Estas características universales se aplican a cualquier lenguaje o dominio de aplicación.

- *Atómico*: no pueden ser descompuestas sin que se pierda información.
- *No ambiguo*: tienen solamente una obvia interpretación.

- *Compacta*: típicamente son frases cortas.
- *Consistente*: juntas, ellas proporcionan una única y coherente descripción.
- *Compatible*: usan las mismas condiciones en el resto del modelo de negocio.

Algunos ejemplos de reglas de negocio se pueden ver a continuación:

1. *A patient shows any one of the following tests positives: HIV, Hepatitis B & C, and CRI*

⇨ *reject him as a recipient.*

2. *When a patient has done the following test: Blood tests, Diagnostic tests, Psychological and social evaluation and their results are negative*

⇨ *he can be placed in the waiting list.*

1.2.2 Características

En términos generales, las reglas de negocio son normas o limitaciones que definen las condiciones que deben de cumplirse en determinadas situaciones. Las reglas de negocio no son la descripción de un proceso o transformación, sino que ellas definen las condiciones bajo las cuales un proceso es realizado o las nuevas condiciones que surgen después de que este proceso ha sido completado.

En otra forma, las normas definen el qué del caso y no el cómo. Estas reglas definen pre y postcondiciones, ellas pueden actuar como una especificación para un proceso, sin limitar los mecanismos a través de los que las condiciones se transforman en postcondiciones.

Las declaraciones de reglas en el modelo del negocio definen la lógica deseada en el negocio. Estas describen un estado de cosas que la empresa quiere que exista, es decir, lo que el negocio exige.

1.2.3 Beneficios de su uso

Varios son los beneficios que pueden derivarse del uso de Reglas de Negocio. De todos, según Lowenthal (Lowenthal, 2005) los tres más relevantes son los siguientes:

Agilidad: respuesta simple y rápida a los requisitos dinámicos.

Reducción del Costo: bajo costo para crear o actualizar las partes de aplicaciones que implementan las políticas del negocio.

Transparencia: las reglas permiten fácilmente la auditoría que los servicios de software llevan a cabo en sus políticas de negocios correspondientes.

1.2.4 Formas de expresión

En la bibliografía se encuentran formas diversas de expresar las RN según variados puntos de vista. De acuerdo con Morgan (Morgan, 2002) las reglas de negocio pueden expresarse de diversas formas, principalmente, según su depuración a lo largo del sistema o incluso en la manera en que se introduzcan a este. Este trabajo se acoge a estos criterios, donde se definen tres niveles (Martínez et al., 2007a, Martínez et al., 2007b, Moreno Montes de Oca, 2008):

Informal: este nivel proporciona una sentencia en lenguaje natural, sin un rango limitado de parámetro, tal y como el cliente del negocio desee.

Técnico: este nivel combina referencias a datos estructurados, operadores y restricciones con el lenguaje natural, nivel intermedio entre la entrada de la regla y su implementación.

Formal: este nivel proporciona sentencias conforme a una sintaxis definida, más cercana a propiedades matemáticas específicas. Este nivel proporciona la funcionalidad automática de la regla.

La figura 1.2 muestra la relación existente entre las reglas de negocio y las diferentes formas de representación teniendo en cuenta sus potencialidades para expresar la semántica de la fuente de datos y los procesos.

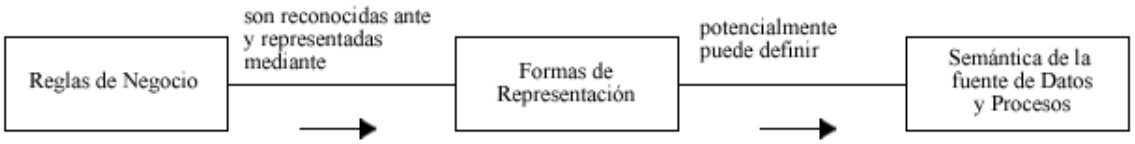

Figura 1:2: Relación entre las Formas de Representación y las Reglas de Negocio

En este trabajo se parte de un repositorio de RN representado a un nivel informal y se obtiene un nuevo repositorio en un nivel técnico, que sirve como entrada a otras herramientas (Pérez Alonso, 2008, Martínez et al., 2007b, Martínez et al., 2007c,

Martínez et al., 2007d), encargadas de realizar procesamientos adicionales, para lograr realizar la generación de la implementación automática de las reglas almacenadas en dicho repositorio.

1.3 Formulación de las reglas de negocio

El proceso de identificación de las reglas de negocio a menudo es un proceso iterativo y heurístico, en el cual las reglas al principio son afirmaciones de política generales. Aún si la política es formal y específica, comúnmente se describe de manera general e informal, y los expertos son los encargados de traducirlas en sentencias específicas y significativas de qué hacer. Sin embargo, estas sentencias más específicas aún se consideran divagaciones del negocio, sin disciplina, sentencias que algunas veces son claras y otras ambiguas, y la mayoría de las veces contienen más de una idea (Halle, 1994). En realidad, estas sentencias pocas veces se originan en una política; por lo general, surgen de las operaciones diarias de la organización (BRG, 2000). Las divagaciones del negocio suelen ser el punto de partida de los analistas para derivar sentencias de reglas de negocio más formales.

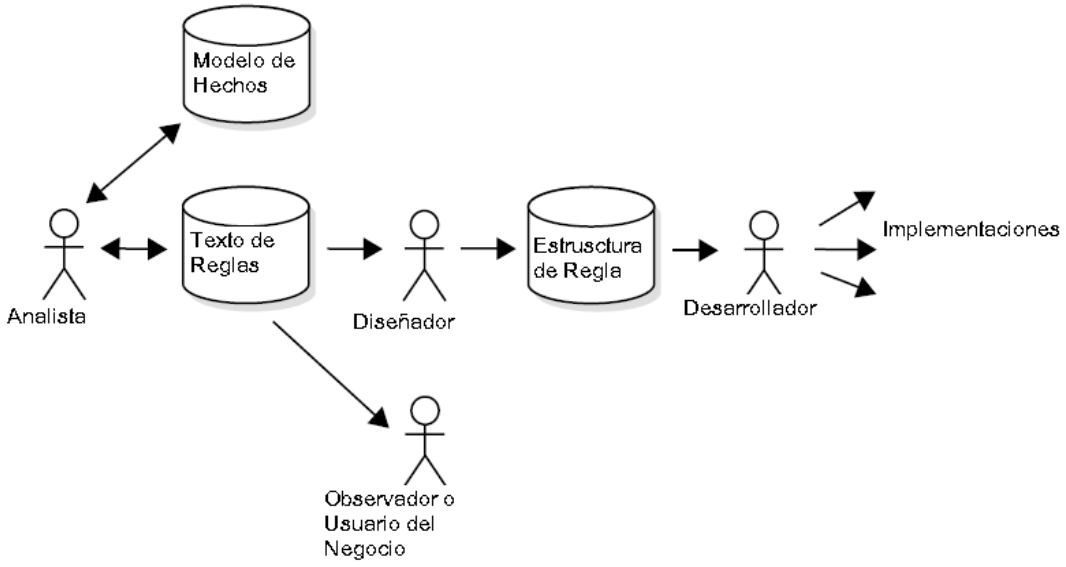

Figura 1.2: Definición de Reglas de Negocio (Morgan, 2002)

Inicialmente, la tarea del analista es descomponer estas divagaciones compuestas hacia reglas de negocio atómicas, que son sentencias específicas y formales de un solo término, hecho, derivación o restricción en el negocio. Entre otras cosas, el analista debe evaluar la estabilidad de la regla como aspecto fundamental del negocio (BRG, 2000). Los aspectos fundamentales pueden ser vistos en la infraestructura de la compañía, mientras que las reglas de negocio más transitorias comúnmente se manifiestan en la práctica del trabajo, como se puede ver en la figura 1.2.

Seguidamente, la tarea del diseñador es identificar las sentencias atómicas como la definición de un término, hecho, restricción o derivación. Los términos, hechos y algunas restricciones pueden ser representados directamente en modelos gráficos. Las restricciones restantes y las derivaciones deben ser traducidas en algún formalismo. Esto puede ser tan simple como oraciones en lenguaje natural o puede ser una expresión más formal, como una especificación en lenguaje lógico o una notación gráfica como la propuesta por Ronald Ross (Ross, 1997). Independientemente de la forma, el diseñador finalmente identificará la tecnología apropiada e implementará las reglas de negocio en el SI.

1.4 Modelo de Hechos como principio básico del enfoque de reglas de negocio

Según Ross (Ross, 2003) el Modelo de Hechos está conformado por los conceptos básicos y sus conexiones. Cada concepto tiene una definición clara y precisa desde la perspectiva del negocio y debe ser reflejada en el Glosario de Términos que es una colección de todos los términos y sus definiciones. Esta es la base de un principio importante del enfoque de reglas de negocio, se trabaja mejor si todos hablan el mismo lenguaje.

Para incluir un término en el Modelo de Hechos, se deben satisfacer las condiciones siguientes:

- Básico: no se puede derivar de otros términos.
- Atómico: es indivisible.
- Conocible: representa cosas que existen, conocimiento acerca del negocio.

Cada conexión lógica o factual entre conceptos tiene una forma estándar; estas conexiones son conocidas como hechos. Según Morgan (Morgan, 2002), un hecho es "una interrelación entre términos identificables en el Modelo de Hechos. Esta interrelación puede estar limitada por otros elementos descriptivos en orden de especificar la aplicabilidad de la regla con precisión".

En general, los hechos representan conocimiento esencial del negocio, relacionan los términos y se expresan mediante sentencias. Los hechos se basan en lenguaje común y extienden el vocabulario del negocio. Todo hecho sigue una rigurosa estructura sujeto-verbo-objeto. En la figura 1.3 se muestra una representación gráfica abstracta de un hecho en el Modelo de Hechos.

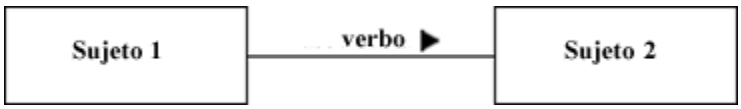

Figura 1.3: Representación gráfica de un hecho

A través del Modelo de Hechos, las reglas de negocio hacen referencia a otros elementos del modelo, principalmente a objetos del negocio y sus atributos (Morgan, 2002). El Modelo de Hechos trata las relaciones entre conceptos; además, proporciona un esquema para organizar los demás componentes. En un enfoque de reglas de negocio, la estructura de un sistema se visualiza por medio del Modelo de Hechos, con sus componentes, los términos y los hechos.

El conocimiento del negocio se expresa usando palabras y frases que tienen sentido para las personas del negocio (Ross, 2003). Los hechos se construyen sobre los términos, conectándolos de forma tal que se refleja el "mundo real" del negocio que se modela. Las reglas usan los hechos para ayudar en el control de las operaciones del negocio y así lograr que el negocio se desarrolle tal y como las personas del negocio desean que el mismo sea desarrollado. Los términos son la base sobre la cual se construyen los hechos. Las Reglas del Negocio usan los hechos, son construidas sobre estos (véase la figura 1.4), y guían las operaciones del negocio interactuando entre sí (Chappel, 2005).

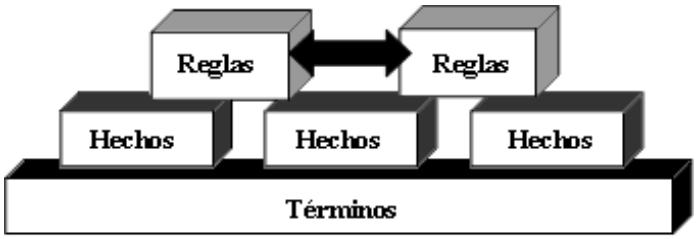

Figura 1.4: Pirámide de Términos, Hechos y Reglas

Para el buen estado de las reglas de negocio es esencial tener un vocabulario bien definido y consistente del negocio. Un Modelo de Hechos estructura el conocimiento básico acerca de las operaciones del negocio desde la perspectiva del mismo. En particular, un Modelo de Hechos se enfoca en la estandarización de la terminología del negocio para establecer un vocabulario de negocio común; por lo que es considerado un punto de partida crucial en su modelado (Ross, 2000).

1.4.1 Modelo genérico de hechos creado para un caso de estudio

En (Moreno Montes de Oca, 2008, Martínez et al., 2007) se selecciona un modelo genérico el cual da una definición para la estructura del Modelo de Hechos y además facilita de forma natural su representación lógica con ayuda de una ontología, este estudio unido a la tesis de (Kafle, 2009) sirve de base para este trabajo, el cual permite a los usuarios del negocio capturar las reglas relacionadas con la esfera médica en un lenguaje usado por los expertos del negocio, y los almacene en un formato que facilite su validación y comprobación de la consistencia.

Como se ha dicho en epígrafes anteriores, el vocabulario, denominado Modelo de Hechos, se debe establecer en la etapa inicial del ciclo de vida de las reglas. En trabajos anteriores (Moreno Montes de Oca, 2008, Martínez et al., 2007) se crea un modelo genérico de hechos que ayuda en la modelación de las relaciones entre los conceptos de un dominio concreto haciendo uso de una ontología, esta se implementa seleccionando una estructura genérica que representa un Modelo de Hechos y se instancia con el

vocabulario perteneciente al domino específico del Trasplante Renal. La estructura creada, a su vez, facilita de forma natural su representación lógica con ayuda de una ontología, que sigue la metodología de Noy y McGuinness de la Universidad de Stanford (Noy and McGuinness, 2005) se generó en formato XML para su integración a una herramienta de propósito más general.

El vocabulario relacionado con el trasplante renal ha sido representado por medio de una ontología en Protégé que se integra a la ontología Drools para facilitar el proceso de autoría de la regla en el ámbito del problema (Kafle, 2009).

Protégé implementa un conjunto de estructuras para modelar el conocimiento y acciones que apoyan la creación, visualización, y manipulación de ontologías en varios formatos de representación. El Protégé puede ser personalizado para proporcionar apoyo a un dominio amistoso para la creación de modelos de conocimiento y entrada de datos.

Además, Protégé se puede ampliar por medio de un plug-in y una arquitectura basada en Java Application Programming Interface (API) para la construcción de herramientas y aplicaciones basadas en el conocimiento.

Esta herramienta de adquisición del conocimiento se diseña para ser específica del dominio, permitiendo a expertos del dominio incorporar fácil y naturalmente su conocimiento del área.

Actualmente se está utilizando en medicina clínica, y ciencias biomédicas, aunque puede ser utilizado en cualquier campo donde los conceptos se pueden modelar como jerarquía de clases.

1.5 Propuestas de patrones para las RN

El reconocimiento de patrones, también llamado lectura de patrones, no sólo es un campo de la informática sino un proceso fundamental que se encuentra en casi todas las acciones humanas. El punto esencial del reconocimiento de patrones es la clasificación, indicando mediante señales las características de cada patrón.

Un sistema de reconocimiento de patrones completo consiste en un sensor que recoge las observaciones a clasificar, un sistema de extracción de características que transforma la información observada en valores numéricos o simbólicos, y un sistema de clasificación o descripción que, basado en las características extraídas, clasifica la medición.

La clasificación utiliza habitualmente uno de las siguientes procedimientos: clasificación estadística (o teoría de la decisión), clasificación sintáctica (o estructural). El reconocimiento estadístico de patrones está basado en las características estadísticas de los patrones, asumiendo que han sido generados por un sistema probabilístico. El reconocimiento estructural de patrones está basado en las relaciones estructurales de las características.

El reconocimiento de patrones es más complejo cuando se usan plantillas para generar variantes. Por ejemplo, en castellano, las frases a menudo siguen el patrón "sujeto-predicado", pero se requiere cierto conocimiento de la lengua para detectar el patrón.

No existe un estándar de cómo hacer Reglas de Negocio, pero si es posible hacer algunas recomendaciones. Los patrones pueden reflejar la forma de las cosas, los hechos que en la industria o los tipos de problemas que un sistema automatizado pretende negociar. (Morgan, 2002).

Patrón 1:

Restricción Básica: Es el más común de los patrones de Reglas de Negocio, establece una restricción sobre un sujeto de una regla.

<determinante> <sujeto> [no] (debe |tiene) <característica>
[(si | a menos que) <hecho>].

< determinante > < sujeto > (puede < característica > solo si <hecho>) |
(no puede < característica >).

Ejemplos:

Every patient with IRCT diagnostic must be valued in Consultation of Nephrology.

A Possible Recipient can be included in the Waiting-list of Transplants only if it was evaluated completely by the Commission of Transplant

Patrón 2:

Lista de Restricción: este patrón también restringe al sujeto, pero restringiendo que son uno o más ítems tomados de una lista.

< determinante > < sujeto > [no] (debe |tiene) < característica >

(si | a menos que) como mínimo <m>
[no mas de <n>] de las siguientes es verdadera:
<lista de hechos>.

< determinante > < sujeto > (puede < característica > solo si) |
(no puede < característica > si)
al menos <m> [y no mas de <n>] de las siguientes es verdadera:
<lista de hechos>.

Ejemplos:

The Group of High Risk must be integrated by Patients who fulfill at least one of following characteristics:
A patient with serología print for hepatitis C or B
A patient with evidences of biochemical activity
A patient with evidence of histological activity
A Potential Donor cannot be a Donor if it presents any of the following Contraindications:
Been VIH,
Hepatitis B,
Hepatitis C

Patrón 3:

Clasificación: este patrón establece una definición para un término en el modelo de hechos. Tales términos son equivalentes a términos de corta vida o de uso temporal. Si esta clasificación es permanente se debe reflejar mejor en el Modelo de Hechos.

< determinante > < sujeto > [no] es definido como <clasificación>
[(si | a menos que) <hecho>].
< determinante > < sujeto > [no] debe ser considerado como < clasificación >
[(si | a menos que) < hecho >]

Ejemplo:

An Operation of Transplant is defined as a Retransplant if the Recipient realized at least an Operation of Transplant

Patrón 4:

Cálculo: este patrón establece una interacción entre términos en el Modelo de Hechos suficientes para realizar los cálculos o el establecer un valor. Note que esto es similar al patrón de clasificación, en cuanto a obtención de valores temporales. Se usa tanto “es definido como” y “debe ser calculado como”

< determinante > <resultado> es definido como <algoritmo>.

< determinante > < resultado > = <algoritmo>.

Ejemplo:

The Hydration of the Recipient in Room of Transplant = Weight of the Recipient in Kg * (30 or 35 cc).

Patrón 5:

Enumeración: este patrón establece el rango de valores que pueden ser legalmente tomados por un término en el Modelo de Hechos.

< determinante > < resultado > debe ser elegido desde la siguiente lista

[abierta | cerrada]: <lista de valores>

Ejemplo:

The Motive of Loss of a Renal Transplant must be chosen of the following open list:

- medical Causes
- surgical Complications
- Sharp precocious rejection

1.6 Consideraciones finales del capítulo

En este capítulo se realiza el estudio del enfoque de RN, como parte de las nuevas investigaciones en el desarrollo de Sistemas de Información, asumiendo el modelo de hechos como principio básico. Se analizan los diferentes patrones como base para el proceso de edición.

Capítulo II: Formato de representación para las reglas de negocio

El uso del Protégé como herramienta para el trabajo con ontologías hace que se disponga de varios formatos para la representación del modelo de hechos y las reglas de negocio. En este capítulo se caracterizan los diferentes formatos, abordando las ventajas del uso del XML. Finalmente se describe la estructura del documento XML, generado por el Protégé, para almacenar el conjunto de reglas.

2.1 Protégé como herramienta para el trabajo con ontologías

Protégé-2000 es una herramienta integrada de software para desarrollar sistemas basados en el conocimiento. Los usos desarrollados con Protégé-2000 se utilizan para la solución de problemas y la toma de decisiones en un dominio particular. Esta herramienta facilita trabajar simultáneamente con las clases y los casos. Tiene una interfaz de usuario amigable, puesto que permite la creación de gráficos, tablas, diagramas, y diferentes componentes de animación para acceder a la base de conocimiento (Moreno Montes de Oca, 2008).

El Protégé se diseña para dirigir a los expertos del dominio en el proceso de desarrollo del sistema y para permitir que estos reutilicen las ontologías del dominio y los métodos problem-solving para el mantenimiento y desarrollo del programa.

Protégé-2000 permite:

- Modelar una ontología de las clases que describen un tema particular.
- Creación de una herramienta de adquisición de conocimiento para recoger conocimiento.
- Entrar en casos específicos de datos y de la creación de una base de conocimiento.
- La ejecución de usos.

2.2 Protégé, formatos a los que exporta una Ontología.

Las ontologías creadas en Protégé pueden ser exportadas a varios formatos como son RDF (acrónimo de Resource Description Framework), OWL (acrónimo de Ontology Web Language) y XML (acrónimo de Extensible Mark-up Language). En las versiones más recientes se tiene la posibilidad de editar clases y sus características, acceder a motores de razonamiento, editar y

ejecutar consultas y reglas, comparar ontologías, visualizar relaciones entre conceptos y obtener instancias usando procesamientos configurables por el usuario.

2.2.1 Formato RDF

El origen de RDF se debe a Ramanathan V. Guha cuando trabajaba en Apple Computer en su forma inicial conocida como MCF, más tarde continuaba durante su etapa en Netscape Communications Corporation. Este modelo se basa en la idea de convertir las declaraciones de los recursos en expresiones con la forma sujeto-predicado-objeto (conocidas en términos RDF como tripletas). El sujeto es el recurso, es decir aquello que se está describiendo. El predicado es la propiedad o relación que se desea establecer acerca del recurso. Por último, el objeto es el valor de la propiedad o el otro recurso con el que se establece la relación. La combinación de RDF con otras herramientas como RDF Schema y OWL permite añadir significado a las páginas, y es una de las tecnologías esenciales de la Web semántica.

RDF - la Infraestructura para la descripción de Recursos - es un estándar para realizar descripciones sencillas. XML es a la sintaxis, lo que RDF a la semántica - un conjunto claro de reglas para proporcionar información descriptiva sencilla. El Esquema RDF entonces proporciona un modo de combinar esas descripciones en un vocabulario único. RDF se integra en una variedad de aplicaciones incluyendo:

- Catálogos de biblioteca.
- Directorios mundiales.
- Sindicación y agregación de noticias, software y contenido.
- Colecciones personales de música, fotos y eventos.

En estos casos, cada uno utiliza XML como sintaxis de intercambio. Las especificaciones RDF proporcionan una infraestructura potente para el intercambio de conocimiento en la Web.

"RDF es parte de la base de un avance significativo en el potencial de la Web. Finalmente, veremos a aplicaciones y usuarios combinar la información representada en RDF de múltiples fuentes en la Web, en modos que, hasta ahora, han sido inconcebibles," explica Brian McBride, Presidente del Grupo de Trabajo RDF Esencial, "El Grupo de Trabajo RDF Esencial ha convertido las especificaciones RDF en una base práctica y matemáticamente precisa sobre la que se puede construir OWL y el resto de la Web Semántica."

2.2.2 Formato RDF Schema.

El formato RDFS o RDF Schema o Esquema RDF es una extensión semántica de RDF. Un lenguaje primitivo de ontologías que proporciona los elementos básicos para la descripción de vocabularios. La primera versión fue publicada en abril de 1998 por la W3C, la versión actual de la recomendación fue publicada en febrero de 2004 también por la W3C. Existen actualmente otros lenguajes de ontologías más potentes, como puede ser OWL.

Características principales:

- Un archivo RDFS es un archivo RDF, es decir, se trata de un archivo con la misma sintaxis y la misma estructura que la que se usa en RDF. La sintaxis está basada en XML.
- Es extensible, cada desarrollador puede extender el esquema RDF de manera independiente.

2.2.3 Formato OWL

El formato OWL ofrece posibilidad de replantación de Ontologías que funcionan en la Web. Se necesita además un modo de desarrollar vocabularios para un asunto o un dominio específico. Mediante la ontología se definen los términos utilizados para describir y representar un área de conocimiento. Son utilizadas por personas, bases de datos o por aplicaciones que necesitan compartir información específica sobre un determinado asunto o dominio; por ejemplo en la medicina, fabricación de herramientas, inmobiliarias, reparación de automóviles, administración financiera, etc. Las Ontologías incluyen definiciones utilizables por máquinas, sobre conceptos básicos del dominio y de las relaciones existentes entre los mismos. Codifican conocimiento en un dominio y también conocimiento que se expande a través de varios dominios. De este modo, hacen que este conocimiento sea reutilizable.

OWL, considerado el Lenguaje de Ontologías Web, proporciona un lenguaje para la definición de Ontologías estructuradas, basadas en la Web, que ofrece una integración e interoperabilidad de datos más rica entre comunidades descriptivas. Los lenguajes anteriores se utilizaron para desarrollar herramientas y ontologías para comunidades de usuarios específicas (particularmente en las ciencias y en aplicaciones de comercio electrónico de compañías específicas), pero no fueron definidos para ser compatibles con la arquitectura de la World Wide Web en general, y de la Web Semántica en particular.

Este formato utiliza URIs para fijar nombres y la infraestructura para descripciones en la Web proporcionada por RDF para agregar las siguientes capacidades a las ontologías:

- Habilidad de ser distribuida por muchos sistemas.
- Escalabilidad a las necesidades de la Web.
- Compatibilidad con estándares Web para la accesibilidad y la internacionalización.
- Apertura y extensibilidad.

Dicho formato se construye sobre RDF y RDF Esquema, añadiendo vocabulario para la descripción de clases y propiedades.

El formato OWL supone un gran paso adelante en la representación y organización de conocimiento en la World Wide Web. Se posiciona como ganador entre las necesidades de la industria de un lenguaje que dirija sus casos de uso en las Web actuales, y las restricciones de desarrollar un lenguaje de ontologías que sirva de engranaje a principios científicos establecidos y experiencia de investigación, como explicaron Jim Hendler y Guus Schreiber, co_presidentes del Grupo de Trabajo de Ontología Web. Más de cincuenta participantes en este grupo han diseñado con éxito un lenguaje que contempla ambos conjuntos de preocupaciones y que es respaldado de la misma forma por académicos y practicantes.

2.2.4 Formato XML

En el desarrollo del siguiente subepígrafe, se dará una introducción al formato XML, presentando su historia y principales características que lo hacen tan utilizado.

2.2.4.1 Historia

El formato XML proviene de un lenguaje creado por IBM en los años setenta, llamado GML (Generalized Markup Language), que surgió por la necesidad que tenía la empresa de almacenar gran cantidad de información. Este lenguaje le agradó a la ISO, por lo que en 1986 trabajaron para normalizarlo, creando SGML (Standard Generalized Markup Language), capaz de adaptarse a un gran abanico de problemas. A partir de él se han creado otros sistemas para almacenar información.

En el año 1989 Tim Berners Lee creó la web, y junto con ella el lenguaje HTML. Este lenguaje se definió en el marco de SGML y se hacen importantes aplicación de este estándar. Los navegadores web sin embargo siempre han puesto pocas exigencias al código HTML que

interpretan y así las páginas web son caóticas y no cumplen con la sintaxis. Estas páginas web dependen fuertemente de una forma específica de lidiar con los errores y las ambigüedades, lo que hace a las páginas más frágiles y a los navegadores más complejos.

Otra limitación de SGML es que cada documento pertenece a un vocabulario fijo, establecido por el DTD (Document Type Declaration), no se pueden combinar elementos de diferentes vocabularios. Asimismo es imposible para un intérprete (por ejemplo un navegador) analizar el documento sin tener conocimiento de su gramática (del DTD). Por ejemplo, el navegador sabe que antes de una etiqueta debe haberse cerrado cualquier <p> previamente abierto. Los navegadores resolvieron esto incluyendo lógica ad hoc para el HTML, en vez de incluir un analizador genérico. Ambas opciones, de todos modos, son muy complejas, se buscó entonces definir un subconjunto del SGML que permita:

- Mezclar elementos de diferentes lenguajes. Es decir que los lenguajes sean extensibles.
- La creación de analizadores simples, sin ninguna lógica especial para cada lenguaje.
- Empezar de cero y hacer hincapié en que no se acepte nunca un documento con errores de sintaxis.

Surge así el formato XML, dejando a un lado muchas de las características, ya obsoletas de SGML, que estaban pensadas para facilitar la escritura manual de documentos.

2.2.4.2 Características

El formato XML, a diferencia de su antecesor, está orientado a hacer las cosas más sencillas para los programas automáticos que necesiten interpretar el documento y está caracterizado por:

- Proporcionar en su base un conjunto de reglas para la creación de vocabularios que dota de estructura a los datos y documentos a la Web.
- Proporcionar reglas claras para la sintaxis.
- Sus esquemas sirven como un método de composición de vocabularios Web.
- Ofrece una sintaxis base para documentos potentes y flexibles, sin imponer restricciones semánticas al significado de esos documentos.

2.3 Selección del formato XML para el trabajo.

Al ser el formato XML una tecnología sencilla tiene a su alrededor otras que la complementan y la hacen mucho más amplia y con unas posibilidades mayores. Tiene el importante papel de

permitir la compatibilidad entre sistemas posibilitando compartir la información de una manera segura, fiable y fácil. Esto hace que sea el formato seleccionado en este trabajo para almacenar el repositorio de RN.

2.3.1 Ventajas del XML

En este epígrafe se muestran ventajas del trabajo utilizado el formato XML, son las siguientes:

- Es extensible: Después de diseñado y puesto en producción, es posible extender XML con la adición de nuevas etiquetas, de modo que se pueda continuar utilizando sin complicación alguna.
- El analizador es un componente estándar, no es necesario crear un analizador específico para cada versión de lenguaje XML. Esto posibilita el empleo de cualquiera de los analizadores disponibles. De esta manera se evitan los errores y se acelera el desarrollo de aplicaciones.
- Si un tercero decide usar un documento creado en XML, es sencillo entender su estructura y procesarla. Mejora la compatibilidad entre aplicaciones.

2.4 Documento XML utilizado como repositorio de RN

A continuación se describe la estructura del repositorio de RN una vez que se ha exportado desde el Protégé al formato XML.

La tecnología XML busca dar solución al problema de expresar información estructurada de la manera más abstracta y reutilizable posible; esto significa que se compone de partes bien definidas y que esas partes, a su vez, se componen de otras. Así se obtiene un árbol de fragmentos de información. Estas partes son nombradas “elementos” y se les señala mediante etiquetas.

Una etiqueta consiste en una marca hecha en el documento que indica una porción de éste como un elemento, o sea, una fracción de información con un sentido claro y definido. Las etiquetas tienen la forma <nombre>, donde “nombre” indica el elemento que se está señalando.

A continuación se muestra la estructura del documento XML generado por el Protégé, este documento es utilizado por la herramienta de edición en la obtención de las clases, sus atributos y las reglas, este es considerado un repositorio temporal de las RN.

El documento XML comienza con unas líneas que describen la versión y la dirección en internet de la documentación, llamado prólogo.

```
<?xml version="1.0" ?>

<knowledge_base
      xmlns="http://protégé.stanford.edu/xml"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://protégé.stanford.edu/xml
http://protégé.stanford.edu/xml/schema/protégé.xsd">
```

Seguidamente aparecen las clases auxiliares:

```
      <class>
            <name>:nombre de la clase</name>
            <type>:tipo de la clase</type>
            <own_slot_value>
                  <slot_reference>:ROLE</slot_reference>
                  <value value_type="string">Abstract</value>
            </own_slot_value>
            <superclass>:CONSTRAINT</superclass>
            <template_slot>:PAL-NAME</template_slot>
                  *
                  *
                  *
            <template_slot>:PAL-NAME</template_slot>
      </class>
```

A partir de aquí es donde se muestran las clases de la ontología creada en el Protégé, siendo KidneyTransplantOntology el nombre de esta ontología específica.

```
      <class>
            <name>KidneyTransplantOntology</name>
            <type>:STANDARD-CLASS</type>
            <own_slot_value>
                  <slot_reference>:ROLE</slot_reference>
                  <value value_type="string">Abstract</value>
            </own_slot_value>
            <superclass>:THING</superclass>
      </class>
```

A continuación se muestra un ejemplo de una clase con todos sus atributos, en este caso Patient es el nombre de la clase y está conformada por el conjunto de sus atributos: testp, sex, age, name, admission_dateP, id_card y blood_group.

```
      <class>
            <name>Patient</name>
            <type>:STANDARD-CLASS</type>
            <own_slot_value>
                  <slot_reference>:ROLE</slot_reference>
```

```
            <value value_type="string">Abstract</value>
      </own_slot_value>
      <own_slot_value>
            <slot_reference>:DOCUMENTATION</slot_reference>
            <value value_type="string">Information about the
                  Patient of the Kidney</value>
      </own_slot_value>
      <superclass>KidneyTransplantOntology</superclass>
      <template_slot>testP</template_slot>
      <template_slot>sex</template_slot>
      <template_slot>age</template_slot>
      <template_slot>name</template_slot>
      <template_slot>admission_dateP</template_slot>
      <template_slot>id_card</template_slot>
      <template_slot>blood_group</template_slot>
</class>
```

A continuación aparecen las clases del paquete Drools, módulo que se le agrega al Protégé para el trabajo con las RN.

```
<class>
      <name>Drools Ontology</name>
      <type>:STANDARD-CLASS</type>
      <own_slot_value>
            <slot_reference>:ROLE</slot_reference>
            <value value_type="string">Abstract</value>
      </own_slot_value>
      <superclass>:THING</superclass>
      <template_slot>title</template_slot>
</class>
```

Luego aparece la etiqueta <slot> que no es más que los atributos de las clases de la ontología, acompañado del tipo de dato. Por ejemplo:

```
<slot>
      <name>admission_dateP</name>
      <type>:STANDARD-SLOT</type>
      <own_slot_value>
            <slot_reference>:SLOT-MAXIMUM-CARDINALITY
              </slot_reference>
            <value value_type="integer">1</value>
      </own_slot_value>
      <own_slot_value>
            <slot_reference>:SLOT-VALUE-TYPE</slot_reference>
            <value value_type="string">String</value>
      </own_slot_value>
</slot>
```

Por último tenemos la etiqueta <simple_instance>, este es el que contiene las reglas de negocio.

```
<simple_instance>
      <name>Rule_Editorv4_Instance_10</name>
      <type>Rule</type>
      <own_slot_value>
            <slot_reference>dialect</slot_reference>
            <value value_type="string">java</value>
      </own_slot_value>
      <own_slot_value>
            <slot_reference>then</slot_reference>
            <value value_type="string">Accept the patient as
                  potential donor for kidney transplantation
              </value>
      </own_slot_value>
      <own_slot_value>
            <slot_reference>when</slot_reference>
            <value value_type="string">A patient is between
                  18 and 55 yrs of old
              </value>
      </own_slot_value>
      <own_slot_value>
            <slot_reference>title</slot_reference>
            <value value_type="string">
                  R04:  Donor&apos;s Age
              </value>
      </own_slot_value>
</simple_instance>
```

2.5 Consideraciones finales del capítulo

En este capítulo:

- Se estudian los diferentes formatos con los que trabaja el Protégé.
- Se analizan razones que hacen que el formato XML sea considerado como el formato válido para este trabajo.
- Se describe la estructura del documento XML generado a partir del Protégé para guardar un conjunto de reglas.

Capítulo III Editor de Reglas de Negocio

En este capítulo se aplican los resultados teóricos de la investigación exponiendo el desarrollo de la herramienta de ayuda a la edición de reglas y el papel que ocupa dentro de la arquitectura del editor de reglas de negocio, detallándose el formato del documento XML utilizado como repositorio, resultante de la herramienta. Se analizan además validaciones sintácticas y semánticas que se le realizan a las reglas editadas.

3.1 Formas modernas de implementación de Reglas de Negocio

Dentro del contexto de las investigaciones de RN, estas representan un meta-elemento importante que necesita ser capturado y formalizado separadamente de otros elementos, conformando así la arquitectura de la aplicación. Actualmente es posible el desarrollo de importantes herramientas, basadas en tecnologías orientadas a RN. Teniendo en cuenta como ellas representan, implementan y ejecutan las RN, son clasificadas como sigue (Bajec et al., 2000, Michael Barnes, 1997):

- Herramientas independientes de la base de datos: las RN son implementadas fuera de la base de datos, usando disparadores y procedimientos almacenados. Sin embargo, son generadas y administradas automáticamente por herramientas desarrolladas, no por herramientas de la base de datos.
- Herramientas basadas en servicios: las RN que son creadas por herramientas desarrolladas que se encuentran en la capa media de servicios de aplicaciones y residen en el servidor de aplicaciones.
- Sistemas basados en reglas: intentan especificar restricciones sobre determinados elementos de datos o tablas, una lógica orientada a obtener la lógica de negocio al más alto nivel y las reglas asociadas a diferentes situaciones. En tiempo de ejecución es usada una máquina especial para procesar las reglas y generar las respuestas apropiadas.

Esta investigación se ajusta a las herramientas independientes de la base de datos como forma de implementación de las RN.

3.2 Editor de RN

El editor de reglas, permite a los expertos del negocio que carecen de formación técnica, manejar y construir las reglas que definen su negocio simplemente utilizando el ratón de su ordenador y una interfaz basada en "seudo siglas". Estas reglas se aplican después de manera dinámica a la aplicación en ejecución.

3.2.1 Construcción de Reglas

La pantalla de construcción de reglas permite el diseño y la gestión de las mismas con una interfaz sencilla, de modo que simplemente con el ratón se pueden crear todas las reglas que la organización necesite. En esta misma pantalla, las secciones y campos seleccionados durante la fase de "Identificación de datos" aparecerán disponibles para ser utilizadas por el usuario junto a:

- Operadores (AND, OR, 'is greater than', 'multiplied by' etc.)
- Funciones de gestión de reglas ('New Rule', 'Clear Rule' etc.)
- Vínculos a datos externos.
- Reglas comunes utilizando funciones Java.

Las reglas se construyen simplemente seleccionando los operadores y los campos que se quieran incluir, y generando la sintaxis básica de la reglas en el editor. Mediante esta interfaz el usuario puede habilitar, deshabilitar y borrar las reglas existentes, como también verificar y desplegar las actualizaciones en tiempo real.

3.2.2 Arquitectura del Editor de RN

El editor de reglas está enmarcado dentro de un ambiente donde interactúan varias herramientas. Esta investigación abarca el analizador sintáctico y semántico, actualizando el modelo de hechos y el conjunto de reglas. Dando como resultado final un documento XML, que sirve de entrada a herramientas que generan automáticamente las nuevas reglas o aquellas que han sido modificadas, véase la figura 3.1.

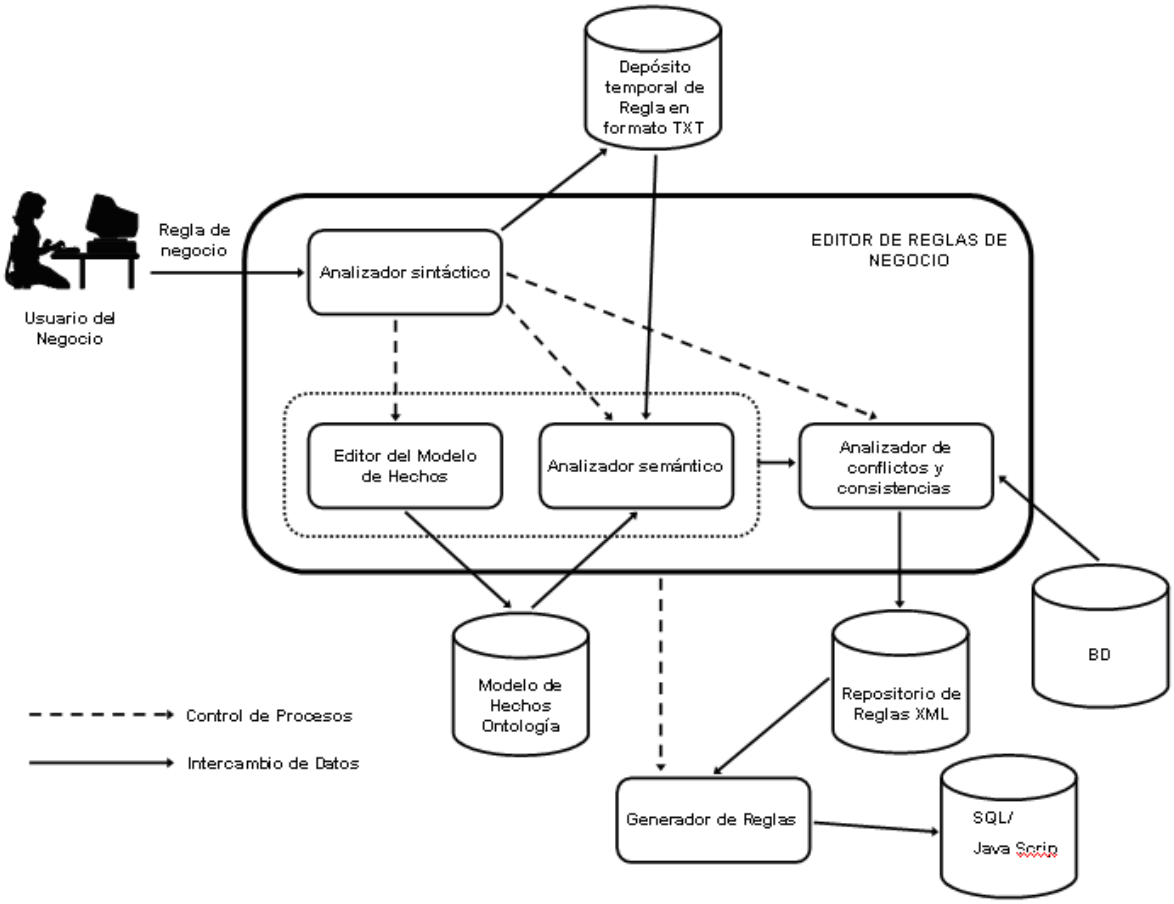

Figura 3.1 Ambiente de desarrollo del Editor de RN

3.2.3 Patrón de reglas procesado por la herramienta

La herramienta de edición de RN se apoya en la estructura ofrecida por Morgan (Morgan, 2002), de ellos se escoge el Patrón 1:que es un patrón sobre Restricción Básica, por la razón de ser el más común de todos y donde se establecen las restricciones sobre un determinado sujeto. Este Patrón 1: se mostró en el Capítulo 1 del presente documento. En etapas iniciales de otros trabajos que abordan el tema de edición y generación automática de reglas (Pérez Alonso, 2008, Martínez et al., 2007) también se trabaja sobre este Patrón.

3.3 Diseño e implementación de la herramienta de Software

Se diseña e implementa una herramienta que permite la edición de RN en lenguaje cercano al natural, permitiendo realizar validaciones sintácticas y semánticas.

3.3.1 Diagrama de Clases

En la figura 3.2 se puede observar un diagrama de clases, luego se da una breve explicación de algunas de las clases más importantes y la relación entre ellas.

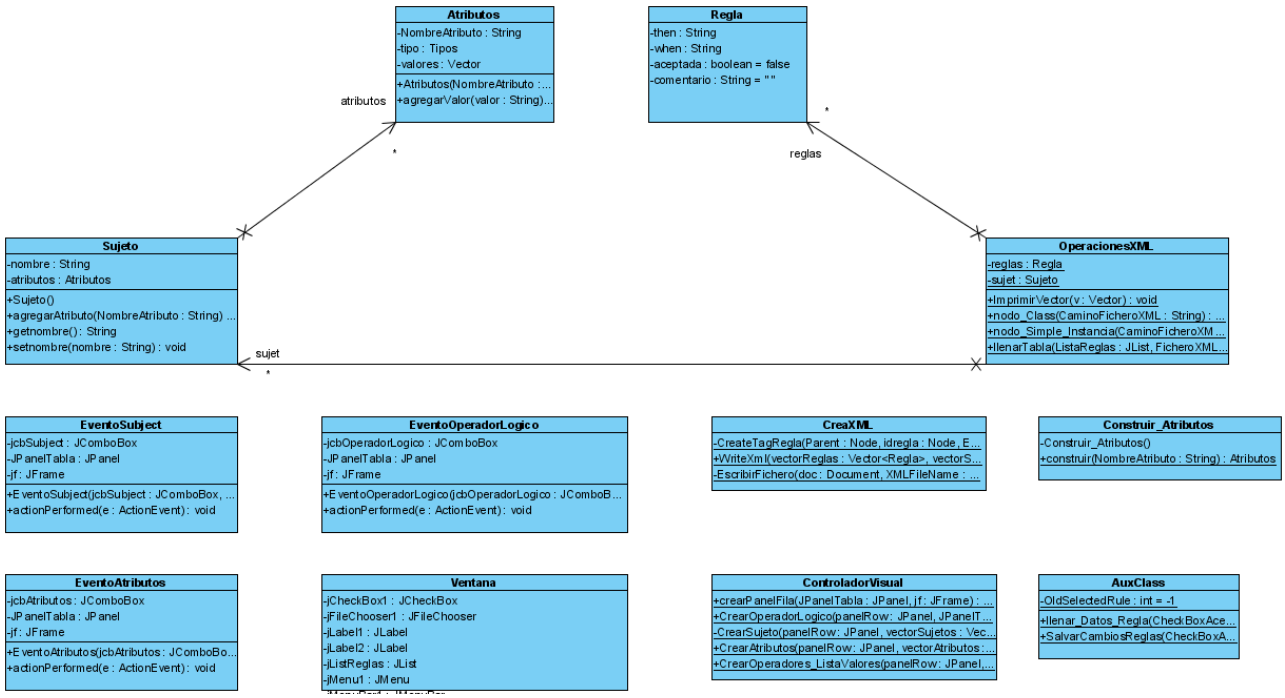

Figura 3.2 Diagrama de clases

3.3.2 Descripción de las principales clases y sus métodos

Primeramente se implementan las clase atributo y sujeto, que tiene como responsabilidad manejar la información tanto de atributos como sujeto para poder conformar las reglas.

Sujeto, compuesto por su nombre y un vector de atributos que a su vez tiene su nombre, tipo y valores. De acuerdo al tipo de atributo es que se le asignan valores o no. Por ejemplo, el tipo CONJUNTO definido en esta clase, tipo que se crea con el objetivo de tener valores asociados a un atributo, da la posibilidad, de definir los posibles valores que puede tomar un atributo, realizándose así una validación semántica de estos valores. Por

ejemplo: si se tiene el atributo “enfermedad” se pueden especificar los nombres de las distintas enfermedades que podría tener un paciente.

```
package javaapplication1;

public class Atributos implements Serializable {
```

La clase Atributos tiene tres atributos privados, correspondientes al nombre, tipo y vector para los posibles tipos.

```
private String NombreAtributo;

private Tipos tipo;

private Vector valores;
```

Consta además de varios métodos públicos que permiten el acceso a los atributos de la clase.

```
public Atributos(String NombreAtributo, Tipos tipo, Vector valores) {

public Tipos getTipo()

public Vector getValores()

public String getNombreAtributo()

public void agregarValor(String valor)

}
```

A continuación se describe la clase Sujeto. Esta se conforma de dos atributos privados que guardan el nombre del sujeto y sus atributos.

```
public class Sujeto implements Serializable{

private String nombre;

private Vector<Atributos> atributos;
```

El siguiente constructor inicializa con vacío atributos de la clase Sujeto. Se muestran además los diferentes métodos de esta clase.

```
public Sujeto() {

        atributos = new Vector<Atributos>();
```

```
}
public void agregarAtributo(String NombreAtributo)
public void insertarAtributo(Atributos a)
public void eliminarAtributo(int index)
public String getnombre()
public Vector<Atributos> getAtributos()
public Atributos getAtributos(int i)
public void setnombre(String nombre)
```

La clase Regla tiene como responsabilidad guardar información acerca de las reglas, comentarios, sujeto, característica, hecho y si es activa la regla o no. Los atributos de esta clase se describen seguidamente.

```
public class Regla implements Serializable{
private  String then;
private  String when;
private  boolean aceptada=false;
private String comentario=" ";
private String sujeto;
private String caracteristica;
private String hecho;
}
```

Los métodos que proporciona esta clase para el acceso a su información son descritos a continuación.

```
public boolean isAceptada()
public void setAceptada(boolean aceptada)
public String getComentario()
public void setComentario(String comentario)
```

```
public String getSujeto()
public void setSujeto(String sujeto)
public String getCaracteristica()
public void setCaracteristica(String caracteristica)
public String getHecho()
public void setHecho(String hecho)
public String getThen()
public void setThen(String then)
public String getWhen()
public void setWhen(String when)
```

La clase OperacionesXml juega un importante papel dentro de este diseño. En ella se implementa todo lo relacionado con la lectura y escritura de ficheros XML, utilizados para el intercambio de la información con otras herramientas de procesamiento de RN. Los atributos de esta clase son un vector de reglas y otro de sujeto.

```
public class OperacionesXML {
private static Vector<Regla> reglas;
private static Vector<Sujeto> sujet;
```

Estos son los métodos que permiten acceder a sus atributos: vector de reglas y vector de sujeto.

```
public static Vector<Regla> getReglas()
public static Vector<Sujeto> getSujet()
```

Se definen otros métodos para el trabajo con esta clase. El método descrito seguidamente tiene la función de recorrer el repositorio de reglas generado por el Protégé en XML y obtener todos los nombres de las clases que son considerados en esta aplicación los sujetos y sus atributos.

```
public static Vector<Sujeto> nodo_Class(String CaminoFicheroXML)
```

El método siguiente, al igual que el anteriormente descrito, recorre el repositorio en documento XML generado por el Protégé, obteniendo todas las reglas de negocio antes editadas.

```
public static Vector<Regla> nodo_Simple_Instancia(String CaminoFicheroXML)
```

La siguiente clase descrita tiene como responsabilidad todo lo referente a la parte visual en el editor de la aplicación, seguidamente se mostrarán algunos de sus métodos.

```
public class ControladorVisual {

    public static void CrearAtributos(JPanel JPanelTabla,
                                      ComboBox    jComboBoxSujeto, JFrame jf)

    public static void llenarCombobox(JComboBox comboboxSujeto)

    public static void CrearOperadores_ListaValores(JPanel panelRow, Atributos a)

    public static void CrearOperadorLogico(JPanel panelRow, JPanel JPanelTabla,
                                           JFrame jf, JComboBox jComboBoxSujeto)

        }
```

Por último se describe la clase CreaXml, encargada de generar un documento XML, repositorio de reglas que será utilizado por el generador automático de reglas. El método CreateTagRegla crea el documento XML con la estructura específica para que pueda ser utilizada por otras herramientas.

```
public class CreaXML {

        private static  void CreateTagRegla(

                Node Parent,

                Node idregla,

                Node Estado,

                Node Sujeto,

                Node Caracteristica, Node Hecho,
```

```
                    Node Ultima_Fecha,
                    Node LenguajeNatural,
                    Node Comentario) {
                        Parent.appendChild(idregla);
                        Parent.appendChild(Estado);
                        Parent.appendChild(Sujeto);
                        Parent.appendChild(Caracteristica);
                        Parent.appendChild(Hecho);
                        Parent.appendChild(Ultima_Fecha);
                        Parent.appendChild(LenguajeNatural);
                        Parent.appendChild(Comentario);
                    }
        }
```

El siguiente método llena la estructura ya creada con los valores correspondientes.

```
public static void WriteXml(Vector<Regla> vectorReglas,
    Vector<Sujeto>   vectorSujeto, String XmlFileName)
```

Finalmente este método permite crear el documento XML en el camino señalado por el usuario.

```
private static void EscribirFichero(Document doc, String XMLFileName)
```

3.3.3 Casos de uso.

Los modelos de casos de uso proporcionan un medio sistemático e intuitivo de capturar requisitos funcionales del sistema basándose en los requerimientos de los usuarios. Ellos dirigen todo el proceso de desarrollo de un software ya que constituyen el punto de partida para llevar a cabo la mayoría de las actividades: el análisis, diseño y prueba del software (Jacobson et al., 2000). Este modelo se realiza identificando cada actor del sistema como los posibles usuarios para los cuales está realizado el mismo.

Esta herramienta está destinada a un solo tipo de actor que es el especialista o experto en el negocio, el cual tendrá conocimiento de cada caso de uso y esta nombrado en el diagrama de la figura 3.3 como Usuario.

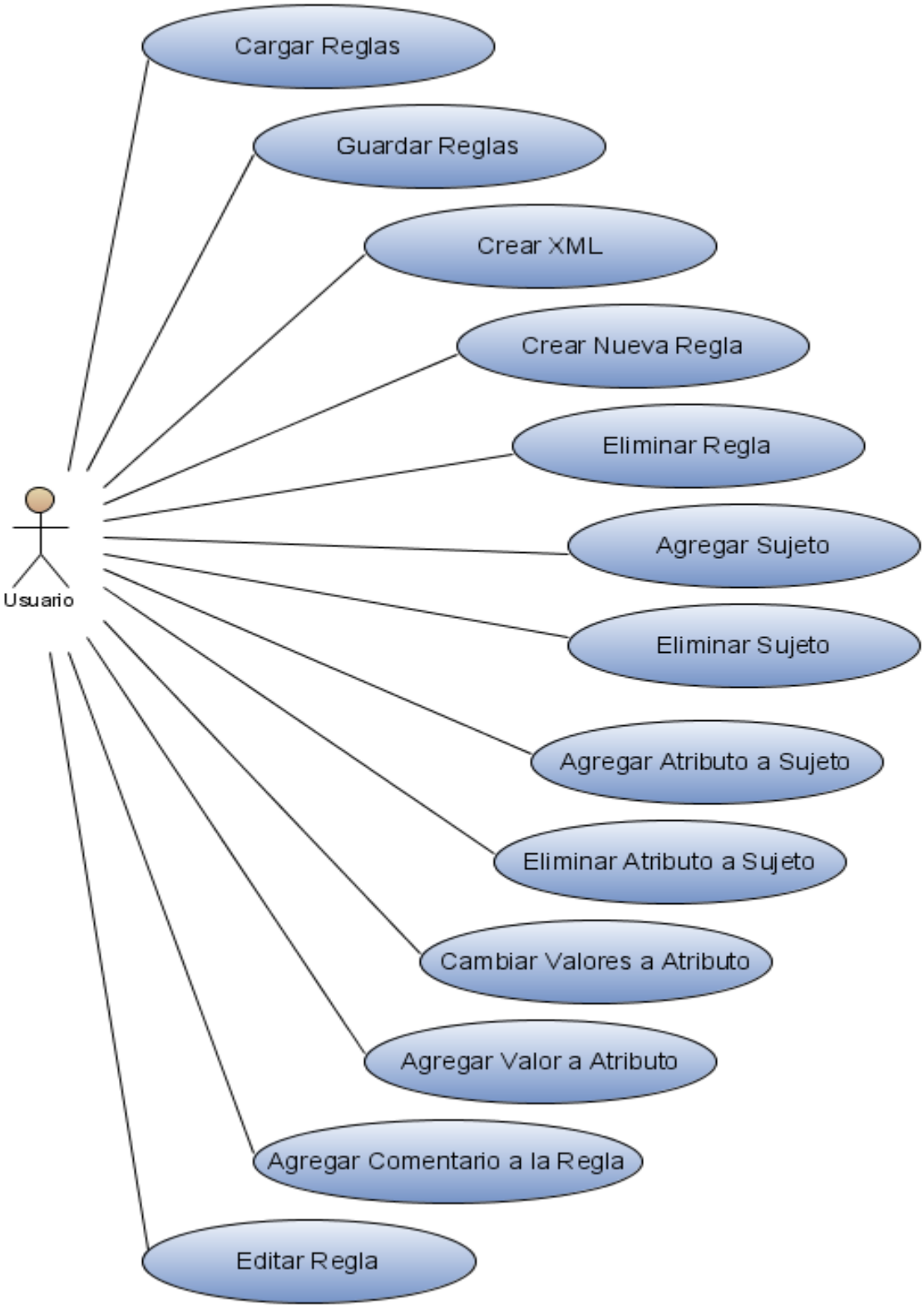

Figura 3.3 Muestra diagrama de casos de uso.

3.3.4 Descripción de los casos de uso.

Caso de uso	Descripción
Cargar Reglas	El actor utiliza este caso de uso para cargar un repositorio de reglas ya este en .xml o .cfg
Guardar Reglas	El actor utiliza este caso de uso para guardar en .cfg los cambios hechos por él, en la aplicación.
Crear XML	El actor utiliza este caso de uso con el fin de crear un documento XML con todas las reglas editadas.
Crear Nueva Regla	El actor utiliza este caso de uso para añadir una nueva regla al repositorio temporal de reglas.
Eliminar Regla	El actor utiliza este caso de uso para eliminar una regla del repositorio temporal de reglas.
Agregar Sujeto	El actor utiliza este caso de uso para añadir un nuevo sujeto a la lista.
Eliminar Sujeto	El actor utiliza este caso de uso para eliminar un sujeto de la lista.
Agregar Atributo a Sujeto	El actor utiliza este caso de uso con el objetivo de añadir un nuevo atributo a un sujeto
Eliminar Atributo a Sujeto	El actor utiliza este caso de uso con el fin de eliminar un atributo de un sujeto.
Cambiar Valores a un Atributo	El actor utiliza este caso de uso con el objetivo de que se puedan arreglar los valores de un atributo en caso de que exista algún error en ellos o simplemente se quiera modificarlos.
Agregar Valor a un Atributo	El actor utiliza este caso de uso para añadir un nuevo valor a un atributo.
Editar Regla	El actor utiliza este caso de uso con el objetivo de editar las RN, reglas que serán guardadas en un repositorio XML y que es el fin de este trabajo.
Agregar Comentario a la Regla	El actor utiliza este caso de uso con el fin de que la regla tenga un comentario que puede ser de mucha utilidad en usos posteriores.

Tabla 3.1: Descripción de los Casos de Uso.

3.4 Documento XML utilizado como repositorio de RN

El editor de reglas utiliza un documento XLM para darlo como salida con el conjunto de reglas procesadas. Este documento sigue una estructura bien definida, acogiéndose a la forma de representación informal de las RN, visto en el Capítulo 1 de este trabajo. Inicialmente muestra el prólogo, aquí aparece la versión y el tipo de codificación, seguidamente aparecen diferentes etiquetas que indican el tipo de información que se almacena. Se describen a continuación las diferentes etiquetas consideradas.

<Repositorio> enmarca cada una de las reglas generadas. Cada vez que comienza una nueva regla se presenta esta etiqueta, dentro de ella se encuentran otras etiquetas, descritas a continuación.

<idregla> indica un número único para cada regla.

<Estado> se especifica si la regla es ACTIVA o INACTIVA.

<Sujeto> indica el sujeto principal de la regla.

<Característica> aparece la condición que debe cumplir el sujeto, en esta se muestran sus atributos

<Hecho> se encuentra la sentencia a seguir en caso de cumplirse la condición.

<Ultima_Fecha> fecha de la última edición de esta regla.

<LenguajeNatural> aparece la regla en lenguaje natural.

<Comentario> se ubica cualquier comentario hecho acerca de la regla.

De esta forma se estructuran todas las reglas del repositorio, permitiendo a otras herramientas apoyarse en él para lograr generar de manera automática el conjunto de reglas almacenadas (Pérez Alonso, 2008).

A continuación se muestra un ejemplo de un fragmento del documento XML, este es obtenido para el caso de la regla número 7, referente a la edad del paciente.

```
<?xml version="1.0" encoding="UTF-8"?>

<Repositorio>
<Reglas>
    <idRegla>RN#7</idRegla>
    <Estado>ACTIVA</Estado>
```

```
    <Sujeto>Patient</Sujeto>
    <Caracteristica>age &gt; 18 and age &lt; 55 </Caracteristica>
    <Hecho>Accept the patient as potential donor for kidney
           transplantation</Hecho>
    <Ultima_Fecha>Fri Jun 19 14:23:52 EDT 2009</Ultima_Fecha>
    <LenguajeNatural>A patient is between 18 and 55 yrs of
            oldAccept the patient as potential donor for kidney
            transplantation</LenguajeNatural>
    <Comentario>Esta regla restringe la edad del
         donante</Comentario>
</Reglas>
           *
           *
           *
</Repositorio>
```

3.5 Descripción del Editor de reglas de negocio

Este epígrafe muestra el ambiente diseñado para la aplicación de software. Se muestran diferentes ventanas que ilustran la filosofía de trabajo en la edición de RN con esta herramienta.

3.5.1 Ventana inicial

Esta ventana es la principal del sistema, permite comenzar el trabajo a partir de (véase la figura 3.4):

- Un documento XML generado por el Protégé.
- Un documento .cfg previamente creado con la propia herramienta.
- Un nuevo documento, que es cuando no hay reglas cargadas en la aplicación.

Una vez que se hace una revisión sobre las reglas existentes, se pueden hacer comentarios sobre ellas, se da la opción al usuario de estructurar estas reglas y realizar la validación semántica apoyándose en los sujetos y atributos almacenados por la herramienta.

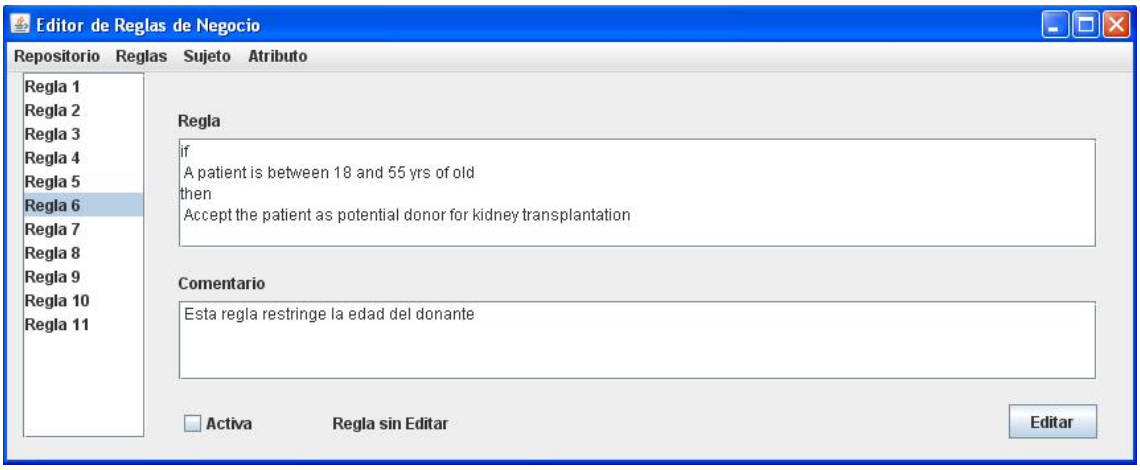

Figura 3.4 Editor de Reglas de Negocio

El "Editor de Reglas de Negocio", que es el nombre de la aplicación, visuliza el repositorio de reglas al experto del negocio, lo que permite que este chequee las reglas directamente, que se le pueda agregar comentarios a las reglas, lo que es de suma importancia para su uso posterior, que se pueda activar la regla o no en dependencia de su uso, luego se diseña un editor que es el que posibilita que estas reglas sean editadas y se genere un documento XML que será utilizado para la integración con otras herramientas. Además de presentar un menú que permite hacer operaciones tales como:

- Agregar o eliminar una regla.
- Agregar o eliminar un sujeto.
- Agregar un atributo a un sujeto o eliminarlo.
- Agregar o modificar un valor a un atributo.

3.5.2 Crear un nuevo documento

Para trabajar con el editor de RN partiendo de un nuevo documento es preciso crear antes un nuevo conjunto de sujetos y atributos para luego poder conformar las nuevas reglas. Estas opciones del sistema también se utilizan para editar las reglas ya existentes.

3.5.2.1 Acciones sobre Reglas

Al seleccionar la opción del menú Agregar Reglas, podemos añadir nuevas reglas según la estructura básica, véase la figura 3.5:

If <condicion> then <accion>

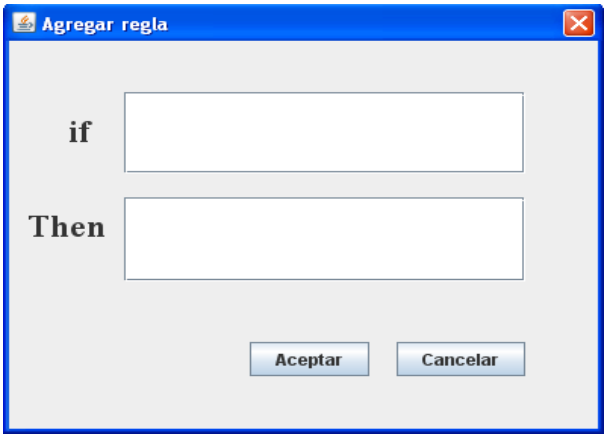

Figura 3.5 Diálogo para agregar una nueva regla.

Este diálogo permite crear nuevas reglas que son añadidas al repositorio, primero se pone en la parte del if la condición y en el then la consecuencia.

Para eliminar una regla se selecciona la regla que se desea eliminar, tomando opción de menú Regla y luego Eliminar.

3.5.2.2 Acciones sobre Sujeto

La opción del menú Sujeto, posibilita agregar y eliminar un sujeto, permitiendo partir desde la aplicación vacía o simplemente basado en los sujetos ya existentes. Al seleccionar la opción del menú Agregar Sujeto se muestra un diálogo similar al de la figura 3.6.

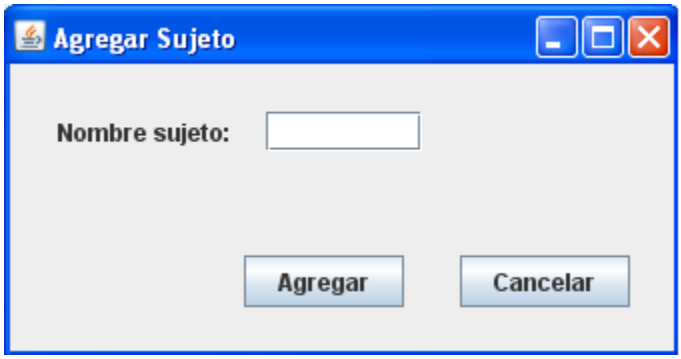

Figura 3.6 Ventana para agregar un nuevo sujeto

Para agregar un sujeto sólo se pone el nombre del nuevo sujeto que se desea añadir y se da Aceptar.

Para eliminar un sujeto se selecciona la acción apropiada mostrándose la ventana similar a la de la figura 3.7 y el sujeto a eliminar.

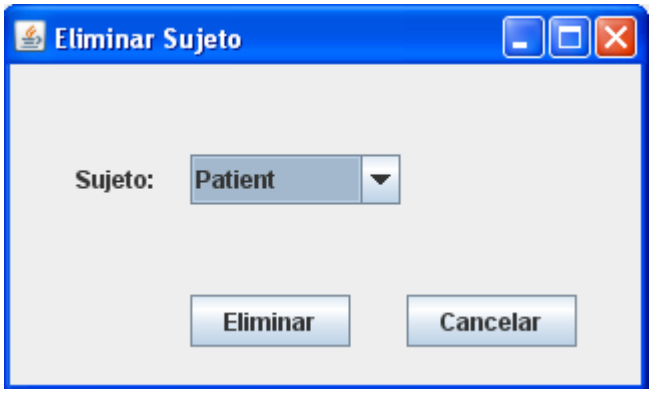

Figura 3.7 Ventana para eliminar sujetos.

3.5.2.3 Acciones sobre Atributos

En este menú se tienen las diferentes opciones:

- Agregar un atributo a un sujeto, véase la figura 3.8.
- Agregar valor a un atributo, véase figura 3.9.
- Modificar valores a un atributo, véase figura 3.10.
- Eliminar Atributo, véase figura 3.11.

Estas opciones se detallan a continuación.

Para agregar un atributo a un sujeto se debe elegir el nombre del sujeto al cual se le va a agregar el atributo, luego se escoge el tipo de atributo (CONJUNTO, ENTERO, CADENA).

Si el tipo que se escogió es CONJUNTO o CADENA se le puede especificar los valores predeterminados. Estos valores predeterminados deben darse separados por coma o cambio de línea en el cuadro de diálogo para este fin.

En caso de ser elegido ENTERO el atributo será considerado como un valor numérico. Si se selecciona tipo CADENA también tendrá valores predeterminados pero de estos sólo podrá tomar un valor. Finalmente se da un nombre al nuevo atributo.

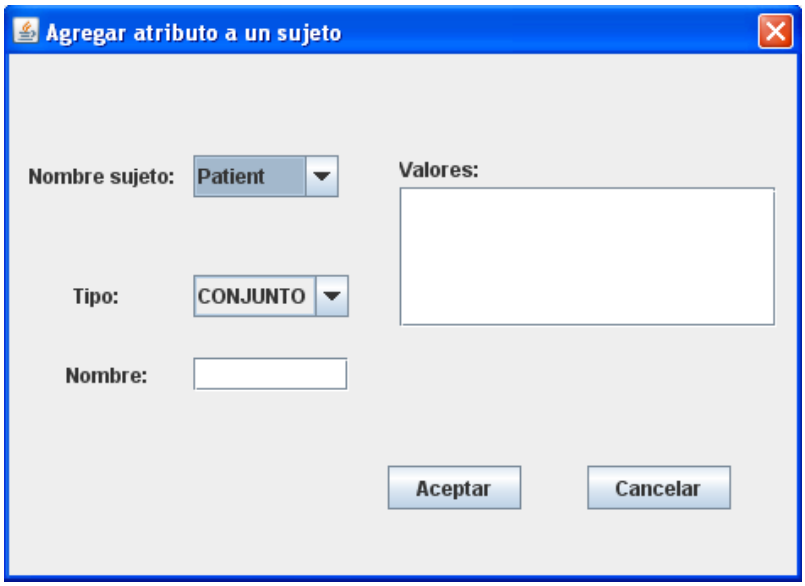

Figura 3.8 Ventana para agregar un nuevo atributo.

Para agregar un valor a un atributo se selecciona el sujeto al cual pertenece este atributo, luego se escoge el atributo deseado, atributo que sea del tipo CONJUNTO O CADENA y se le agrega el valor, véase la figura 3.9.

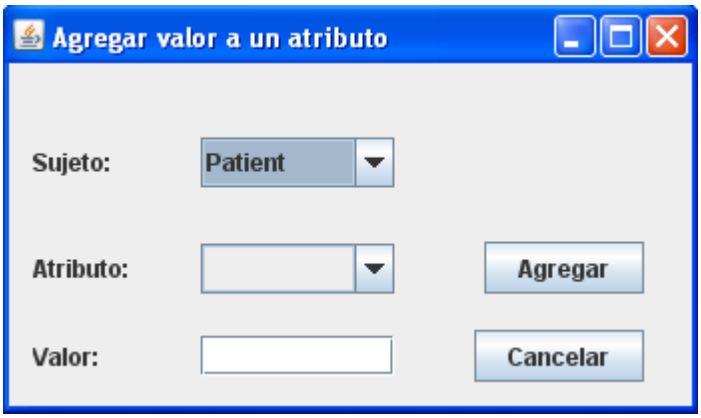

Figura 3.9 Ventana que agrega un nuevo valor a un atributo.

Para modificar los valores de un atributo, se escoge el sujeto y luego aparecerá en atributos aquellos cuyo tipo son CONJUNTO o CADENA que son los que tienen valores predeterminados. Luego en el área de texto aparecen los valores y ahí mismo se modifican como el usuario desee, para finalizar se da Aceptar y los cambios son guardados. véase la figura 3.10.

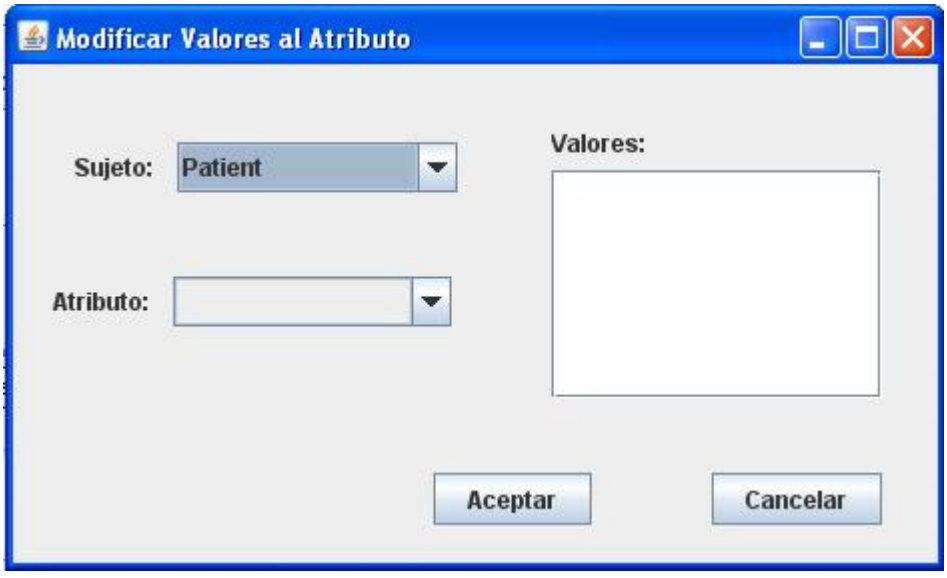

Figura 3.10 Ventana que modifica los valores a un atributo.

Por último, para eliminar un atributo se elige el sujeto al cual el corresponde y se escoge el atributo que se desea eliminar, veáse la figura 3.11.

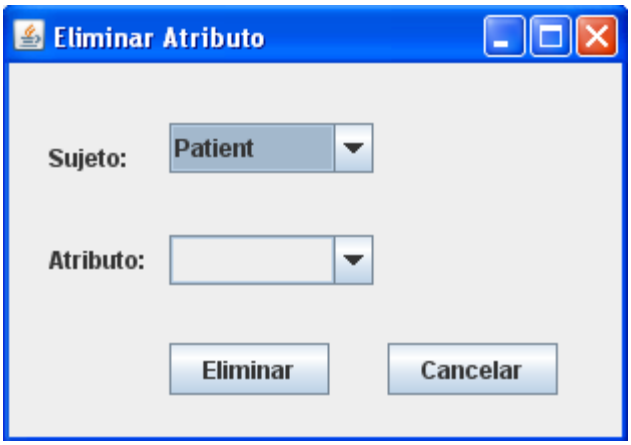

Figura 3.11 Ventana para eliminar un atributo.

3.5.3 Editar la Regla de Negocio

Esta ventana tiene un ambiente similar a la ventana principal, ampliándose ahora con otras opciones después de haber oprimido el botón Editar, véase la figura 3.12, la cual permite editar las reglas de negocio.

Cuando se desea editar una regla primero se escoge el sujeto de esta, de una lista ya formada, después el atributo al cual se refiere en la regla, luego se selecciona el operador, el valor y en caso de que esta regla sea compuesta el operador lógico, que añadirá una nueva fila para que nuevamente elijas un nuevo atributo y así sucesivamente. Al oprimir el botón Aceptar serán guardados en la memoria temporal del programa, para que posteriormente, si usted desea en el menú Repositorio hacer clic en el submenú Crear XML y generará un XML con todas la reglas editadas, que constituye el resultado final de este editor de reglas.

Si por alguna razón no pudo terminar de editar las reglas y tuvo que cerrar la aplicación, entonces se guarda en el fichero config.cfg para que la próxima vez que inicie todos los cambios efectuados estén guardados y puedas seguir sin necesidad de empezar nuevamente.

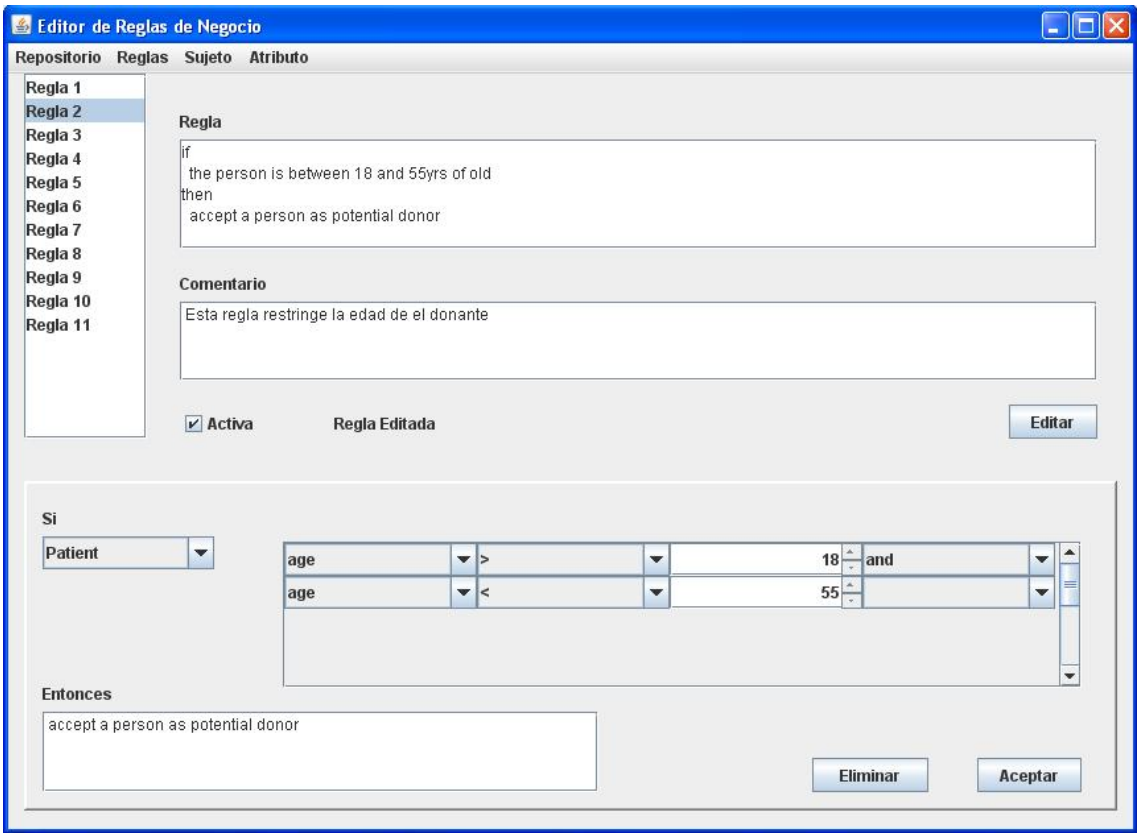

Figura 3.12 Ventana para la edición de las reglas.

3.6 Requerimientos del software.

Para el trabajo con esta herramienta se necesitan los siguientes requerimientos de software:

- Instalar la máquina virtual de java, versión 1.6 o superior.
- Documento XML generado por el Protégé el cual se carga al iniciar la aplicación por primera vez.

Este último requerimiento es opcional, pues la herramienta de edición de RN permite comenzar el trabajo desde cero, o sea, partir desde el punto en que se comienzan a entrar las RN.

3.6.1 Procedimiento a seguir

Al iniciar la aplicación se carga un fichero, nombrado “config.cfg”, el cual debe estar ubicado en la carpeta donde se encuentra el software, de no existir, se inicia la aplicación sin reglas en el repositorio, o sea, vacía.

En el caso de iniciarse la aplicación vacía, se puede mediante la opción del menú Repositorio, Cargar Reglas, véase la figura 3.13, acceder a un repositorio de reglas ya existente en XML que sea generado por el protégé, o un .cfg creado en una ocación anterior por el mismo programa.

Figura 3.13 Opciones del menú *Repositorio*.

El diálogo que se muestra en la figura 3.14 es consecuencia de elegir la opción del menú anterior. Este tiene un filtro que es el que permite cargar las reglas solo desde un archivo .xml o .cfg creado anteriormente, para esto solo tiene que seleccionar el tipo de formato que desea cargar y luego el fichero.

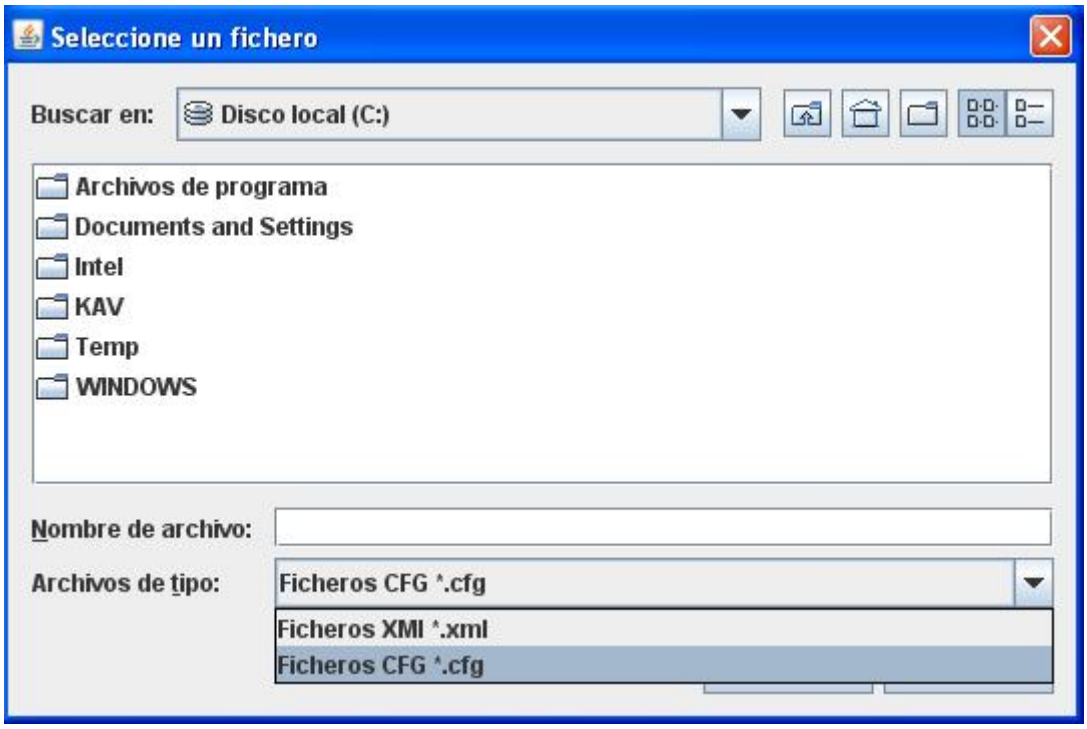

Figura 3.14 Opciones del menú Repositorio.

1.1. Consideraciones finales del capítulo

En el desarrollo de este capítulo se abordan diferentes formas modernas de implementación de RN, enmarcando el trabajo en las herramientas independientes de la base de datos como las apropiadas para esta investigación. Se describe un ambiente de trabajo para el editor de reglas, definiendo la estructura del documento XML que se da como salida, asumiendo una forma de representacion informal para las RN. Se documenta también el software , las clases, los métodos que se implementan, así como una ayuda para la utilización del mismo con todos los pasos a seguir, sus requerimientos y manera de empleo, lo que le facilita al usuario el manejo del mismo.

.

CONCLUSIONES

Con el desarrollo de este trabajo se logra una herramienta que permite a los usuarios del negocio el manejo de las reglas en un lenguaje cercano al natural. Se implementa, además, un editor que permite hacer validaciones sintácticas y semánticas, dando una estructura específica y ofreciendo al usuario un conjunto de opciones organizadas que guían la estructura y el significado de cada regla. Se elige el formato XML como base para el almacenamiento del repositorio, por su compatibilidad entre sistemas que posibilita compartir la información de una manera fácil, con una estructura dada, facilitando la integración con otras herramientas de procesamiento de reglas de negocio.

RECOMENDACIONES

Para dar continuidad a este trabajo se hacen las siguientes recomendaciones:

1. Trabajar sobre otros patrones de reglas de negocios encontrados en la bibliografía.
2. Probar la herramienta con otros repositorios temporales generados con la herramienta Protégé.
3. Realizar el análisis de conflictos y consistencia al repositorio.

REFERENCIAS BIBLIOGRAFICAS

APPELTON, D. S. (1984) “Business Rules - The Missing Link”. Datamation, Vol.30, pp. 145-150.

APPELTON, D. S. (1984) Business Rules - The Missing Link. Datamation, 30, Nro 16, 145-150.

BAJEC, M., KRISPER, M. & RUPNIK, R. (2000) Using Business Rules Technologies To Bridge The Gap Between Business and Business Applications. IN RECHNU, G. E. (Ed.) Proc. of the IFIP 16th World Computer Congress 2000, Information Technology for Business Management. Peking, China.

BAJEC, M. K., M.; RUPNIK, R. (2000) Using Business Rules Technologies To Bridge The Gap Between Business and Business Applications. IN RECHNU, G. E. (Ed.) Proc. of the IFIP 16th World Computer Congress 2000, Information Technology for Business Management. Peking, China.

BRG (2000) Defining Business Rules ~ What Are They Really?

CHAPPEL, O. (2005) "Term-Fact Modeling, the Key to Successful Rule-Based Systems". Business Rules Journal, Vol. 6.

GASEVIC, D., DJURIC, D. & DEVEDZIC, V. (2006) Model Driven Architecture and Ontology Development, Springer.

HALLE, B. V. (1994) “Back to Business Rule Basics”. Database Programming & Design, pp. 15-18.

HENDRYX, S. (2003) Business Rules and Standards. A Fair Isaac White Paper.

JACOBSON, I., BOOCH, G. & RUMBAUGH, J. (Eds.) (2000) El Proceso Unificado de Desarrollo de Software., Addison Wesley.

KAFLE, B. (2009) Edition of Business Rules for Kidney Transplantation. "Marta Abreu" de Las Villas.

KAFLE, B. (2009) Edition of business rules for kidney transplantation. departamento de Computacion. Santa Clara, Villa Clara, Universidad Central de Las Villas.

MARTÍNEZ, D. B. M. E., BOGGIANO, C. M. B., MORENO, M. D. O. I., GONZÁLEZ, G. L. & HERNÁNDEZ, H. P. (2007) Modelo de Hechos para un Sistema basado en Reglas de Negocio en el Sistema de Salud. UCIencia 2007. Cuba, Universidad de Ciencias Informaticas.

MARTÍNEZ, D. B. M. E., BOGGIANO, C. M. B., PÉREZ, A. A., GONZÁLEZ, G. L. & PÉREZ, V. R. (2007) Implementación de reglas de negocio de restricciones y patrones de sentencias de restricciones con triggers. UCIencia 2007. Cuba.

MICHAEL BARNES, D. K. (1997) Play by the Rules, Byte (Special Report).

MORENO MONTES DE OCA, I. (2008) Representación del Modelo de Hechos Mediante Ontologías. Departamento de Computacion. Santa Clara, Universidad Central de Las Villas.

MORGAN, T. (2002) Business Rules and Information Systems: Aligning IT with Business Goals, Addison Wesley.

NOY, N. F. & MCGUINNESS, D. L. (2005) Ontology Development 101: A Guide to Creating Your First Ontology.

PÉREZ ALONSO, A. (2008) Aplicación para Reglas de Restricción en Negocios. Departamento de Computacion. Santa Clara, Villa Clara, Universidad Central de las Villas.

ROSS, B. R. G. (2000) What are Fact Models and why do you need them? Business Rules Journal.

ROSS, R. G. (1997) "The Business Rule Book: Classifying, Defining and Modelling Rules". Business Rule Solutions, Inc.

ROSS, R. G. (2003) Principles of the Business Rule Approach, Addison-Wesley.

STRUCK, D. L. (1999) "Business Rule Continuous Requirements Environment". Colorado Springs. Colorado, Colorado Technical University.