UNIVERSIDAD CENTRAL "MARTA ABREU" DE LAS VILLAS
VERITATE SOLA NOBIS IMPONETUR VIRILISTOGA. 1948

Facultad de Matemática, Física y Computación
Licenciatura en Ciencias de la Computación
Departamento de Base de Datos

Trabajo de Diploma

Título:

"Modelo de Hechos para Trasplante Renal mediante ontologías"

Autor: *José Álvarez Prado*

Tutores: *M.Sc María Elena Martínez*
Lic. Isel Moreno

Santa Clara - 2008

Resumen

Las reglas de negocio representan un aspecto muy significativo en el ambiente de los negocios. Ellas definen y restringen los procesos de negocios en las diferentes empresas. Las reglas de negocio son representadas mediante el Modelo de Hechos. Este modelo muestra los hechos que componen al negocio. Los hechos se forman a partir de las relaciones existentes entre los diferentes términos del negocio.

El objetivo de este trabajo consiste en diseñar e implementar el Modelo de Hechos para el área de nefrología mediante una ontología, permitiendo la validación semántica mediante otros módulos que conforman un proyecto más amplio.

El software utilizado para la implementación de la ontología que representa el Modelo de Hechos es el Protégé. Este software brinda un conjunto de funcionalidades que agilizan el proceso de implementación, además permite exportar la ontología creada a diferentes formatos como el XML (formato escogido para la ontología de este trabajo), OWL, etc. El archivo XML que contiene la ontología será usado por el Editor de Reglas para validar semánticamente las reglas de negocios para trasplante renal.

Abstract

The business rules represent a very significant aspect in the business environment. They define and they restrict the processes of business in the different companies. The business rules are represented by means of the Facts Model. This model shows the facts that compose to the business. The facts are formed starting from the existent relationships among the different terms of the business.

The objective of this project consists on to design and to implement the Facts Model for the nephrology area by means of ontology, allowing the semantic validation by means of other modules that conform a wider project.

The software used for the implementation of the ontology that represents the Facts Model is the Protégé. This software offers a group of functionalities that speed up the implementation process; it also allows exporting the ontology to different formats like the XML (chosen format for the ontology of this work), OWL, etc. The file XML that contains the ontology is used by the Rules Editor to validate semantically the Business Rules for Renal Transplant.

Índice

Introducción

Los servicios de salud se caracterizan en general por un control de la evolución de los pacientes en consultas generales y su continuidad en consultas especializadas en la medida que los diagnósticos ganan en precisión y los casos así lo requieren. Esta práctica médica, aparentemente sencilla supone un respaldo informativo que en su fase primaria puede ser bastante estándar pero que en la medida que los casos pasan a una atención más especializada, requieren un tratamiento diferenciado y difícilmente se pueden abordar soluciones generales; razón por la cual se inicia esta experiencia con el estudio de los servicios de nefrología y más particularmente, los servicios relacionados con los trasplantes de riñón.

Se reconoce que el éxito que han tenido los trasplantes de riñón, se debe a dos razones: en primer lugar al desarrollo que sin lugar a dudas han tenido las nuevas técnicas quirúrgicas y a los tratamientos médicos asociados. Si bien es cierto que la razón anteriormente mencionada está asociada a avances en Medicina; la segunda razón no menos importante está relacionada con la aparición y desarrollo de estructuras organizativas que permiten el control coordinado de todas las fases que intervienen en un protocolo para un proceso de trasplante y del control de los receptores y de la donación de los órganos, acorde a políticas y leyes locales, nacionales e incluso internacionales relacionadas con la donación de órganos. Es obvio que la organización y coordinación de trasplantes, es una tarea compleja que requiere varias actividades clínicas que involucran numerosas personas y equipos de trabajos; a la vez que un proceso administrativo paralelo a los procesos clínicos.

Desde el punto de vista computacional, para la coordinación y realización de dichas actividades se requiere usualmente la aplicación de tecnologías que responden a diferentes paradigmas computacionales y en base a ello las aplicaciones pueden variar en dependencia del peso que se le asigne a alguno

de ellos; en la actualidad el enfoque conocido por Reglas de Negocio coloca en un primer plano la captación de las políticas, regulaciones, leyes, etc. que deben hacerse cumplir o simplemente observar durante los procesos que se llevan a cabo.

Las Reglas de Negocio son definiciones explícitas que regulan cómo opera un determinado negocio y cómo el mismo es estructurado; se entiende por negocio cualquier tipo de servicio que ofrezca determinada institución. Estas Reglas de Negocio son imprescindibles para el funcionamiento de la empresa o institución correspondiente, así como para el Sistema de Información (SI) que soporta sus procesos.

Las Reglas de Negocio usan los hechos para guiar los procesos del negocio con el objetivo de que este funcione de la manera en que se quiere. Dada la necesidad de representar las reglas de negocio de la forma más cercana posible al lenguaje natural, en este caso para el Sistema de trasplante renal, se desarrollará una herramienta que facilite la validación semántica de las reglas y pueda ser utilizada por otros módulos del Sistema. Estas reglas estarán escritas en español y en un lenguaje semi-formal. Lo planteado anteriormente se logrará mediante el Modelo de Hechos, el cual muestra los objetos del negocio, sus interrelaciones y sus atributos, siendo capaz de distinguir entre las cosas que se almacenarán de forma persistente, probablemente en una Base de Datos, y los objetos, propiedades e interrelaciones que tienen valores definidos pero que no existen como artefactos reconocibles en el Sistema de Información Operacional. El Modelo de Hechos para el trasplante renal será representado mediante el uso de ontologías, ya que estas generalmente se usan para especificar y comunicar el conocimiento del dominio de una manera genérica y son muy útiles para estructurar y definir el significado de los términos.

Objetivo general:

Diseñar e implementar el Modelo de Hechos para el área de nefrología mediante una ontología.

Objetivos específicos:

- Diseñar el Modelo de Hechos para el área de nefrología.
- Implementar el Modelo de Hechos haciendo uso de ontologías.

CAPÍTULO I. LAS REGLAS DE NEGOCIO Y EL MODELO DE HECHOS.

1.1 Introducción

El uso de las reglas de negocio en el diseño e implementación de diferentes aplicaciones ha conllevado a la necesidad de desarrollar herramientas que faciliten la representación de dichas reglas. En este caso se utilizarán las ontologías para describir el Modelo de Hechos para el trasplante renal lo que facilitará la validación de las reglas

Los sistemas de información(Antoniou, 2004) son esencialmente artefactos de conocimiento que capturan y representan el conocimiento sobre ciertos dominios. Los profesionales e investigadores de los Sistemas de Información (SI) han tratado tradicionalmente con los problemas de identificar, capturar, y representar el conocimiento del dominio dentro de los SI.

Las ontologías generalmente se usan para especificar y comunicar el conocimiento del dominio de una manera genérica y son muy útiles para estructurar y definir el significado de los términos.

Se puede afirmar que un SI tiene su propia ontología implícita, ya que se atribuye significado a los símbolos usados según una visión particular del mundo. Sin embargo, de manera explícita, una ontología puede tener distintos roles en un SI.

1.2 Ontología

Una ontología define un vocabulario común para investigadores que necesitan compartir información en un dominio. Ella contiene definiciones de conceptos básicos y sus relaciones que pueden ser interpretadas por una máquina. Las ontologías nos permiten organizar el vocabulario relacionado con una entidad, con una actividad, etc.; de forma que describe los términos básicos, las relaciones que se establecen entre ellos y reglas que permiten combinarlos.

Existen varias definiciones de ontología en dependencia del contexto en que se trabaje pero de manera general se puede decir que una ontología es '*una especificación explícita de una conceptualización*'(Gruber, 1994). Es imposible representar el mundo real, o alguna parte de él, con todos los detalles. Para reproducir algún fenómeno o parte del mundo, llamado dominio, es necesario focalizar o limitar el número de conceptos que sean suficientes y relevantes para crear una abstracción del fenómeno. Así, el aspecto central de cualquier actividad de modelización consiste en realizar una conceptualización, o sea, identificar los conceptos (objetos, eventos, comportamientos, etc.) y las relaciones conceptuales que se asume que existen y son relevantes.

De este modo, independientemente del ámbito en que se desarrollen, la base para una ontología es la conceptualización junto con un vocabulario para referirse a las entidades de un dominio particular. Las ontologías para representar el conocimiento precisan los siguientes componentes*: conceptos, relaciones, funciones, instancias y axiomas.*

Una ontología se define como un vocabulario más una especificación del significado de dicho vocabulario. Esta visión permite distinguir ontologías basadas en el grado de formalidad en la especificación del significado. Las ontologías informales usan un lenguaje natural, las ontologías semiformales proporcionan axiomatizaciones débiles tales como taxonomías y las ontologías formales definen la semántica del vocabulario por una axiomatización completa y efectiva.

Algunas de las razones por las que se desearía crear una ontología son las siguientes(Natalya and McGuinness, 2005):

- Compartir el entendimiento común de la estructura de información entre personas o agentes de software.
- Permitir la reutilización de conocimiento de un dominio.

- Explicitar suposiciones de un dominio.
- Separar el conocimiento del dominio del conocimiento operacional.
- Analizar el conocimiento de un dominio.

A continuación se amplían cada una de estas razones.

Compartir el entendimiento común de la estructura de información entre personas o agentes de software*:*

Esto es uno de los objetivos más importantes al desarrollar una ontología. Por ejemplo, supóngase que existen distintos sitios Web que contienen información sobre algún dominio. Si estos sitios Web compartieran y publicaran la misma ontología subyacente de los términos que usan, entonces agentes de software podrían extraer y agregar información de estos sitios diferentes. Los agentes podrían usar esta información agregada para responder solicitudes de los usuarios o servir como datos de entradas a otras aplicaciones.

Permitir la reutilización de conocimiento de un dominio:

Por ejemplo, existen modelos para diferentes dominios que necesitan representar la noción de tiempo. Esta representación incluye las nociones de intervalos de tiempos, puntos en el tiempo, medidas relativas de tiempo, y cosas por el estilo. Si tal ontología se desarrolla en detalle, otros podrían simplemente rehusarla en sus dominios. Además, si se necesita construir una ontología grande, se puede integrar varias ontologías existentes que describan porciones del dominio más grande.

Explicitar suposiciones de un dominio:

Permite cambiar esas suposiciones fácilmente si el conocimiento del dominio cambia. Suposiciones codificadas explícitamente acerca del mundo en algún

lenguaje de programación hacen que las suposiciones no solo sean difíciles de hallar sino también difíciles de cambiar.

Separar el conocimiento del dominio del conocimiento operacional:

Un ejemplo es describir la tarea de configuración de un producto a partir de sus componentes de acuerdo a especificaciones requeridas e implementar un programa que haga independiente esta configuración de los productos y componentes en sí (McGuinness, 1998). Se puede entonces desarrollar una ontología de componentes de Computadoras y características y aplicar el algoritmo para configurar Computadoras ordenadas a medida. Luego se puede usar el mismo algoritmo para configurar elevadores si se "alimenta" tal ontología con elevador como componente.

Analizar el conocimiento de un dominio:

El análisis formal de los términos es extremadamente valioso al intentar rehusar ontologías existentes y al extenderlas.

1.2.1 Clasificación de las ontologías

La clasificación de las ontologías está en dependencia del grado de generalidad o nivel de dependencia a una tarea o visión en particular(Antoniou, 2004):

- *Ontologías de Alto Nivel:* describen conceptos muy generales como espacio, tiempo, materia, objeto, evento, acción, etc., los cuales son independientes de un problema o dominio en particular. Por lo tanto, parece razonable, al menos en teoría, tener ontologías unificadas de alto nivel para grandes comunidades de usuarios.

- *Ontologías de Dominio y Ontologías de Tarea:* describen, respectivamente, el vocabulario relacionado a un dominio genérico (como medicina o automóviles) o una tarea o actividad genérica

(diagnóstico o venta), mediante la especialización de los términos introducidos en la ontología de alto nivel.

- *Ontologías de Aplicación:* describen conceptos que dependen tanto de un dominio como de una tarea en particular, los cuales frecuentemente son especializaciones de ambas ontologías. A menudo, estos conceptos corresponden a los roles desempeñados por entidades del dominio mientras realizan cierta actividad. Contienen conocimiento esencial para modelar una aplicación particular bajo consideración.

- *Ontologías Terminológicas:* especifican los términos que son usados para representar el conocimiento en el universo del discurso. Suelen ser usadas para unificar vocabularios en un campo determinado.

- *Ontologías de Información:* especifican la estructura de almacenamiento de bases de datos. Ofrecen un marco para el almacenamiento estandarizado de información.

- *Ontologías de Modelado de Conocimiento:* especifican conceptualización del conocimiento. Contienen una rica estructura interna y suelen estar ajustadas al uso particular del conocimiento que describe.

1.3 Reglas de negocio

Las reglas de negocio (RN) son *restricciones*(Kolber, 2000, Ross): ellas definen condiciones que deben mantenerse verdaderas en situaciones específicas. Las RN no describen los procesos ni el modo en que estos se van a llevar a cabo. Las RN definen las condiciones bajo las cuales un proceso se va a ejecutar o las condiciones que beberán existir después de culminado un proceso, véase la figura 1.1.

Las sentencias de reglas de negocio son el elemento clave en la definición de las necesidades e intenciones de los negocios(Morgan, 2002).

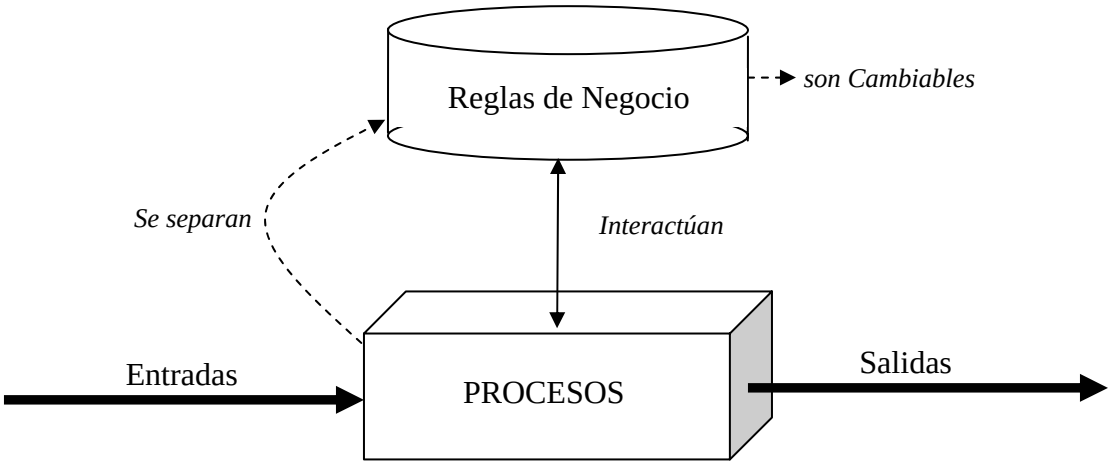


Figura 1.1.Reglas de Negocio.

Las RN conforman una herramienta en el desarrollo de aplicaciones flexibles y Bases de Datos modificables con facilidad(Bajec and Marjan, 2006, Bajec, 2000, Barne and Kelly, 1997, Date, 2000, Youdeowei, 1997). Por lo tanto las RN no son más que declaraciones de políticas o condiciones que deben ser satisfechas.

El trabajo de un analista de negocio es especificar una serie de sentencias claras sobre la lógica implícita del negocio. El énfasis en la claridad es crucial. La sentencia de RN debe ser de forma tal que el dueño del negocio pueda inmediatamente aceptarla como válida o rechazarla como no válida.

En la figura 1.2 se muestra como el analista es el intermediario entre el dueño del negocio y la definición de las RN:

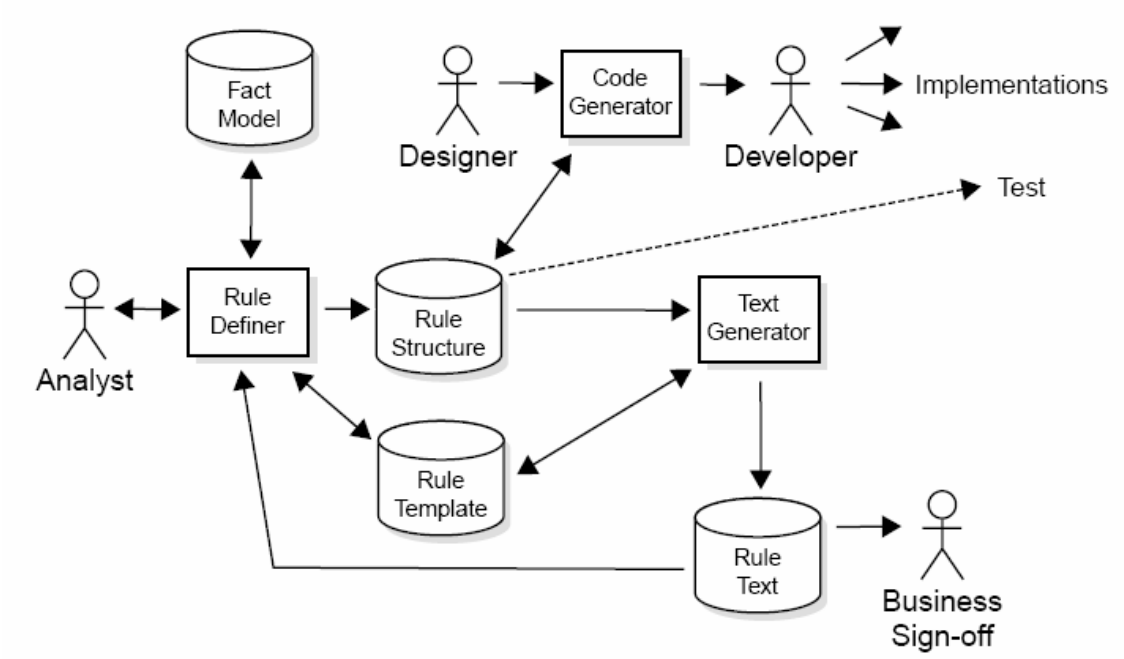


Figura 1.2. Analista de sistemas como intermediario entre el propietario del negocio y las RN

Las sentencias de reglas de negocios deben cumplir con las siguientes características(Morgan, 2002):

- *Atómicas*: no pueden ser divididas sin perder información.
- *No ambiguas:* tienen una sola interpretación obvia.
- *Compactas:* típicamente son sentencias cortas.
- *Compatibles:* usan los mismos términos que son usados en el modelo de negocios.
- *Consistente:* juntas ellas proveen una descripción única y coherente.

Las RN se presentan al menos en tres niveles de expresión(Morgan, 2002):

- *Informal:* sentencias en lenguaje-natural dentro de un rango limitado de patrones.

- *Técnico:* combinación de referencias a datos estructurados, operadores y un lenguaje natural restringido.

- *Formal:* sentencias conforme a una definición mas cerrada de la sintaxis con propiedades matemáticas particulares.

En este trabajo, a la hora de representar las reglas de negocios, lo haremos en un nivel informal, siguiendo la línea planteada por algunos patrones como son: *el patrón de restricción básica, patrón de lista de restricciones, patrón de cálculo, etc*. Sobre esto se amplía en el capitulo III.

1.4 Modelo de hechos

Los términos del negocio son palabras o frases que tiene un significado para las personas del negocio en el contexto en que estos términos son usados. El conocimiento del negocio es expresado usando palabras y frases que tiene sentido para el negocio(Ross, 2003). Los Hechos son combinaciones de términos que describen el conocimiento que presentan las personas acerca de sus negocios. Las reglas de negocio usan los hechos para ayudar a controlar las operaciones de negocios(Chappel, 2005) y así asegurarse de que el negocio funcione de la manera en que las personas quieran que el negocio se lleve a cabo. Los Términos forman la base sobre la cual se construyen los hechos del negocio. Las Reglas de Negocio se construyen encima de los Hechos(Ross, 2005). Las reglas usan a los hechos e interactúan entre sí para guiar las operaciones del negocio como se muestra en la figura 1.3:

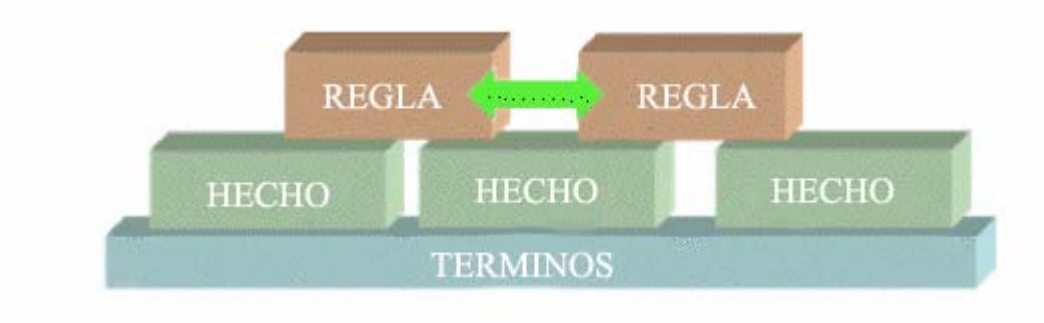


Figura 1.3. Relación entre Términos, Hechos y Reglas de Negocio

Los hechos, sobre los cuales son construidas las reglas de negocio, son descritos en el Modelo de Hechos. Este modelo muestra los objetos del negocio, sus interacciones y sus atributos. Debe tenerse en cuenta que el Modelo de Hechos debe ser capaz de distinguir entre los ítems de artículos del negocio, que no son más que cosas almacenadas de forma persistente, probablemente en la Base de Datos y objetos, propiedades e interrelaciones que tienen valores definidos pero que no existen como artefactos reconocibles en el SI operacional y si lo hacen tienen una existencia más pasajera.

Los términos usados en la construcción de los hechos se categorizan en dos tipos: términos de negocio y términos comunes, donde los términos comunes no son más que aquellos términos que tienen un significado común independientemente del contexto en que se trabaje a diferencia de los términos del negocio que tienen un significado específico en dependencia del contexto del negocio. Formando parte de los Hechos también se encuentran los roles, estos roles no son más que el papel que juegan los diferentes términos en determinados hechos(Kolber, 2000), se debe tener en cuenta que un término puede jugar más de un rol en un mismo Hecho. En la figura 1.4 se puede observar de una manera más detallada la relación término-hecho.

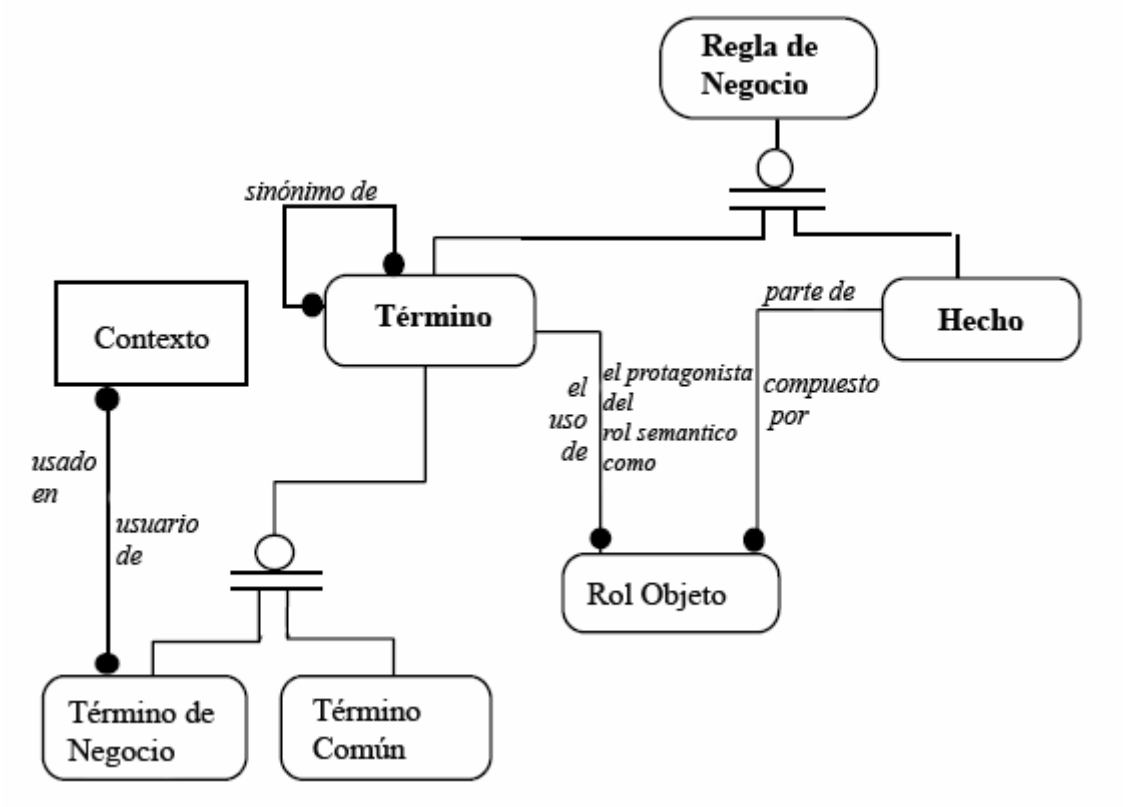


Figura 1.4. Términos y Hechos.

1.5 Relación de las RN con el Modelo de hechos

Las RN contribuyen a elaborar el conjunto de conocimiento de los hechos descritos en un modelo, conceptualmente ese Modelo de Hechos muestra objetos del Negocio, sus interacciones y sus atributos. En particular, el Diagrama de clases de UML completa el rol de un Modelo de Hechos.

La forma más conveniente de crear sentencias de RN es seleccionando de una corta lista de patrones disponibles, el más apropiado. Una forma básica de definir una sentencia sería:

<sujeto> debe <restricción >

Para conformar un patrón apropiado también la RN debe hacer referencia a otros elementos del modelo, principalmente a objetos del negocio y a sus atributos. Esto se puede lograr a través del Modelo de Hechos.

En la figura 1.5 se muestra el rol que desempeñan las sentencias de RN junto con el Modelo de Hechos.

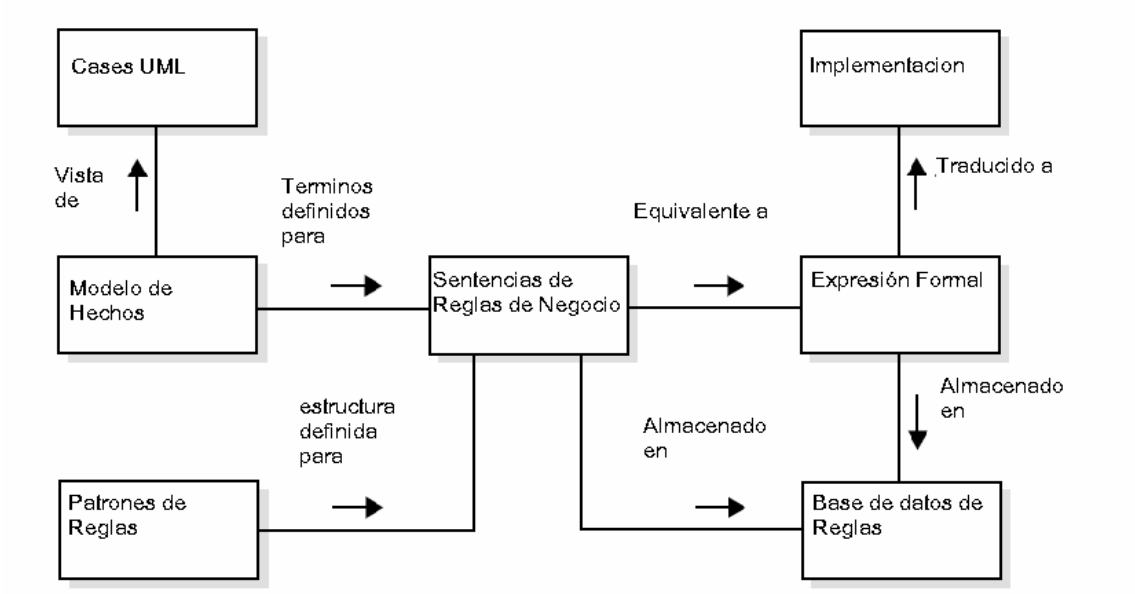


Figura 1.5. Relación de las sentencias de negocio y el Modelo de Hechos

1.6 Las ontologías para la representación del Modelo de Hechos

1.6.1 Introducción

Los hechos junto a los términos del negocio son una forma de representar el conocimiento que tienen las personas acerca de un determinado negocio o dominio. El Modelo de Hechos muestra conceptos del negocio, relaciones entres los diferentes términos, etc. Para lograr un mejor entendimiento del negocio es conveniente confeccionar, junto con el Modelo de Hechos, un vocabulario que contenga todos los términos usados con sus definiciones, sinónimos, equivalentes, etc. De esta manera es que se hace conveniente el

uso de las ontologías para la representación del Modelo de Hechos ya que las ontologías nos permiten organizar el vocabulario relacionado con una entidad, con una actividad, etc.; de forma que describe los términos básicos, las relaciones que se establecen entre ellos y reglas que permiten combinarlos., independientemente del ámbito en que se desarrollen, la base para una ontología es la conceptualización junto con un vocabulario para referirse a las entidades de un dominio particular.

1.6.2 Ontología Terminológica para representar el Modelo de Hechos

La ontología que se construye en el presente trabajo se acerca mucho a una Ontología Terminológica, especifica todos los términos usados en la construcción de los hechos representados en el Modelo de Hechos para trasplante renal. De cada término se tiene en cuenta su significado, sinónimos, términos equivalentes, además de sus relaciones con otros términos lo cual da lugar a los hechos.

Como resultado de lo planteado anteriormente se obtiene un vocabulario unificado, en lenguaje español junto con la representación del Modelo de Hechos logrando un mejor entendimiento del negocio, en este caso para trasplante renal.

CAPÍTULO II. VALORACION DE HERRAMIENTAS USADAS PARA EL TRABAJO CON ONTOLOGIAS

2.1 Metodologías utilizadas en el proceso de creación de ontologías.

Antes de entrar en un análisis de los diferentes lenguajes y herramientas existentes para la implementación de ontologías sería bueno abordar las principales metodologías involucradas en el proceso de creación de ontologías ya que actualmente no existe un estándar definido para la creación de estas. Hoy en día cada grupo de desarrollo usa sus propios conjuntos de principios, criterios de diseño y etapas en el proceso de diseño de una ontología. A continuación se presentan las metodologías más relevantes:

Esquema metodológico para construir ontologías de Uschold & Gruninger:

Este esquema metodológico propuesto por Uschold, constituye la base de muchos métodos propuestos y usados en la actualidad. El esquema está constituido por cuatro pasos y considera además un conjunto de guías o recomendaciones de diseño que se deben tener presentes en cada paso del método.

Paso 1. Identificar propósito y alcance

Señalar claramente el propósito para el cual se construirá la ontología y el alcance de la misma.

Paso 2. Construir la ontología

Este paso considera tres aspectos que son necesarios para llevar adelante la construcción de la ontología, como son:

- *Captura*: Identificar conceptos claves y sus relaciones en el dominio (el autor propone un esquema intermedio para ejecutar esta identificación que

va desde el concepto más general al más particular, y obtener el resto de la jerarquía generalizando y especializando).

- *Codificación*: Representar en lenguaje formal la conceptualización capturada en el estado anterior.
- *Integración*: Examinar ontologías existentes y verificar si pueden ser integradas a la que se está construyendo.

Paso 3. Evaluar

Hacer un juicio técnico a la ontología considerando la conceptualización, el ambiente, el software y la documentación, con respecto a una referencia. Esta referencia puede ser: requerimientos de especificación, preguntas de competencias y/o el mundo real.

Paso 4. Documentar

Documentar adecuadamente el conocimiento expresado en la ontología, para así garantizar que sea apropiadamente compartido y reutilizado

Las guías de diseño que Uschold recomienda aplicar en cada paso de la metodología son similares a las presentadas en la sección 3 de este documento.

Una guía para crear ontologías de la Universidad de Stanford:

Esta guía desarrollada en la Universidad de Stanford está compuesta de siete pasos:

Paso 1. Determinar el dominio y alcance de la ontología.

Definir el dominio y el alcance de la ontología, respondiendo preguntas como, ¿Cuál es el dominio que la ontología cubrirá? ¿Para qué se desarrolla la ontología? ¿Quién usará la ontología? ¿Qué tipo de información proporcionará la ontología?

Paso 2. Considerar reutilizar ontologías existentes.

Chequear si es posible usar y extender fuentes de conocimientos ya existentes, y que puedan ser de utilidad para el dominio del problema.

Paso 3. Enumerar términos importantes en la ontología.

Elaborar una lista de los términos proporcionados por el usuario, indicando propiedades de cada uno. El contenido de la lista debe ser preciso y carente de ambigüedades.

Paso 4. Definir clases y jerarquías de clases.

De la lista creada en el paso 3, seleccionar aquellos términos independientes para constituir las clases. A partir de éstas organizar la jerarquía

Paso 5. Definir propiedades de las clases.

Describir la estructura interna de los conceptos, por lo general los términos que no fueron seleccionados en el paso 4 pasan a considerarse propiedades de las clases (comúnmente denominados *"slots"*).

Paso 6. Definir las características ("facets") de los "slots"

Definir los diferentes tipos de valores que describan a los *"slots"*, tales como, tipo de valor asociado, cardinalidad, valores permitidos (rangos), etc.

Paso 7.Crear Instancias

Crear instancias de las clases de la jerarquía, de la siguiente manera: Seleccionar una clase, crear una instancia, llenar los *"slots"* con los valores posibles.

Método para construir ontologías según Nianbin & Xiaofei:

Este método considera que la construcción de una ontología se lleva a cabo en cuatro pasos, tomando como base la propuesta de Uschold de un esqueleto o guía metodológica:

Paso 1. Análisis del dominio y cadenas de conocimiento.

Adquirir y clasificar los aspectos de conocimiento más importantes en el dominio.

Paso 2. Estructurar la ontología.

Construir la estructura de la ontología según las cadenas de conocimiento, lo cual considera: identificación de conceptos del dominio, identificación de atributos para los conceptos y valores relacionados.

Paso 3. Formalización de la ontología.

Formalizar la ontología utilizando un lenguaje como KIF, Ontolingua, KQML, etc.

Paso 4. Evaluación de la ontología.

Evaluar el rendimiento de la ontología con la finalidad de mejorarla y corregirla de ser necesario.

Methontology.

Esta metodología fue desarrolla en el laboratorio de Inteligencia Artificial de la Universidad Politécnica de Madrid. Permite la construcción de ontologías a nivel de conocimiento e incluye la identificación del proceso de desarrollo de la ontología, un ciclo de vida basado en la evolución de prototipos y técnicas particulares a usar en cada paso. Los pasos de la metodología son los siguientes:

Paso 1. Especificación

Desarrollar un documento que contenga la meta de la ontología, nivel de granularidad, alcance, propósito, etc. Identificar los términos a ser representados, sus características y relaciones.

Paso 2. Conceptualización.

Organizar el conjunto de términos y sus características en una representación intermedia que el desarrollador de la ontología y los expertos puedan entender. En este paso se construye un glosario de términos, diagramas de relaciones binarias, diccionario de conceptos, tablas de atributos instancias, tablas de atributos clases, tabla de axiomas lógicos, tablas de constantes, tablas de instancias.

Paso 3. Adquisición de conocimiento.

Este paso se lleva a cabo de manera independiente de la metodología y su ejecución puede coincidir con otros pasos. Por lo general la adquisición de conocimiento se realiza en tres etapas: reuniones preliminares con los expertos, análisis y revisión de la bibliografía asociada al dominio y una vez que se tiene un conocimiento base se refina y detalla el conocimiento hasta completar la ontología.

Paso 4. Integración

Identificar ontologías candidatas que puedan ser reutilizadas en la ontología que se está construyendo e incorporar aquellas piezas de conocimiento que sean de utilidad.

Paso 5. Implantación.

Consiste en la codificación del modelo conceptual en un modelo codificado en lenguaje ontolingua. La traducción a ontolingua es llevada a cabo por el software ODE *(Ontology Design Environment).*

Paso 6. Evaluación.

Realizar un juicio técnico a la ontología, al ambiente de software asociado y a la documentación con respecto a un esquema de referencia en cada paso de la metodología. El esquema de referencia puede ser: requerimientos de especificación, preguntas de competencias y/o el mundo real

Paso 7. Documentación

Detallar clara y exhaustivamente cada paso completado y los productos generados.

Methontology proporciona un ambiente agradable para adquirir conocimiento, a personas que no están familiarizadas con este proceso, además permite la construcción de modelos de conocimiento de manera efectiva y su posterior validación.

A diferencia de los métodos descritos anteriormente, que proponen la codificación del conocimiento en lenguaje formal, Methontology sugiere que sea expresado mediante un conjunto de representaciones intermedias.

2.2 Lenguajes usados para la descripción de ontologías.

Las **ontologías** son teorías formales acerca de un dominio de discurso y por eso requieren de un lenguaje lógico formal para ser expresadas. De los lenguajes desarrollados algunos están basados en Lógica de Predicados de primer orden, como *KIF* y *Cycl* que proveen poderosas primitivas de modelado. Otros lenguajes están basados en "*Frames"*, con más poder expresivo pero menos capacidad de inferencia, como "*Ontolingua"* y "*Frame Logic"*; otros lenguajes están orientados a ser robustos en el razonamiento que provee Lógica de Predicados, como "*Loom"* y *Classic.*

Los lenguajes basados en *"Frames"* generalmente proveen un conjunto bastante rico de primitivas, pero imponen muy fuertes restricciones sintácticas en cómo esas primitivas pueden ser combinadas, y cómo pueden ser usadas para definir una clase. Los lenguajes basados en Lógica de Predicados (LP), en cambio, tienen un conjunto más restringido de primitivas (se restringe su cantidad para lograr claridad semántica, poder de decisión y la posibilidad de proveer mas procedimientos de razonamiento) pero permite que sus primitivas sean combinadas en expresiones booleanas arbitrariamente y usadas para definir diferentes tipos de clases (en particular clases primitivas, donde la definición es tomada como una condición necesaria de pertenencia, y clases no-primitivas, donde la definición es tomada como una condición necesaria y suficiente para que un miembro pertenezca a la clase). Una diferencia importante entre enfoques basados en *"Frames"* y enfoques basados en LP es que los primeros confían solamente en sentencias explícitas de inclusión en una clase, mientras los otros son capaces de computar eficientemente la relación de pertenencia entre clases basándose en la definición intencional de dichas clases. La capacidad de computar automáticamente dichas relaciones es importante para que la ontología tenga la capacidad de chequeo de consistencia.

A continuación se hace un breve esbozo de varios lenguajes utilizados para la especificación de ontologías:

OKBC (Open Knowledge Base Connectivity).

Proporciona un modelo uniforme de representación de conocimiento basado en una conceptualización común de clases, individuos, *"slots", características ("facets")* y herencia. Existen implantaciones de OKBC en CommonLisp, Java. OKBC Constituye un lenguaje estándar de intercambio para ontologías.

XOL (Ontology Exchange Language).

Es un lenguaje basado en esquemas (*"Frames"*) con una sintaxis en XML para el intercambio de ontologías. La definición de una ontología en XOL incluye información de meta-data tal como definiciones de clases e información de objetos.

KIF (Knowledge Interchange Format).

Desarrollado por el grupo de trabajo Interlingua de la universidad de Standford en 1992, KIF está basado en Lógica de predicados de primer orden, es uno de los más populares entre la comunidad científica. Es un lenguaje diseñado para intercambio de conocimiento entre sistemas de computación. Este lenguaje tiene una notación parecida a LISP para representar los axiomas. Los objetos son denotados como constantes o expresiones de términos, las que son construidas de listas cuyo primer elemento es una constante función. Las sentencias se forman de listas cuyo primer elemento es una constante relación y el resto son términos, o por operaciones lógicas sobre sentencias.

Ontolingua.

Lenguaje desarrollado por KSL *("Knowledge Systems Lab")* en la Universidad de Stanford. Usa una notación tipo LISP para expresar la ontología, pretende ser un lenguaje neutral, sin embargo no soporta máquina de inferencia, simplemente almacena todos los conceptos de un dominio sin inferir algo acerca de ese dominio. No es adecuado para el Diseño de Bases de Datos

debido a su poco poder de expresividad. Proporciona un ambiente de desarrollo con un conjunto de funciones para la construcción de la ontología como: crear, editar, etc., así como una biblioteca de ontologías modulares y reusables. Soporta el diseño y especificación de ontologías en una semántica lógica, definida por un conjunto de axiomas en KIF. La sintaxis de Ontolingua se ha extendido para proporcionar facilidades en la construcción de axiomas en forma de definiciones y en la definición de términos en lenguajes con orientación a objetos y *"Frames"*. La definición de términos considera clases, sub-clases, *"slots"*, cardinalidad de los *"slots", "facets"*, entre otros.

OIL (Ontology Inference Layer)

Es un lenguaje de representación basado en la web que contempla capas de inferencias. Combina la modelación de primitivas propias de los lenguajes basados en *"Frames"* con semántica formal y servicios de razonamiento a través de descriptores lógicos. Incluye una semántica precisa para describir el significado de los términos.

Cada capa adicional agrega funcionalidad y complejidad a la capa anterior. Esto permite que agentes (humanos o máquinas) que se ubiquen en una capa particular puedan entender parcialmente ontologías que se expresan en cualquiera de las capas más altas.

SHOE (Simple HTML Ontology Extensions)

Fue desarrollado en la Universidad de Maryland en 1995, fue el primer lenguaje de etiquetado para diseñar ontologías en la Web, antes de la aparición de la Web Semántica. Es un lenguaje de representación de conocimiento basado en HTML. SHOE es un súper conjunto de HTML que permite adicionar etiquetas a documentos web. Los tipos de etiquetas son: para construir ontologías (conjuntos de reglas que definen qué clase de aserciones pueden hacerse en el documento y que significado adquieren) y

anotaciones en documentos web que permiten suscribirse a una o varias ontologías, declarar entidades de datos y hacer aserciones sobre esas entidades de acuerdo a las reglas de las ontologías.

Ofrece construcciones tales como categorías, interrelaciones, inferencias, renombres y descripciones. Se utiliza para mejorar búsquedas en la Web. Su principal carencia es que no dispone de mecanismos para expresar negaciones o disyunciones

2.3 Herramientas usadas para la implementación de ontologías.

Los editores de ontologías, son herramientas especializadas que apoyan la construcción de estas. Las facilidades que proporcionan van desde la definición y modificación de conceptos, propiedades, relaciones, restricciones y axiomas, hasta la inspección y navegación en ontologías. Algunos de los editores más importantes se describen a continuación.

Ontolingua:

Herramienta de desarrollo para navegar, crear, editar, modificar, verificar, evaluar y usar ontologías. Contiene una librería de ontologías cuyas definiciones, axiomas y términos no-lógicos, pueden ser reutilizadas en la construcción de nuevas ontologías.

Ontolingua basa la construcción de ontologías en el principio de diseño modular. Esto permite que las ontologías de las librerías puedan ser reutilizadas de cuatro diferentes maneras:

- *Inclusión:* Una ontología A es explícitamente incluida en una ontología B.
- *Polimorfismo:* Una definición de una ontología es incluida y refinada en otra.
- *Restricción:* Una versión restringida de una ontología es incluida en otra

- *Inclusión de Ciclos:* Situaciones como la siguiente se pueden dar, más no son recomendables: la ontología A se incluye en la B, la ontología B se incluye en la C y la ontología C se incluye en la A.

Chimaera:

Herramienta que permite crear y mantener ontologías en la web, proporciona un ambiente distribuido para navegar, crear, editar, modificar y usar ontologías. Entre las facilidades que ofrece la herramienta se tienen: cargar bases de conocimiento en diferentes formatos, reorganizar taxonomías, resolver conflictos de nombres y editar términos. Destaca la capacidad para cargar datos de entrada en 15 diferentes formatos, tales como, KIF, Ontolingua, OKBC, Protégé, etc.

OilEd:

OilEd es un editor de ontologías para OIL y DAML-OIL, desarrollado en la Universidad de Manchester. La interface del editor es orientada a *"Frames".* La principal característica de este editor es su capacidad para tratar con un lenguaje expresivo y el mecanismo de razonamiento lógico que usa para chequear la consistencia de las clases y las relaciones de inferencia.

OilEd permite la definición de clases, *"slots"* y axiomas, así como el uso de combinaciones booleanas de *"Frames"* o clases conectadas a través de "*and", "or" o "not*".

The Ontolingua Server

"Ontology Server" (1995): es un conjunto de herramientas y servicios que dan soporte a la construcción de ontologías que comparten grupos geográficamente distribuidos. Se desarrolló en el laboratorio de Sistemas de Conocimiento de la universidad de Standford. Este servidor es una extensión de Ontolingua. Al comienzo el término Ontolingua se usaba para referirse tanto al lenguaje para representar ontologías como a la herramienta utilizada para

construirlas. Hoy en día el término se utiliza para referirse al lenguaje proporcionado por el "*Ontology Server"*. La arquitectura de este servidor permite el acceso a una librería de ontologías, a traductores de lenguajes (LOOM, IDL, CLIPS, etc.) y a un editor para crear y navegar por una ontología. Consta de un conjunto de herramientas y servicios que soportan la construcción colaborativa de ontologías entre grupos geográficamente dispersos. Provee un repositorio de ontologías ya creadas y la posibilidad de adicionar nuevas a él.

Hay tres modos de interactuar con el servidor de ontolingua:

- Colaboradores remotos que pueden editar y visualizar ontologías.
- Aplicaciones remotas que pueden consultar y modificar ontologías a través de internet usando el "*Generic Frame Protocol*" (GFP), ya que KIF, la base de especificación de Ontolingua, es puramente declarativo y no provee comandos para consulta o manipulación de datos.
- Aplicaciones "*stand-alone"*.

WebOnto

Completamente accesible por internet, desarrollada por el *"Knowledge Media Institute of the Open University"*. Soporta la visualización, creación y edición colaborativa. El lenguaje usado para la construcción es OCML *("Operational Conceptual Modeling Languaje")*. Brinda la posibilidad de salvar diagramas de estructuras, visualización de relaciones, clases y reglas separadamente.

Una característica sobresaliente de la herramienta es la claridad de su interfaz gráfica. La visualización gráfica de la ontología presentada por WebOnto es una de las mejores en relación a las restantes herramientas consideradas. No solo la ontología completa puede ser visualizada, sino también partes seleccionadas de la misma, por ejemplo, solo las clases, o solo las instancias, etc.

Todas las estructuras juntas, pueden volverse confusas, sin embargo, debido a la capacidad de ver solo partes de ella, el usuario es capaz de conseguir una clara visión del dominio.

OntoSaurus

Basada en la *"web",* similar a Ontolingua, pero usando representación en LOOM ODE, una herramienta basada en *"Windows",* para construir ontologías y estrategias de razonamiento.

Está diseñado para ser usado sobre bases de conocimiento soportadas por Loom. Es un proyecto de investigación del grupo de AI de la *"University of Southern California's Information Sciences Intitute"* (desarrolladores de LOOM). Consiste de dos partes: Un servidor de ontologías que usa al sistema Loom para la representación de conocimiento y un servidor de edición que dinámicamente crea páginas HTML que despliegan la jerarquía de la ontología y usa "*forms html*" para permitir al usuario la edición de la misma. De esta forma se puede ver al mismo tiempo el contenido de un concepto y el entorno de la estructura ontológica a la que pertenece. Han sido desarrollados interpretadores de Loom para Ontolingua, KIF y C++.

ODE – Ontology Design Environment

La principal ventaja de ODE es el módulo de conceptualización para construir ontologías, que permite al creador desarrollar la ontología usando un conjunto de representaciones intermedias que son independientes del lenguaje final en que será representada la ontología. Una vez que la conceptualización está completa, el código es generado automáticamente para Ontolingua. Usando este ambiente, usuarios no expertos en el lenguaje en el cual la ontología será implementada, serán igualmente capaces de especificar y validar la misma. ODE está siendo diseñado por la Universidad Politécnica de Madrid.

ODE necesita ser instalado localmente, y solo es soportado el trabajo de manera asíncrona. Tampoco existe un manejador de versiones. Internamente, la ontología ODE está almacenada en una base de datos relacional, lo que

facilita su integración con software tradicional. Con respecto a facilidades de exportación e importación, la versión actual de ODE genera Ontolingua y F-Logic, así como también es posible importar ontologías expresadas en esos lenguajes.

Protégé:

Es una herramienta implantada en java que permite la construcción de ontologías, personalizando las formas de adquisición de conocimiento y los dominios de conocimiento. Su propósito es construir ontologías de modelos de dominio, y ha sido diseñado por *"The Stanford's Medical Informatics Section".* Fue diseñado para asistir a los desarrolladores de software en el proceso de crear y mantener modelos explícitos de dominio.

Es capaz de operar como una plataforma para acceder a otros sistemas basados en conocimiento o aplicaciones integradas, o como una librería que puede ser usada por otras aplicaciones para acceder y/o visualizar bases de conocimiento. La herramienta ofrece una interfaz gráfica que permite al desarrollador de ontologías enfocarse en la modelación conceptual sin que requiera de conocimientos de la sintaxis de los lenguajes de salida, como OIL o RDFS. Además facilita la creación de una estructura de *"Frames"* con clases, *"slots"* e instancias de una forma integrada.

Es un programa basado en *"Windows".* Posibilita la herencia múltiple. La herramienta chequea en tiempo real en el editor de la ontología si se mantiene la consistencia al agregar un nuevo dato, en otras partes de la herramienta no hay chequeos de consistencia pues el chequeo está relacionado al ambiente de edición y no a la ontología en sí. No es posible trabajar en la edición de una misma ontología más de un usuario a la vez, ya que Protégé es una herramienta que se instala localmente.

Las aplicaciones desarrolladas con Protégé son empleadas en resolución de problemas y toma de decisiones en dominios particulares. Dispone de un conjunto importante de "*plugin*" con orígenes diversos, con la posibilidad de

que sean conectados otros de estos módulos externos y proporcionar una funcionalidad adicional, haciendo así a Protégé extensible, aumentando su utilidad y adaptación según las necesidades. En el 2003 fue extendido para soportar OWL a través de la creación del OWL plugin, el cual también proporciona interfaces para razonadores de lógica descriptiva.

El modelo de conocimiento de Protégé es OKBC. Esto hace que esta herramienta incluya además de las clases, jerarquías de clases con múltiples herencias, se facilita la creación de una estructura de *"Frames"* con clases, formularios, *"slots"* e instancias de una forma integrada. Las ontologías creadas en Protégé pueden ser exportadas a varios formatos como RDF(S), OWL y XML(S). 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.

De todas las herramientas existentes para la implementación de ontologías, Protégé es la más usada. Al ser software libre y por las características antes mencionadas, Protégé será la herramienta escogida para la implementación de la ontología para representar el Modelo de Hechos para trasplante renal.

CAPÍTULO III: DISEÑO DEL MODELO DE HECHOS.

3.1 Introducción

Entre los diferentes aspectos a considerar durante la construcción de las reglas de negocios se encuentra el uso del Modelo de Hechos, en el cual las reglas de negocio puedan relacionarse con otras partes del modelo de negocios. Estas reglas siempre deben relacionarse con elementos visibles en el Modelo de Hechos. La acción de definir una regla de negocio puede conllevar a que se requieran modificaciones en otras facetas del modelo de negocio, por tanto el modelo entero se debe tratar como algo evolutivo. Si es necesario realizar cambios no se debe olvidar de chequear el posible impacto en otras reglas ya definidas anteriormente.

3.1.1 Metodología para el desarrollo del Modelo de Hechos

Se debe seguir una metodología de trabajo para obtener el Modelo de Hechos. A continuación se describe una serie de pasos(Ross and Lam) que permiten conformar el conjunto de términos del negocio que posteriormente se utilizarían para el diseño del Modelo de Hechos formando así la base que daría soporte a las reglas de negocios:

- *Captura de términos y definiciones ya existentes:* Hallar y recuperar conjuntos y definiciones existentes en los glosarios de la industria, glosarios corporativos, documentación del proyecto, manuales, repositorios, diccionarios de datos, y/o otras fuentes disponibles. Evaluar estos términos y condiciones para la calidad, integridad y relevancia, y determinar donde existen limitaciones, huecos y/o discrepancias desde la perspectiva del negocio.

- *Revisar términos concretos:* Identificar y examinar términos concretos, es decir, esos términos que se refieren a personas, lugares y cosas. Desarrollar categorías relevantes y seleccionar nombres apropiados.

- *Repasar términos abstractos:* Identificar y examinar términos abstractos, es decir, esos términos que se refieren a conceptos y eventos (transacciones e interacciones del negocio). Seleccionar nombres apropiados y desarrollar el primer corte de definiciones.

- *Desarrollar el Modelo de Hechos:* Desarrollar el Modelo de Hechos, mostrando los términos y hechos, incluyendo Categorías y Propiedades donde sea apropiado. Esto permite una vista unificada del conocimiento operacional básico necesario para dar soporte al flujo de trabajo y al funcionamiento continuo de los Procesos de Negocios.

- *Definir los Términos:* Crear una definición, concisa y orientada al negocio, para cada término que sea consistente con su objetivo comercial y refleje su uso en el Modelo de Hechos. Organizar las definiciones dentro del Catálogo de Conceptos.

- *Revisar las Reglas:* Revisar las Reglas existentes para asegurarse que toda la terminología este representada en el Modelo de Hechos.

- *Compilar las Reglas y el Modelo de Hechos:* Un Modelo de Hechos bien desarrollado proporciona la base definiendo las Reglas fundamentales para guiar los procesos de negocios. Examinando el Modelo de Hechos y las Definiciones relacionadas permiten que las Reglas sean capturadas y examinadas.

3.2 Herramienta utilizada para el diseño del Modelo de Hechos

Actualmente no existe una herramienta en específico para el diseño del Modelo de Hechos. En este trabajo se utilizo el software MagicDraw para tal diseño, ya que permite hacer diagramas con una estructura muy similar a la que presenta un Modelo de Hechos.

MagicDraw es un modelador de UML y una herramienta CASE con soporte para trabajo en equipo. La misma está considerada una versátil y dinámica

herramienta de desarrollo para el análisis y diseño de sistemas orientados a objetos y base de datos. Además, dispone de un reconocido mecanismo de ingeniería de código (con total soporte tanto en ingeniería directa como inversa en entornos como J2EE, C#, C++, CORBA, lenguaje de programación IDL, .NET, XML, Schema, WSDL).

Ediciones de MagicDraw:

- Edición Personal
- Edición Estándar
- Edición Empresarial
- Edición de Lectura
- Edición para la Comunidad

La versión utilizada en este trabajo es MagicDraw 9.0, Edición Empresarial.

3.3 Modelo de Hechos para el trasplante renal

Como se ha dicho en otras ocasiones, las reglas de negocios (RN) son sentencias que definen o restringen algunos aspectos del negocio con la intención de controlar o al menos influir en el comportamiento de estos. Las reglas de negocio deben ser atómicas, o sea, que una regla no puede ser dividida o descompuesta en otras reglas y si esto ocurre, causa pérdida de información importante para el negocio.

Para lograr un fácil manejo de las RN es necesario llevarlas del lenguaje natural a un lenguaje semiformal, esto se logra aplicando patrones ya definidos con anterioridad que con llevan a una estructura común para todas las sentencias de reglas de negocio. A continuación se muestra detalladamente el proceso de transformación de una regla escrita en lenguaje natural a una sentencia en lenguaje semiformal.

Primeramente se parte de una regla escrita en lenguaje natural sin ajustarse a ningún patrón:

r0: Todo aquel que sea *Posible Receptor* es asociado a un grupo de *Posibles Donantes.*

r1: el *Seguimiento para Receptor* se puede clasificar en dos tipos: *Seguimiento Intrahospitalario* o *Seguimiento Extrahospitalario*

r2: Todos los *Pacientes* con *Diagnóstico IRCT* (Insuficiencia Renal Crónica Tratable) tienen como opción de *Tratamiento: Tratamiento Dialítico o Trasplante de Riñón* (tiene que tener un *Donante Vivo* o un *Donante con Muerte Encefálica).*

3.3.1 Convenios para escribir los patrones de reglas

Posteriormente se hace un análisis para seleccionar, de los patrones existentes, el más adecuado para representar dicha regla en un formato más compacto (Morgan, 2002), donde:

Paréntesis ( ): encapsula a un grupo de términos.

Corchetes [ ]: encapsula términos opcionales.

Barra vertical **|:** separa términos alternativos.

Corchetes Angulares < >: encapsula términos especiales.

La siguiente tabla muestra los términos especiales que son de nuestro interés para representar los patrones que asumimos para reasentar las reglas en un formato informal:

Término	**Descripción**
<determinante>	El determinante del sujeto, ejemplo: el, cada, uno, una, etc.

<sujeto>	Una entidad reconocible del negocio, como es un objeto del negocio visible en el Modelo de Hechos, un nombre de un roll, o una propiedad de un objeto. La entidad puede ser cualificada por otros elementos descriptores, tales como la existencia en un estado particular, relacionada con una aplicabilidad específica de la regla con cierta precisión.
<características>	El comportamiento del negocio que debe tomar lugar o la interrelación que debe ser establecida
<hecho>	Una interrelación entre términos identificables en el Modelo de Hechos. La interrelación puede ser cualificada por otros elementos descriptivos en orden de especificar la aplicabilidad de la regla precisamente

3.3.2 Uso del patrón de restricción básica

Analizando la regla r0 se encuentra que el patrón que más se ajusta es el de Restricción básica,

r0: Todo aquel que sea *Posible Receptor* es asociado a un grupo de *Posibles Donantes.*

Este patrón es el más común de los patrones de reglas de negocio, establece una restricción sobre un sujeto de una regla y su estructura es:

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

[(si | a menos que) <hecho>].

otra variante de este patrón sería:

<determinante> <sujeto> puede (<característica> solo si <hecho>) |

(no puede <característica>).

Aplicando el patrón de restricción básica a *r0* se obtiene la siguiente sentencia:

R1: Un *Posible Receptor* debe ser asociado al menos a un *Posible Donante*

Dado que las reglas de negocios se construyen basadas en los hechos del negocio, dichos hechos son reflejados en el Modelo de Hechos. Este modelo debe ser capaz de darle soporte a todas las reglas de negocio, o sea, no puede existir una regla que haga referencia a términos o hechos que no estén reflejados en el Modelo de Hechos.

La figura 3.1 muestra parte del Modelo de Hechos derivado a partir de la sentencia de regla de negocio *R1.*

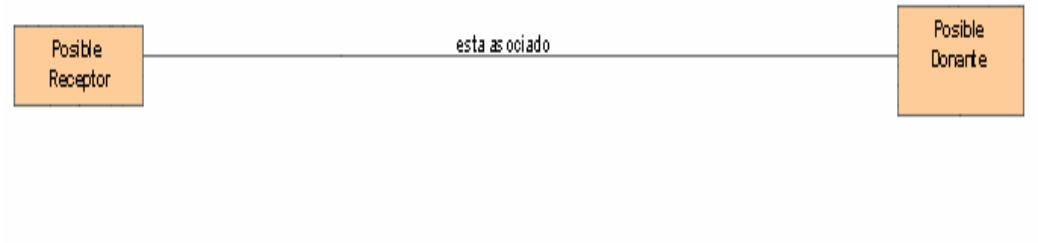


Figura 3.1.Modelo de Hechos asociado a R1

3.3.3 Uso del patrón de enumeración

Muchas veces en el Modelo de Hechos es necesario representar las diferentes categorías que conforman a un determinado término. Un ejemplo de un término que necesite ser categorizado es el *Seguimiento para Receptor*, el cual se puede clasificar en dos tipos: *Seguimiento Intrahospitalario* o *Seguimiento Extrahospitalario* por lo que es necesario diferenciarlos en el Modelo de Hechos quedando de la forma en que se muestra en la figura 3.2. Según el patrón de enumeración dado por Morgan (Morgan, 2002), es posible establecer al rango de valores que legalmente puede tomar un término, en este caso el *Seguimiento para Receptor.* El patrón de enumeración sigue esta sintaxis:

< determinante > < resultado > debe ser elegido desde la siguiente lista [abierta | cerrada]:
<lista de valores>

De esta forma se puede escribir la regla descrita en párrafos anteriores como sigue:

R2: El *Seguimiento para Receptor* debe ser elegido desde la siguiente lista cerrada:

- *Seguimiento Intrahospitalario*
- *Seguimiento Extrahospitalario*

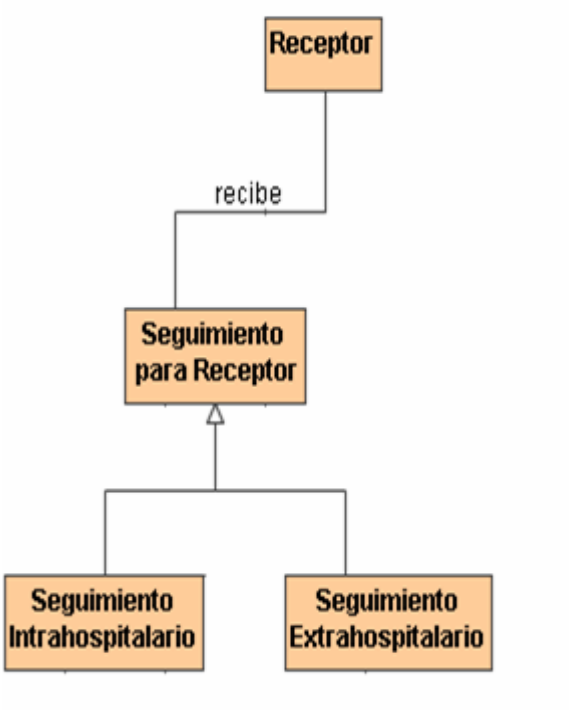


Figura 3.2. Categorías de Seguimiento para Receptor

Una de las reglas de negocio a la cual daría soporte esta parte del Modelo del Hechos sería:

R3**:** Todo *Receptor* recibe *Seguimiento para Receptor.*

El número de categorías en que puede dividirse un término en el Modelo de Hechos no tiene restricción alguna, esto significa que un término del negocio

puede presentar tantas categorías como sean necesarias para el propio negocio o dominio.

Para el caso del personal médico que trabaja en el hospital y se relaciona con nefrología se tiene la siguiente regla siguiente:

R4: Todo *Personal Médico* debe ser elegido desde la siguiente lista cerrada:

- *Nefrólogo*
- *Urólogo*
- *Cardiólogo*
- *Cirujano*
- *Estomatólogo*
- *Anestesiólogo*
- *Enfermero*
- *Asistente*
- *Laboratorista*

En la figura 3.3 se muestra la representación de la regla *R4* para dar las diferentes categorías del *Personal Médico* para de trasplante renal.

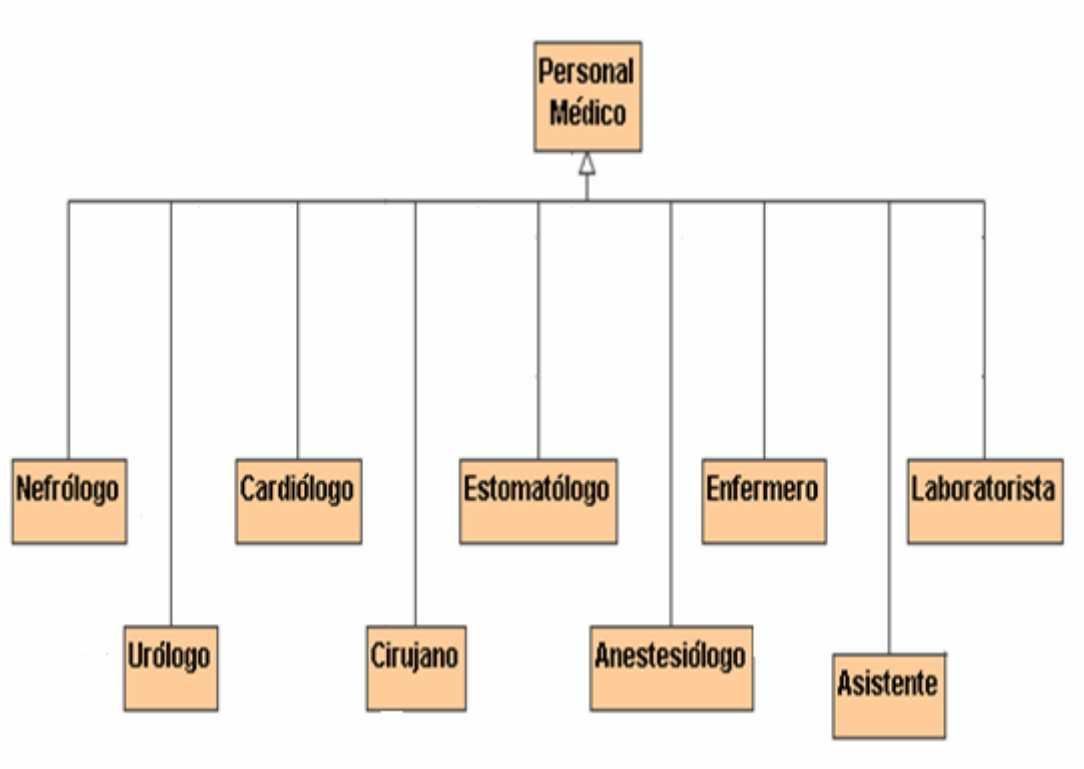


Figura 3.3. Categorías de Personal Médico.

3.3.4 Representación de reglas complejas

Durante el proceso de transformación de las reglas de negocio en lenguaje natural a un lenguaje semiformal ajustado a un determinado patrón, es muy común encontrarse con reglas que necesiten ser descompuestas en más de una sentencia dando como resultado nuevas reglas. Esto viene dado porque las RN escritas en lenguaje natural en ocasiones pueden ser divididas en más de una regla sin que ocurra pérdida de información, o sea, no cumplen con la propiedad de que toda regla de negocio debe ser atómica. A continuación se muestra un ejemplo donde se presenta tal situación:

La regla escrita en lenguaje natural sin ajustarse a ningún patrón sería:

r2: Todos los *pacientes* con *diagnostico IRCT* (Insuficiencia Renal Crónica Tratable) tienen como opción de *Tratamiento: Tratamiento Dialítico o*

Trasplante de Riñón (tiene que tener un *Donante Vivo* o un *Donante con Muerte Encefálica).*

Como se puede observar *r2* no es una regla atómica. Dicha regla se puede descomponer en las sentencias *R2.0, R2.1 y R2.2* sin que ocurra pérdida de información, de esta manera se obtienen tres sentencias atómicas y compactas en un lenguaje semiformal. Notar que el patrón usado para este caso fue el de restricción básica:

R2.0: Un *Paciente* con *Diagnóstico IRCT* puede ser tratado con *Trasplante de Riñón* solo si se tiene un *Donante Vivo.*

R2.1: Un *Paciente* con *Diagnóstico IRCT* puede ser tratado con *Trasplante de Riñón* solo si se tiene un *Donante con Muerte Encefálica.*

R2.2: Un *Paciente* con *Diagnóstico IRCT* puede ser tratado con *Tratamiento Dialítico.*

En la Figura 3.4 muestra como se representan las reglas *R2.0, R2.1 y R2.2* en el Modelo de Hechos.

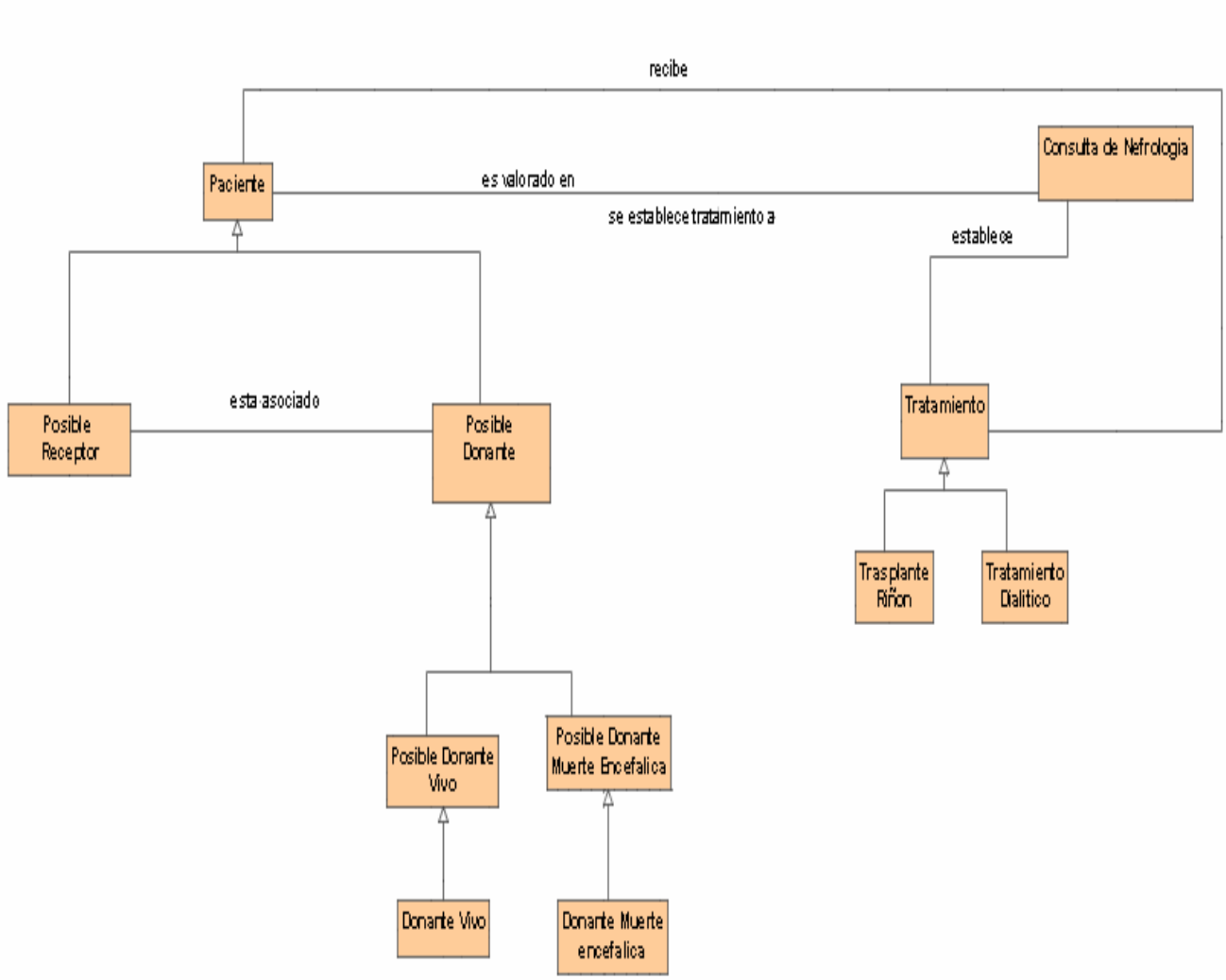


Figura 3.4. Modelo de Hechos asociado a *R2.0, R2.1 y R2.2.*

3.3.5 Otras reglas representadas en el Modelo de Hechos

A continuación se muestran diferentes conjuntos de reglas de negocios pertenecientes al trasplante renal con sus correspondientes representaciones en el Modelo de Hechos.

Conjunto de reglas asociadas a la figura 3.5:

- Un *Posible Receptor* debe ser asociado al menos a un *Posible Donante.*
- El *Estado General* de un *Posible Receptor* es determinado por el *Equipo de Trasplante.*
- Un *Receptor* puede presentar *Rechazo Hiperagudo* provocándole *Atrofia Renal.*
- El *Rechazo Agudo Acelerado* tiene como consecuencia la *Atrofia Renal.*
- El *Equipo de Trasplante* realiza la *Operación de Trasplante* al *Receptor.*
- *Posible Donante* y *Posible Receptor* son valorados por el *Equipo de Trasplante*

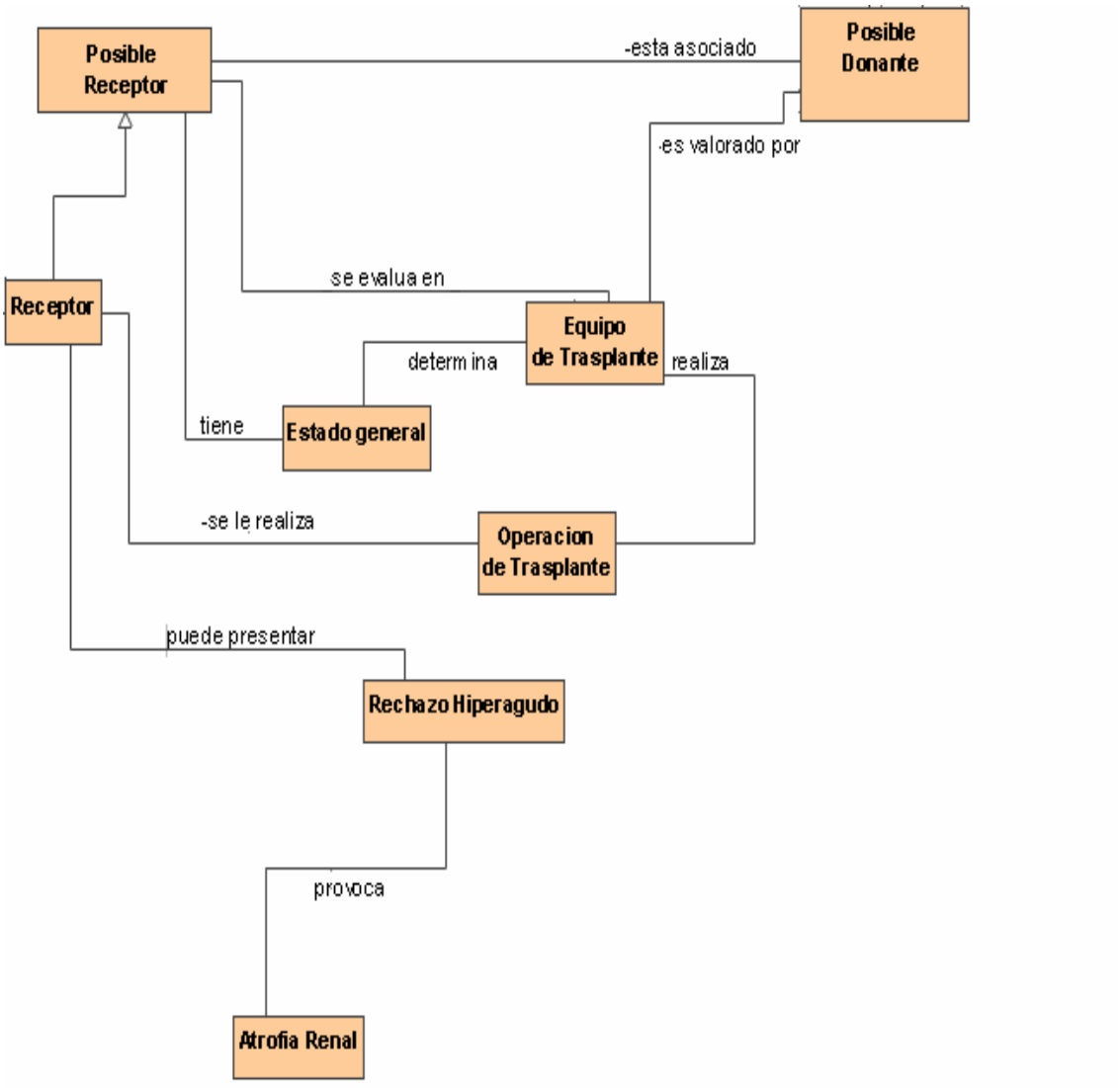


Figura 3.5. Modelo de hechos.

Conjunto de reglas asociadas a la figura 3.6:

- Todo *Receptor* recibe *Seguimiento para Receptor.*
- *Seguimiento para Receptor* puede ser *Seguimiento Intrahospitalario* o *Seguimiento Extrahospitalario.*
- A un *Posible Receptor* con *Historia sugerente de Enfermedad Vascular Periférica* se le debe realizar *Evaluación Radical.*
- *Paciente* puede tener *Diagnóstico* de tipo *IRCT.*
- Un *Paciente* puede ser un *Posible Receptor.*
- La *Lista de Espera* es integrada por cada *Receptor.*
- A todo *Receptor* se le realiza *Protocolo de Estudio para Trasplante.*

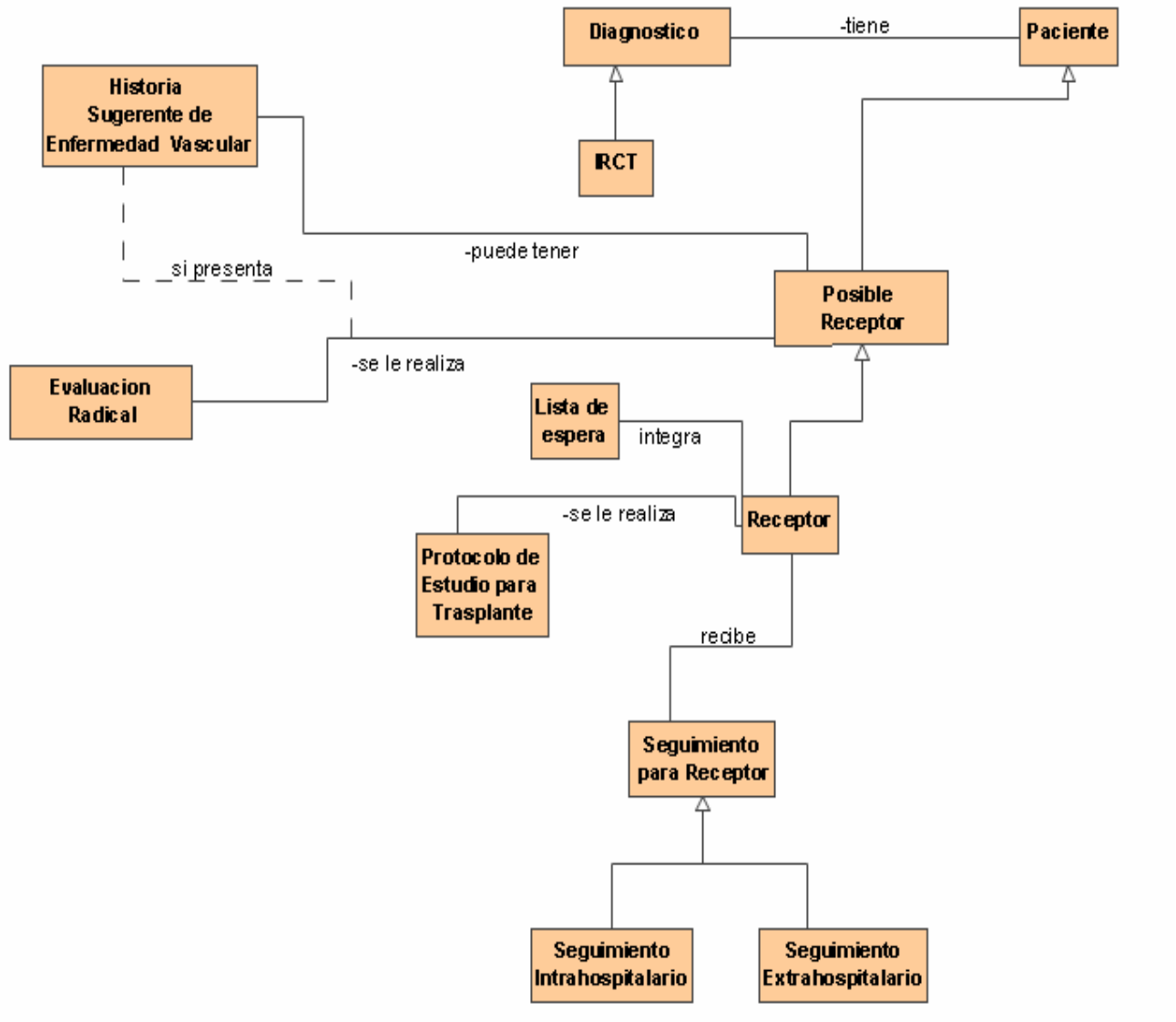


Figura 3.6. Modelo de Hechos.

Conjunto de reglas asociadas a la figura 3.7:

- Un *Posible Donante* puede presentar *Contraindicaciones Absolutas.*
- Un *Posible Donante* puede presentar *Contraindicaciones Relativas.*
- Un *Donante Vivo* recibe *Baño Higiénico* antes de la *Operación de Trasplante.*
- El *Tratamiento* para *Paciente* es establecido por la *Consulta de Nefrología. (*véase también figura 3.4*)*
- *Tratamiento* puede ser *Tratamiento Dialítico* o *Trasplante de Riñón.*
- Al *Posible Donante* se le tiene que realizar *Protocolo de Estudio para Donante*

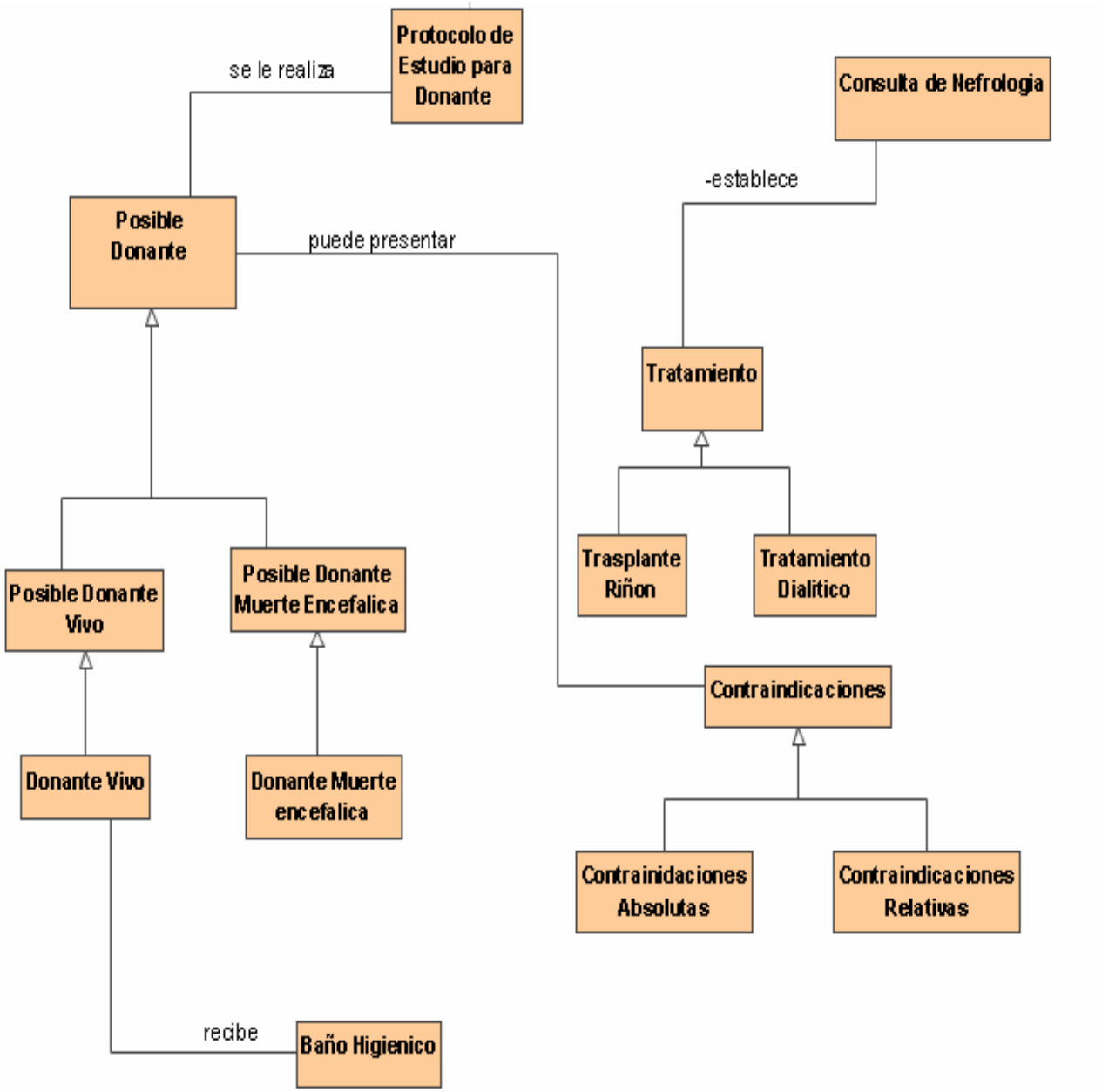


Figura 3.7. Modelo de Hechos.

CAPÍTULO IV. IMPLEMENTACION DEL MODELO DE HECHOS PARA TRASPLANTE RENAL.

4.1 Arquitectura del Editor de Reglas

El editor de reglas de negocio esta integrado de varios módulos, véase en la figura 4.1, tiene como función fundamental captar las reglas de negocios, permitiendo que sean introducidas por un usuario del negocio. El editor de reglas permite la validación sintáctica, semántica y realiza el análisis de conflictos e inconsistencias, quedando así conformado el repositorio de reglas de negocio. Otros módulos se encargan del resto del procesamiento, lográndose el objetivo final que es la generación automática de reglas de negocios, que es el objetivo final de este proyecto.

Dentro del editor de reglas resulta de especial interés para el presente trabajo el analizador sintáctico, encargado de realizar la edición y el análisis sintáctico de las reglas introducidas por el usuario en un lenguaje cercano al natural, haciendo uso de patrones previamente definidos, referidos en capítulos anteriores.

Estas reglas analizadas sintácticamente son depositadas en un fichero temporal en formato texto**,** cumpliendo con una determinada estructura, los cual no forma parte de os objetivos del presente trabajo. El analizador semántico hace uso de la información como entrada para realizar su trabajo; además se apoya en el Modelo de Hechos para validar la semántica de la regla. El Modelo de Hechos está implementado mediante una ontología haciendo uso del software Protégé.

El Modelo de Hechos se debe tratar de forma evolutiva pues está sujeto a transformaciones o incorporación de nuevos hechos en dependencia del comportamiento del negocio. Se debe proponer una interface para los desarrolladores de software que facilite la edición del Modelo de Hechos,

evitando así el uso del software Protégé siempre que se requieran modificaciones del modelo.

El módulo encargado de la validación de conflictos e inconsistencias de reglas, evita ambigüedades, chequea que no existan reglas repetidas o contradictorias garantizando la integridad del repositorio de reglas.

Luego de que se validen correctamente las reglas de negocio junto con la creación o actualización del repositorio de reglas, entonces es que pasan a ser utilizadas por otros módulos que componen al sistema.

En la figura 4.1 se muestra de forma gráfica la arquitectura que tendría el sistema en general y el rol que juega el Editor de Reglas de Negocio en dicho sistema.

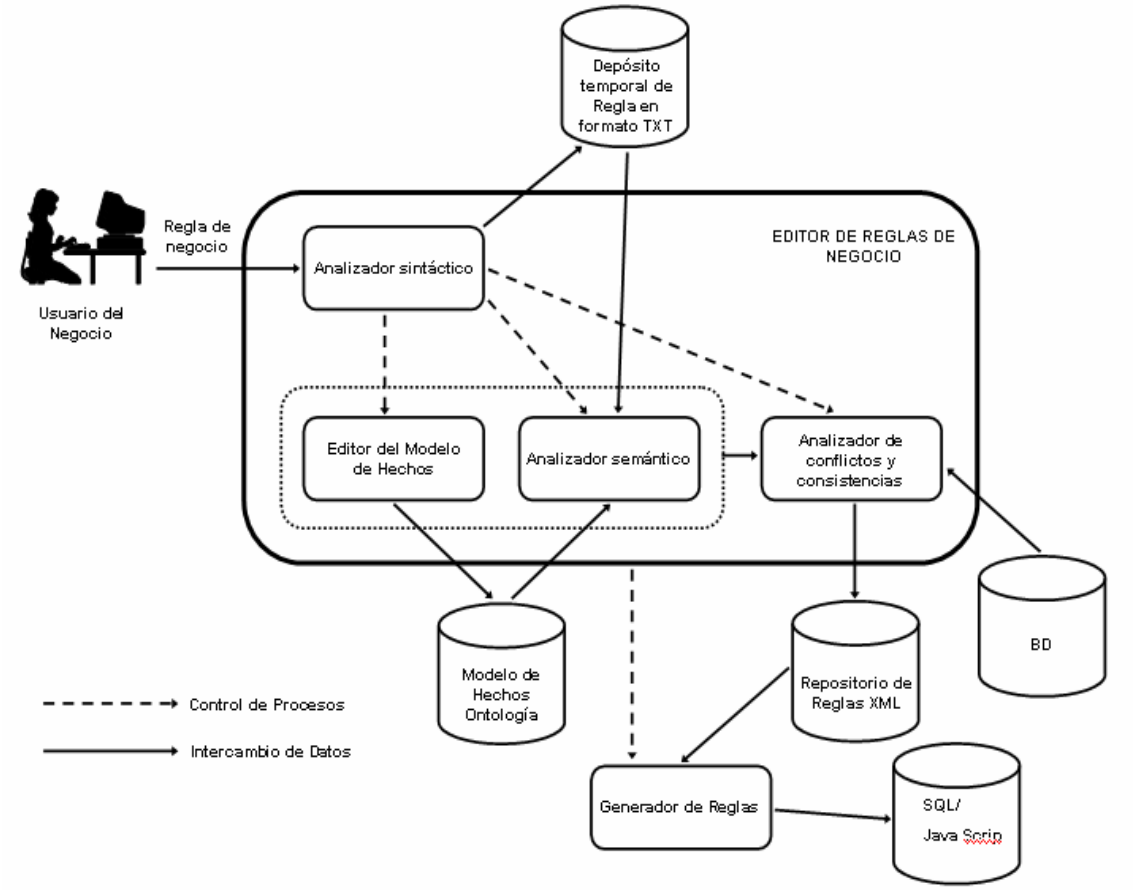


Figura 4.1 Arquitectura del Editor de RN

4.2 Editor Semántico de reglas

Como se mencionó en el epígrafe anterior, para poder validar semánticamente las reglas de negocio es conveniente desarrollar un aplicación capaz de procesar el archivo XML que contiene la ontología que representa al Modelo de Hecho junto con la terminología del negocio, en nuestro caso de estudio, los términos que intervienen en el negocio vinculados al trasplante renal. Esta aplicación debe ser capaz de contestar a varias interrogantes para lograr una buena validación de las reglas de negocio tales como:

¿El término *A* se relaciona con el término *B*?

¿Existe la relación *X* en al cual intervienen los términos *A, B, etc.*?

¿El término *A* es sinónimo del término *B*?

¿El término *A* es equivalente al término *B*?

¿Cuáles son los términos equivalentes al término *B*?

Para darles respuestas a estas interrogantes, el editor semántico se apoyaría en la ontología del Modelo de Hechos.

4.2.1 Comunicación entre el Editor Sintáctico y Semántico

Para la validación sintáctica y semántica de las reglas de negocio es necesario que exista algún tipo de comunicación entre ambos analizadores (sintáctico y semántico). El analizador semántico se apoya en el Modelo de Hechos para la validación, esto se hace a través de un archivo XML generado por el software Protégé en el cual se exporta la ontología de la Modelo de Hechos implementada mediante dicho software.

4.2.2 Implementación del Modelo de Hechos mediante el Protege

El Modelo de Hechos para trasplante renal ha sido implementado en forma de ontología haciendo uso del software Protégé. A continuación se muestran de manera general algunos de los aspectos más importantes de dicha implementación.

Partiendo de que el Modelo de Hechos muestra, a grandes rasgos, los diferentes hechos del negocio y estos hechos a su vez están formados por relaciones entre términos, es por ello que se define una clase *Término* de la cual heredarían dos subclases: *Término Común* y *Término de negocio*, ya que existen términos que tienen un significado específico para el negocio y términos con un significado común independiente del contexto del negocio.

Otra clase definida es *Significado*, pues la ontología no solo representa el Modelo de Hechos, también contiene el vocabulario del negocio, o sea, todos los términos del dominio junto con sus respectivos significados, sinónimos, etc. Las otras clases que componen la ontología que representa el Modelo de Hechos para trasplante renal son las clases *Hecho* y *Rol***.** En la figura 4.2 se muestra una vista del Protégé, específicamente de la pestaña *"Classes",* la cual es utilizada para visualizar y editar las clases. Las clases y sus relaciones jerárquicas se visualizan en el *"Class Browser",* donde se puede apreciar al árbol formado por las clases que componen la ontología que representa el Modelo de Hechos para trasplante renal.

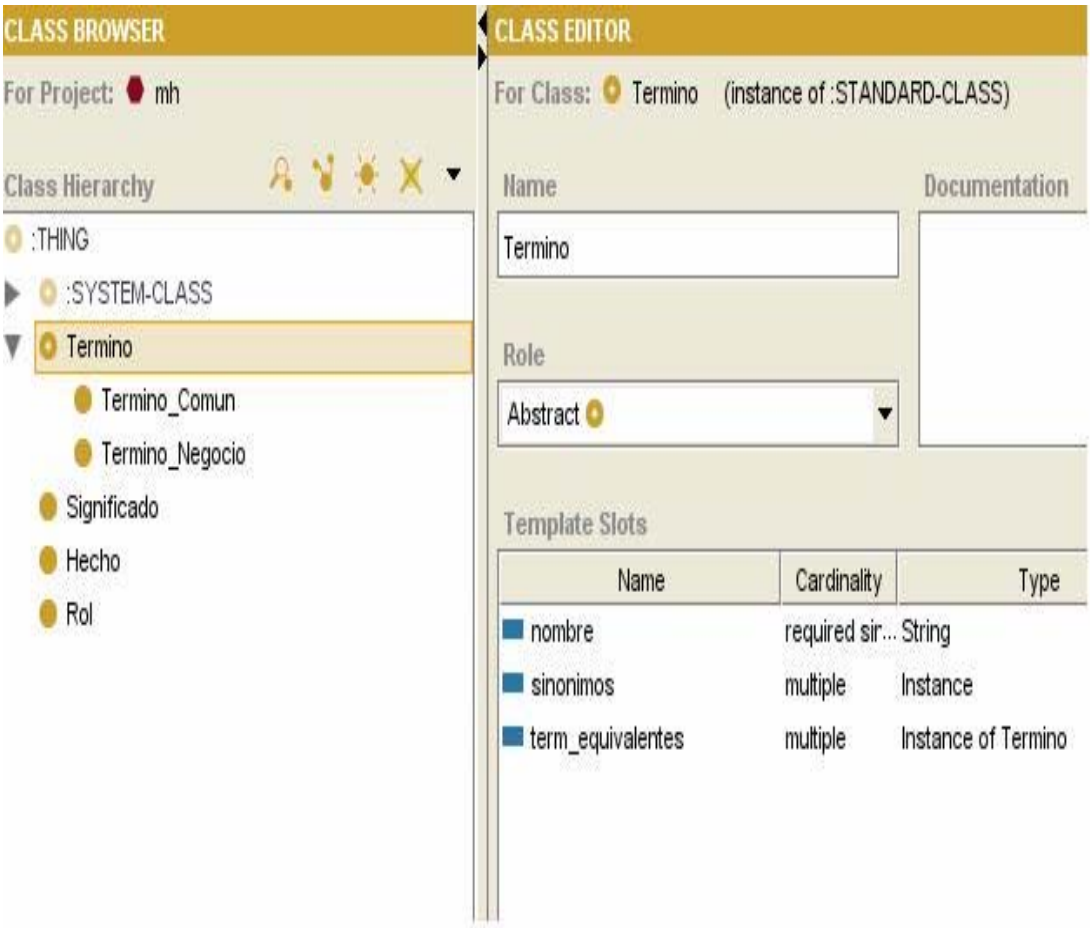
Figura 4.2. Clases de la ontología que representa el Modelo de Hechos para trasplante renal.

En una ontología representada en Protégé las clases se organizan de forma jerárquica, mediante la relación “subclase de” (o “es un”). Por ejemplo: dentro de la clase *Término*, existen subclases que se corresponden con distintos subconjuntos (tipos) de términos: *Término_Comun*, *Término_Negoico*.

La clase *Término* presenta varios atributos o *"slots"* los cuales son descritos a continuación:

- *nombre:* nombre del término. *"slot"* de tipo *String.*
- *sinónimos:* se refiere a los sinónimos que pueda tener un término en específico. *"slot"* de tipo *Instance*, en este caso se refiere a instancias de la clase *Término.*
- *Term_equivalentes:* se refiere a los términos equivalentes que pueda tener un término en específico, o sea, términos que no necesariamente son sinónimos pero que pueden ser sustituidos el uno por el otro. *"slot"* de tipo *Instance*, en este caso se refiere a instancias de la clase *Término.*

En la figura 4.3 se muestra como quedaría la clase *Término* implementada en el Protégé, notar que en el *"class editor"* se muestra un campo llamado *Role* el cual se refiere al tipo de clase (abstracta o concreta), en este caso la clase *Término* es abstracta ya que las instancias serán de las subclases *Término_Comun* o *Término_Negocio.*



ç

Figura 4.3. Clase Término

La clase *Significado* es definida debido a que puede darse el caso en que dos o más términos tengan el mismo significado por lo que dicha clase tendría los atributos o *"slots"* siguientes:

- *término:* nombre del término asociado. *"slot"* de tipo *Instance*, en este caso se refiere a instancias de la clase *Término.*
- *sign_general:* Significado general que presenta el término al cual se hace referencia en el atributo anterior. *"slot"* de tipo *String.*

En la Figura 4.4 se muestra como quedaría implementada la clase *Significado* en el Protégé.

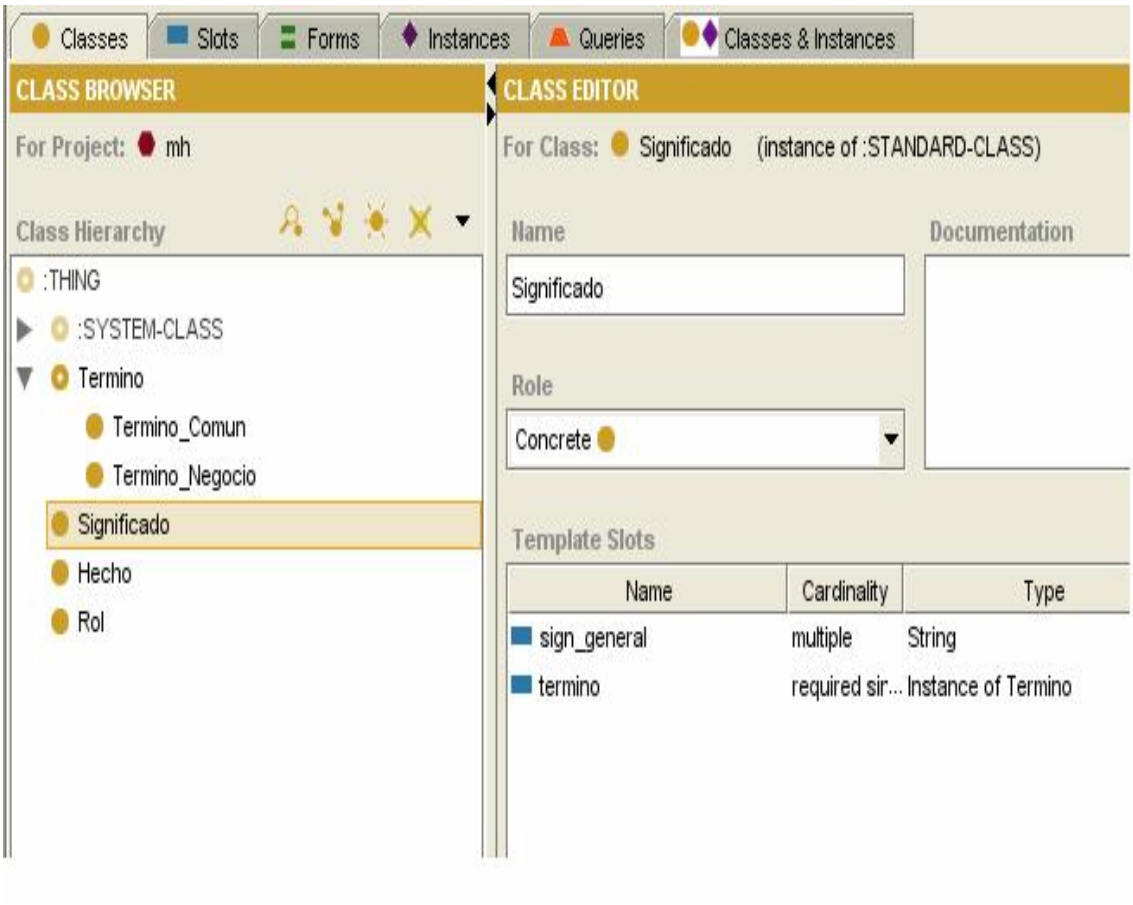


Figura 4.4. Clase *Significado*

Cada instancia de la clase *Hecho* muestra los términos que intervienen en cada hecho junto con los roles que juegan, teniendo en cuenta que en un hecho puede existir mas de un rol. La clase *Hecho* tiene los siguientes atributos:

- *términos_relacionados:* términos que intervienen en el hecho. *"slot"* de tipo *Instance,* en este caso se refiere a instancias de la clase *Término*.
- *rol:* viene siendo la manera en que se relacionan dos términos, esta relación varía en dependencia de la dirección en que se trate. *"slot"* de tipo *Instance*, en este caso se refiere a instancias de la clase *Rol.*

En la figura 4.5 se muestra como quedaría implementada la clase *Hecho* en el Protégé

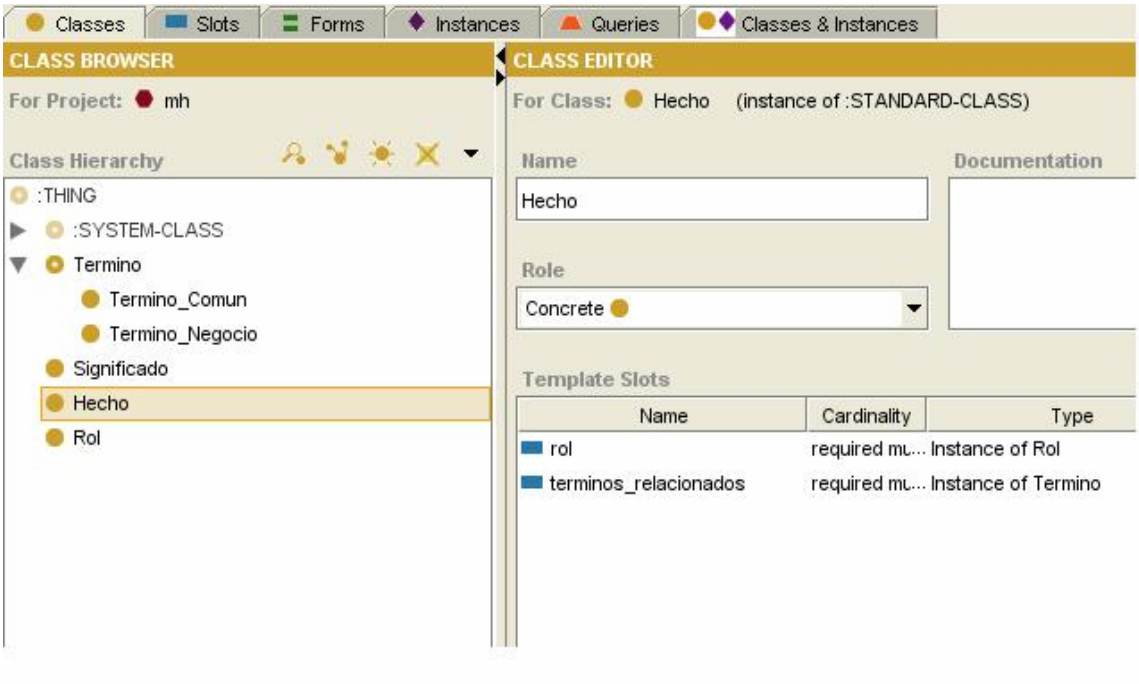


Figura 4.5. Clase *Hecho.*

Como en un mismo hecho puede haber más de un rol, es por eso que se crea una clase *Rol*, la cual tiene los siguientes *"slots"* o atributos:

- *nombre:* nombre del rol como tal, o sea, el nombre de la relación que presenta ambos términos. *"slot"* de tipo *String.*
- *sign_general:* significado que tiene para el negocio determinado rol. *"slot"* de tipo *String.*
- *Término_de_partida:* término del cual se parte a la hora de ver una relación entre términos, o sea, esto se hace con el objetivo de establecer un sentido o una dirección entre ambos términos ya que en una misma relación pueden haber dos instancias de la clase *Rol*, una para cada dirección. *"slot"* de tipo Instance, en este caso se refiere a instancias de la clase *Término.*

- *Término_2:* se refiere al segundo término de una relación. *"slot"* de tipo Instance, en este caso se refiere a instancias de la clase *Término.*

En la figura 4.6 se muestra como quedaría implementada la clase *Rol* en el Protégé.

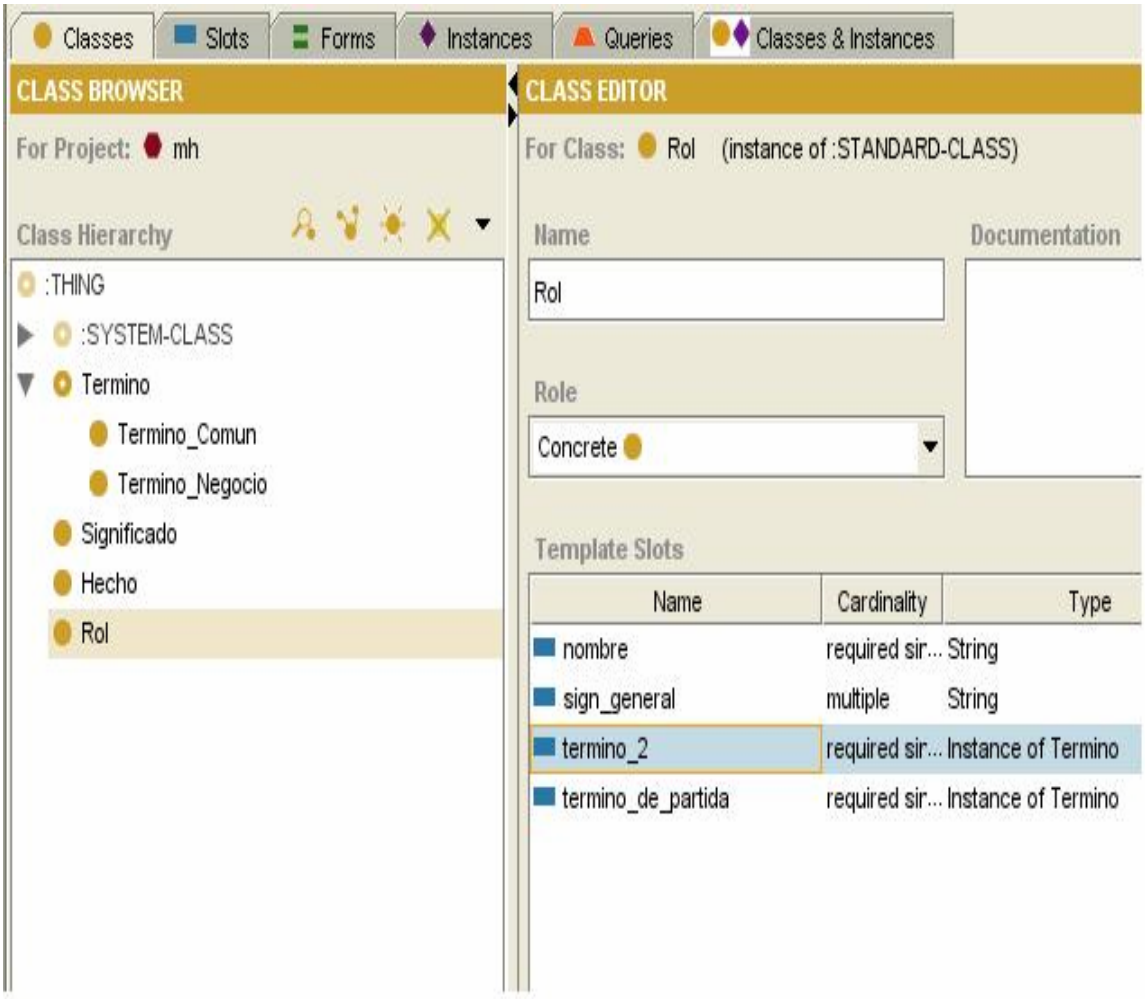


Figura 4.6. Clase *Rol.*

Los objetos del dominio se representan en Protégé como instancias de las clases definidas en la ontología. En realidad, las instancias son la verdadera base de conocimiento, y la estructura de clases, propiedades y relaciones forman el esquema (o la estructura) del conocimiento.

Las instancias se editan en la pestaña *Instances*, que tiene tres secciones:

- *Class Browser,* donde se muestra la jerarquía de clases pero ahora indicando entre paréntesis el número de instancias definidas para cada clase concreta.

- *Instance Browser,* donde se muestran las instancias de una clase seleccionada en navegador de clases. Para identificar cada instancia se visualiza el valor de uno de sus *"slots"*. El *"slot"* utilizado para visualizar la instancia no es fijo y es posible decir a Protégé qué *"slot"* se utilizará en el navegador de instancias (es denominado *Display slot)*.

- *Instance Editor*, donde se muestran en un formulario los campos editables de una instancia (cada campo está asociado a un *"slot"*). El formulario para editar instancias de una clase es también configurable (a través de la pestaña *Forms)*

En la figura 4.7 se muestran algunas instancias de la clase *Término_Negocio***.**

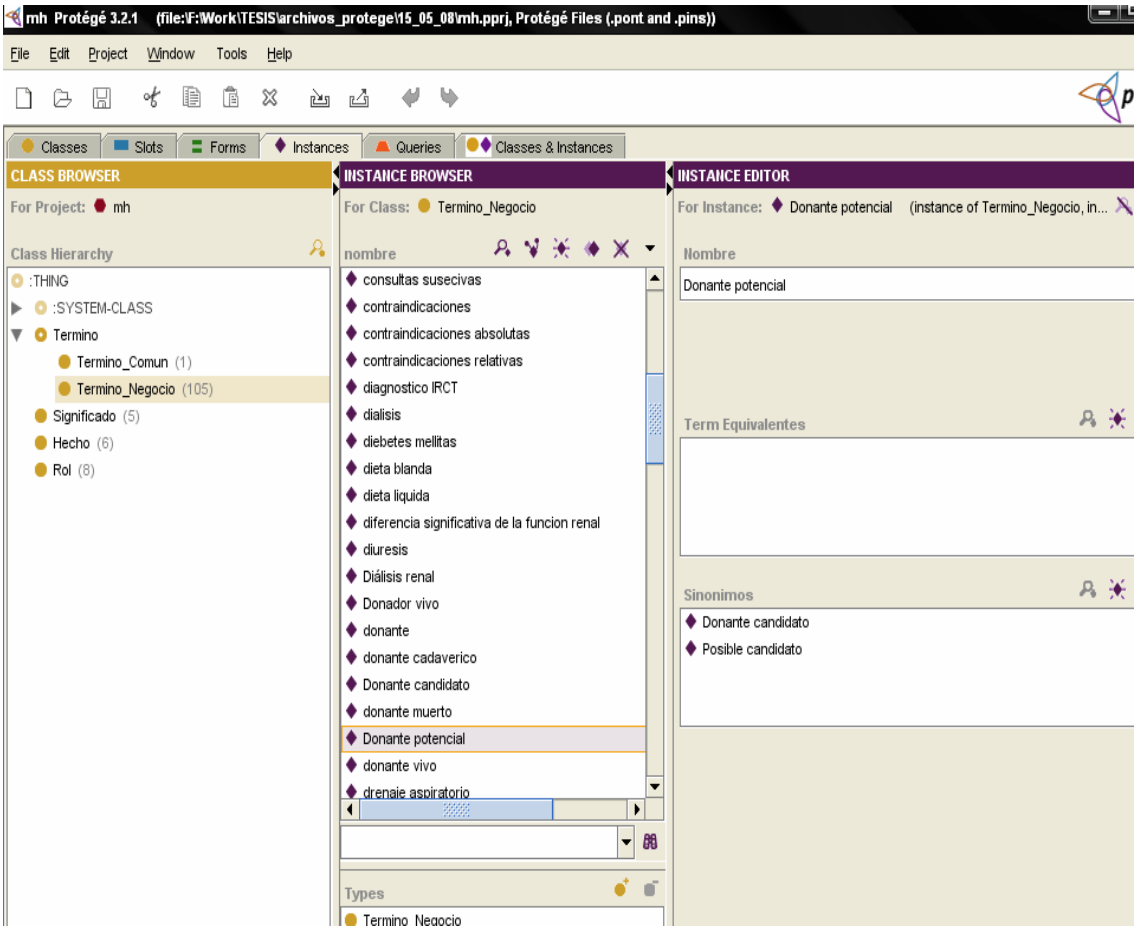


Figura 4.7. Instancias de la clase Término_negocio.

Como se ha visto en páginas anteriores, para cada clase se pueden definir *"slots"* (que se corresponden con las propiedades de los objetos de la clase) que sirven para definir la estructura de las instancias de esa clase.

Los *"slots"* tienen varias características (que pueden editarse desde la pestaña *"slots"* [Figura 4.8b]), aunque las fundamentales son:

- *Cardinality*: que puede ser *single* o *multiple*, para indicar que el *"slot"* admite un único valor o una lista de valores
- *Type*: que se refiere al tipo de valor que almacena un *"slot"*. Puede ser "*boolean", "integer", "float", "string", "instance"* (si los valores permitidos

para el *"slot"* son instancias de alguna clase), *"class"* (si los valores permitidos pueden ser directamente clases) o "*any"* (cualquier tipo, o lo que es lo mismo, el tipo vacío).

- *Domain*: el dominio de un *"slot"* es la clase (o clases) para la que se ha definido.

Por ejemplo: si los objetos de la clase *Término_Comun* tienen un *"slot" nombre,* con "*cardinality=single"* y *"type=string",* esto significa que todas las instancias de *Término_Comun* tendrán un nombre que sólo permitirá una cadena como único valor.

En la figura 4.8a se muestra el ambiente que provee el Protégé para la edición de los *"slots"* referentes a una clase, y en la Figura 4.8b aparece la pestaña *"slots"* mostrando todos los *"slots"* definidos en la ontología.

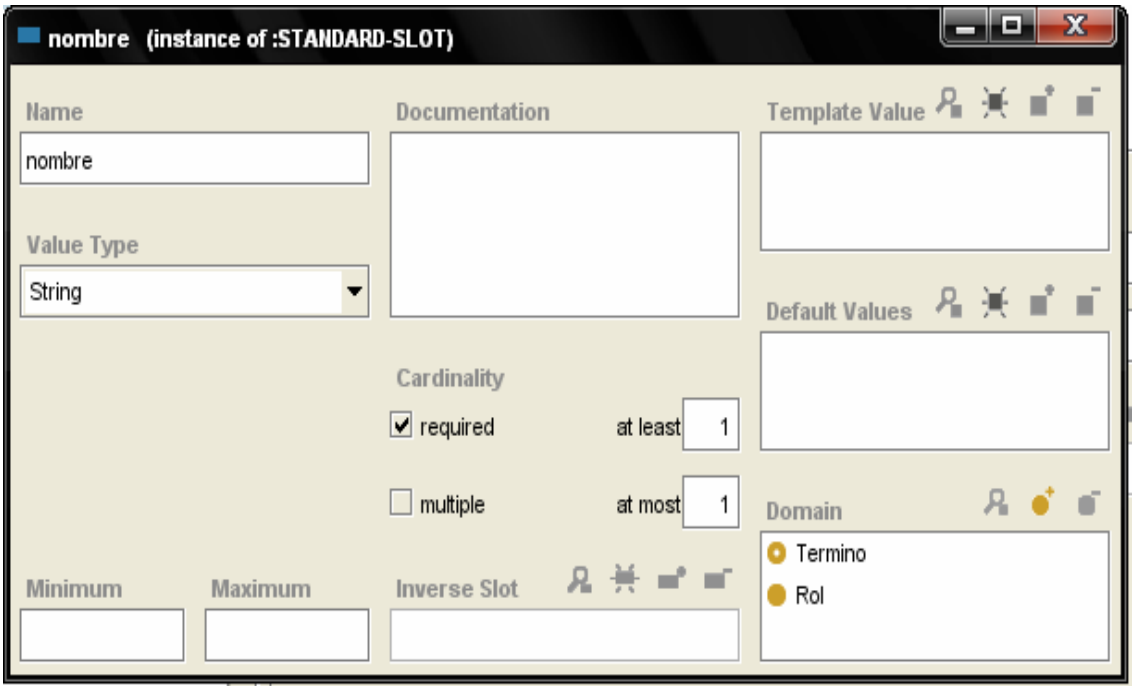


Figura 4.8a. Características de los *"slots"*.

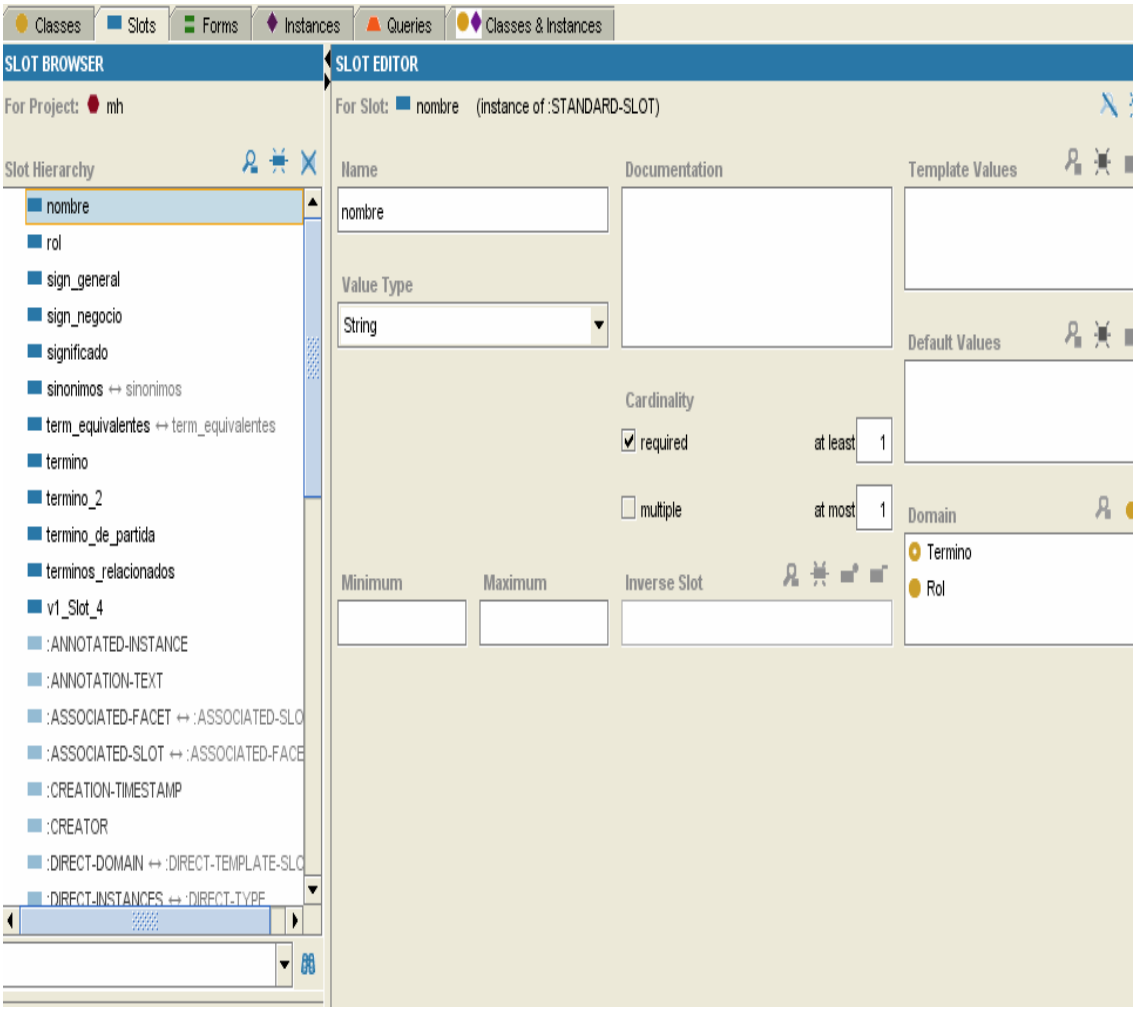


Figura 4.8b. Pestaña *"slots"*

A la hora de definir un nuevo *"slot"* se debe tenerse en cuenta que un valor de un *"slot"* puede depender del valor de otro. Por ejemplo, si el término del negocio *Donante potencial* tiene como sinónimo a *Posible donante*, entonces *Posible donante* también tiene como sinónimo a *Donante potencial*. Para manejar esta situación se pueden definir propiedades para los *"slots"*, en este caso la propiedad sería "*inverse-slot*" (Figura 4.9).

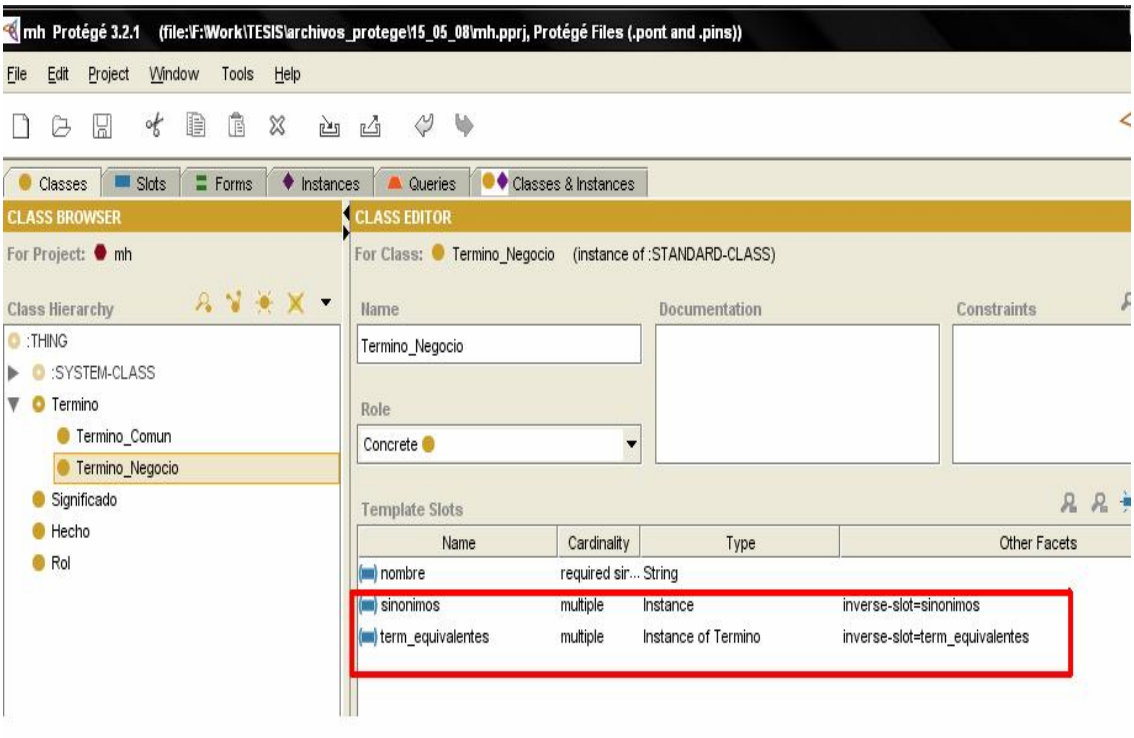


Figura 4.9. *"slot"* inverso.

Esto permite que cuando se introduzca *Posible Candidato* como sinónimo en el objeto *Donante Potencial* (Figura 4.10a), automáticamente se adiciona *Donante Potencial* en el *"slot"* sinónimo de la instancia *Posible Candidato* (Figura 4.10b). Lo mismo sucede para el *"slot"* term_equivalentes.

Figura 4.10a

Figura 4.10b

A continuación se hará una pequeña descripción de cómo adicionar un nuevo hecho a la ontología. Para ello tomaremos como ejemplo la inserción del hecho (Figura 4.11):

Figura 4.11. Hecho: *Receptor* puede presentar *Rechazo Hiperagudo*

Como se puede apreciar, los términos que se relacionan en el hecho mostrado en la figura 4.11 son *Receptor* y *Rechazo Hiperagudo*. El *rol* que juega el *receptor* en este hecho es que *puede presentar rechazo hiperagudo*. Para introducir esto en la ontología a través del Protégé, es necesario crear una

nueva instancia de la clase *Hecho* donde receptor y rechazo hiperagudo son los términos relacionados (ver Figura 4.12).

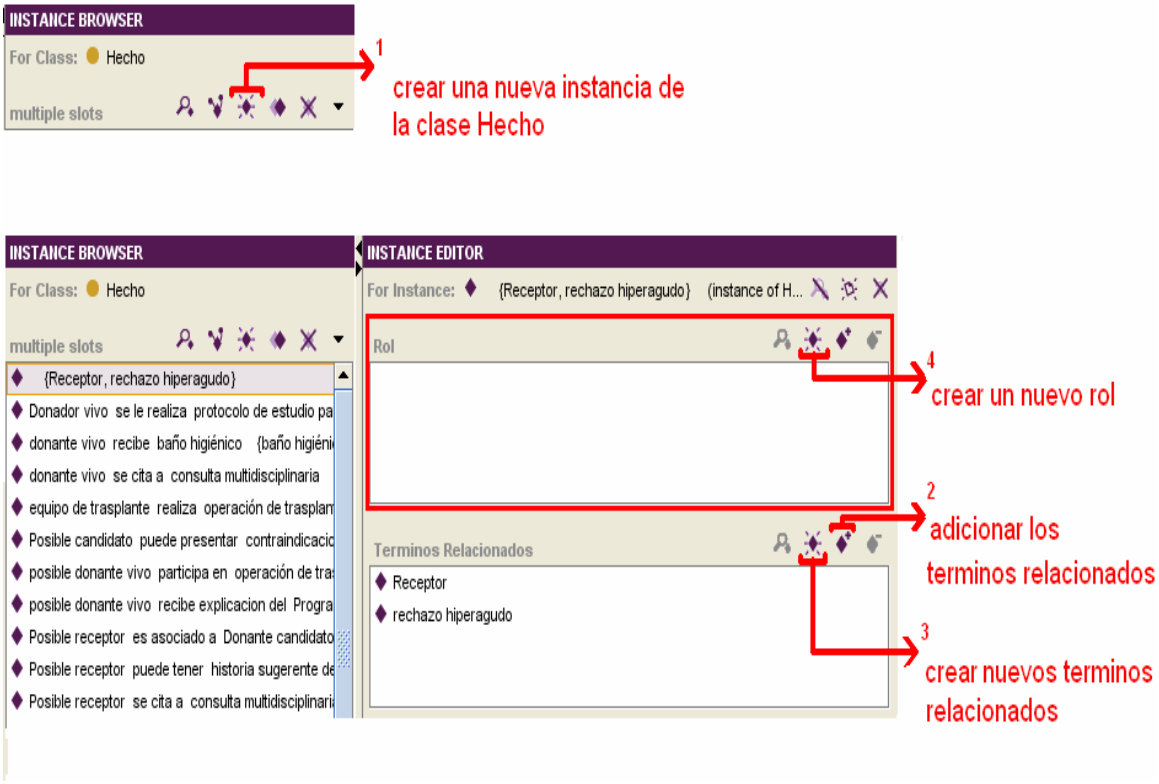


Figura 4.12. Edición del Modelo de Hechos mediante el Protégé.

En el paso 4(figura 4.12), seleccionar la opción de crear un nuevo rol, se refiere a la creación de una nueva instancia de la clase *Rol* vinculada al hecho que se este introduciendo en ese momento, en este caso al hecho descrito en la figura 4.11. Para la introducción de un nuevo rol es necesario especificar el término del cual se parte *(receptor),* el segundo término que interviene en el hecho (*rechazo hiperagudo*) y el nombre del rol, o sea, el nombre de la relación que existe entre ambos términos en dependencia del sentido en que se trate *(puede presentar).* Además se puede especificar, de manera opcional, el significado general para cada rol mediante el *"slot" Sign General*. El rol mostrado a continuación (Figura 4.13) es: *receptor* puede presentar *rechazo Hiperagudo.*

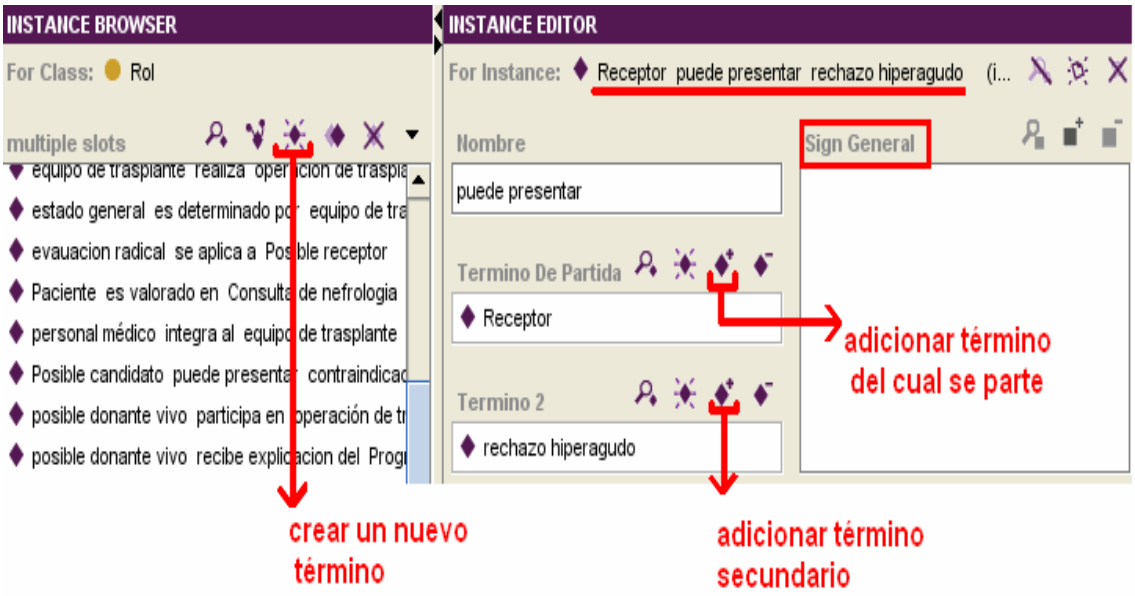


Figura 4.13. Como crear instancias de la clase Rol mediante el Protégé.

E hecho de la figura 4.11 quedaría como se muestra en la figura 4.14:

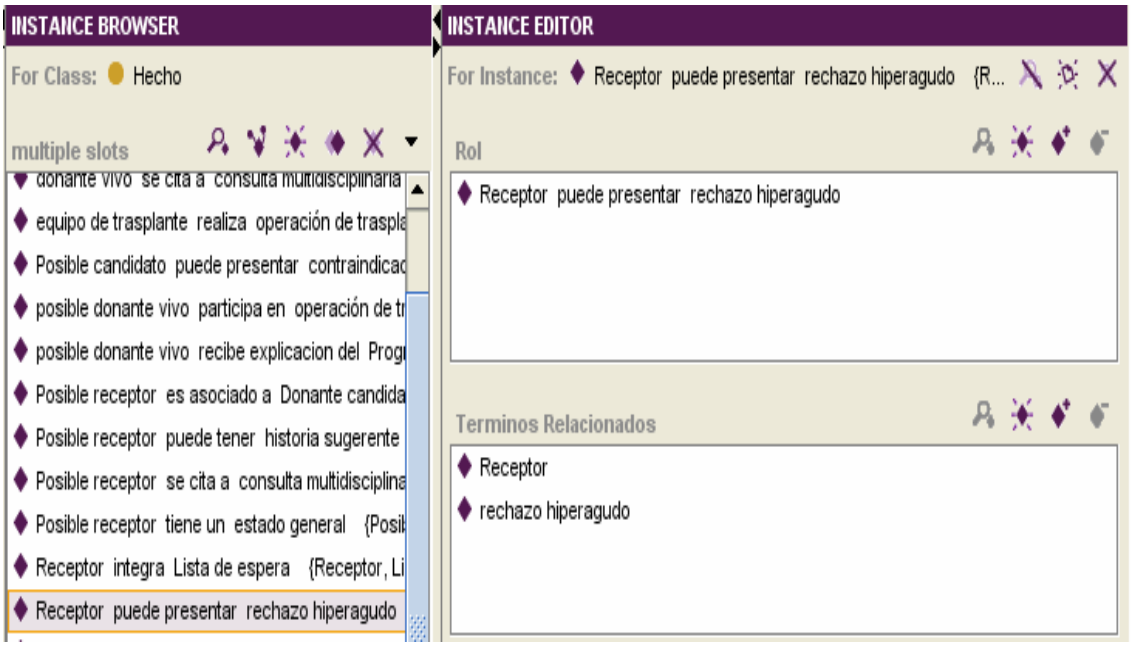


Figura 4.14.Hecho: Receptor puede presentar rechazo hiperagudo.

Se ha visto de manera general gran parte del proceso de implementación de la ontología que representa el Modelo de Hechos para trasplante renal

haciéndose énfasis en la estructura definida para dicha ontología, así como en el manejo del Protégé para la creación de clases, *"slots"*, instancias, etc.

Conclusiones

El presente trabajo surgió bajo la necesidad de crear una herramienta para facilitar la validación semántica de las Reglas de Negocio para trasplante renal, a través del mismo se han arribado a las siguientes conclusiones:

- Se diseñó el Modelo de Hechos para trasplante Renal junto con el vocabulario asociado creándose las bases para la implementación de una Ontología para su representación. Se vieron además, las ventajas que proporciona dicho modelo para la validación de las reglas de negocios.

- Se logró implementar una ontología que representa el Modelo de Hechos para trasplante renal mediante la utilización del software Protégé. Además esta ontología contiene el vocabulario del negocio en el cual se muestran los términos por categorías (términos comunes y términos del negocio). Finalmente el Modelo de hechos se pudo exportar en formato XML para ser usado por el editor de Reglas de Negocio.

Recomendaciones

- Desarrollar una aplicación que procese la ontología en formato XML y establezca una comunicación entre el Editor Sintáctico y el Modelo de Hechos favoreciendo la integración entre estos módulos.
- Crear un ambiente que permita realizar operaciones básicas sobre una ontología (adicionar o eliminar un término, adicionar o eliminar un hecho, etc.) permitiendo mayor acceso a usuarios no avanzados, eliminando así la dependencia al Protégé.

Referencias bibliográficas

ANTONIOU, G. T., K. BERNDTSSON, M. WAGNER, G. SPREEUWENBERG, S. (2004) A First-Version Visual Rule Language. *REWERSE, Report IST-*.

BAJEC, M. & MARJAN, K. (2006) Managing business rules in enterprises. *Faculty of Computer and Information Science,*. Ljubljana, University of Ljubljana.

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.

BARNE, M. & KELLY, D. (1997) Play by the Rules. *Byte (Special Report)*, 22, Nro 6**,** 98-102.

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

DATE, C. J. (2000) What Not How: The Business Rules Approach To Application Development. *Addison Wesley Longman,*.

GRUBER (1994) Toward Principles for the Design of Ontologies Used for Knowledge Sharing.

KOLBER, A. (2000) Defining Business Rules - What are they really?

MCGUINNESS, D. L. A. W., J. (1998) Conceptual Modeling for Configuration: A Description Logic-based Approach. Artificial Intelligence for Engineering Design, Analysis, and Manufacturing - special issue on Configuration.

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

NATALYA, F. & MCGUINNESS, D. L. (2005) Ontology Development 101: A guide to creating your first ontology.

ROSS, R. Business Rules from A –Z: What You Need to Know. *Business Rules Journal*.

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

ROSS, R. (2005) *Business Rule Concepts*.

ROSS, R. & LAM, G. Developing the Business Model. *Business Rules Journal*.

YOUDEOWEI, A. (1997) The B-Rule Methodology: A Business Rule Approach to Information Systems Development. *Department of Computation UMIST*, . Manchester, United Kingdom.