UNIVERSIDAD DE MURCIA

Departamento de Ingeniería de la Información y las Comunicaciones

TESIS DOCTORAL

Un Entorno para la Extracción Incremental de Conocimiento desde Texto en Lenguaje Natural

Rafael Valencia García

Enero 2005

Directores:

Dr. Rodrigo Martínez Béjar

Dr. Jesualdo Tomás Fernández Breis

Agradecimientos

A Mari.
A toda mi familia y amigos.
A los compañeros de investigación con los que he trabajado durante estos años.

ÍNDICES

ÍNDICE DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

La extracción de conocimiento a partir de textos en lenguaje natural es una tarea muy importante dentro de la inteligencia artificial y la ingeniería de conocimiento, ya que permite simplificar los procesos de adquisición de conocimiento de tal forma que los ingenieros de conocimiento pueden llegar a ser innecesarios para dicha actividad y el conocimiento puede ser extraído directamente por los expertos a partir dicho tipo de textos.

Además, la creciente necesidad de enriquecer la Web con grandes cantidades de ontologías que capturen el conocimiento del dominio ha generado multitud de estudios e investigaciones en metodologías para poder salvar el cuello de botella que supone la construcción manual de ontologías (Omelayenko, 2001; Shamsfard & Barforoush, 2004). Esta necesidad ha conducido a definir una nueva línea de investigación para conseguir la construcción de ontologías de manera semiautomática denominada *Ontology Learning*.

Actualmente existen diversos entornos para la adquisición de ontologías a partir de textos en lenguaje natural, pero la mayoría de ellos sólo tratan con jerarquías de conceptos (Maedche & Staab, 2001).

Las razones expuestas en los párrafos anteriores han sido las motivaciones primordiales para la realización de la investigación descrita en esta memoria. La solución que proponemos en este trabajo se basa en el desarrollo de un nuevo entorno para extracción incremental de conocimiento desde texto en lenguaje natural. Para ello, se ha adoptado el punto de vista de la ingeniería ontológica, de modo que el conocimiento adquirido por el entorno desarrollado se representa por medio de ontologías. Este trabajo aporta un nuevo método para la construcción semiautomática de ontologías a partir de textos en lenguaje natural que no sólo se centra en la obtención de jerarquías de conceptos, sino que tiene en cuenta también un amplio conjunto de relaciones semánticas entre conceptos.

Para lograr este objetivo, se ha seguido la siguiente metodología:

- Análisis del estado del arte en Ingeniería Ontológica. Esto implicó, por ejemplo, estudiar las diferentes definiciones del término ontología y las posibles clasificaciones y tipos de ontologías.

- Análisis de las metodologías actuales para diseñar y construir ontologías.

- Análisis de las metodologías actuales para la construcción automática de ontologías a partir de textos en lenguaje natural.

- Definición y formalización de un entorno para la extracción incremental de conocimiento desde texto en lenguaje natural. La forma de representación del conocimiento elegida han sido las ontologías. Este entorno está formado por tres fases secuenciales: POS-Tagging, Fase de búsqueda de conceptos, y Fase de inferencia.

- Diseño e implementación de una aplicación software para la extracción de ontologías a partir de texto en lenguaje natural de una manera semisupervisada e incremental. En este sistema los expertos interactúan seleccionando entidades de conocimiento explícitas que sean relevantes para el dominio, y que estén contenidas en el texto, entrenando así el sistema en dominios determinados.

Las cuestiones planteadas en la metodología anterior se han realizado con éxito, y los resultados obtenidos se presentan en esta memoria. La organización de la misma es como sigue:

En el capítulo 1 se tratan diferentes aspectos relacionados con ingeniería ontológica. En primer lugar, se recoge una serie de definiciones sobre el concepto ontología. A continuación, se describen aspectos importantes relacionados con ingeniería ontológica. Algunos ejemplos de los aspectos tratados en este capítulo son: tipos de ontologías, elementos de ontologías, uso y rol de las ontologías, y ejemplos de ontologías.

Además, se han analizado diferentes mecanismos y metodologías para diseñar y construir ontologías. Finalmente, este capítulo recoge un estudio sobre las metodologías existentes para la construcción automática de ontologías a partir de textos que ayuden a la consecución de la Web Semántica.

En el capítulo 2 se describe la importancia de la ingeniería lingüística, y más concretamente, de los POS-Taggers en las tareas de reconocimiento de lenguaje natural. Asimismo, se trata el estado del arte en POS-Tagging. Finalmente, en este capítulo se muestra el desarrollo de un POS-Tagger necesario para el entorno de extracción de conocimiento a partir de texto en lenguaje natural descrito en esta tesis.

En el capítulo 3 se describe un entorno de construcción de ontologías a partir de texto en lenguaje natural. El proceso de adquisición de conocimiento se compone de tres fases secuenciales:

- POS-Tagging: El objetivo principal de esta fase es obtener la categoría gramatical de cada palabra en la frase actual.

- Fase de búsqueda de conceptos: A través de esta fase, se identifican las expresiones lingüísticas que representan conceptos del dominio.

- Fase de inferencia: La idea principal sobre el modus operandi de esta fase es que, en lenguaje natural, las relaciones entre conceptos suelen estar asociadas a los verbos. Aunque este subproceso se encarga principalmente de obtener las relaciones entre conceptos, puede utilizarse también para adquirir otras entidades de conocimiento, como conceptos, atributos y valores.

En el capítulo 4, se muestran varios ejemplos que describen el funcionamiento del entorno de ontology learning descrito en el capítulo tres. En concreto, dichos ejemplos ilustran el proceso de adquisición del conocimiento a través de las tres fases en que se basa, utilizando para ello diferentes textos de distintos dominios.

Posteriormente, se evalúa la utilidad y aplicabilidad de la metodología para ontology learning en dos áreas principalmente, a saber, la construcción de ontologías del dominio para la Web Semántica y el reconocimiento de órdenes en lenguaje natural, en el contexto de la simulación de modelos de robótica médica.

La aplicación software desarrollada en este trabajo de investigación realizado se describe en el capítulo 6, donde se analizan diferentes aspectos de la herramienta tales como el tipo de usuarios o cómo construir una ontología a partir de texto, por medio de su uso. Finalmente, en el capítulo 7, se presentan algunas conclusiones finales así como líneas futuras de trabajo para mejorar los resultados obtenidos a través de esta tesis.

CAPÍTULO 1

ONTOLOGÍAS – ESTADO DEL ARTE

1.1 ONTOLOGÍAS

1.1.1 Definición

La noción de ontología ha recibido múltiples definiciones a lo largo de la historia. En este apartado presentaremos diferentes definiciones propuestas por individuos y estamentos de diferentes áreas centrándonos sobre todo en las definiciones mas utilizadas en en Inteligencia Artificial y Representación del Conocimiento.

Dentro de la Filosofía ha habido grandes nombres que han definido el concepto de ontología de diferentes maneras, como se puede ver a continuación:

> *"Ontología es la ciencia de algo y de nada, del ser y del no ser, de la cosa y del modo de la cosa, de la sustancia y el accidente" (Leibniz) (Couturat, 1903)*

> *"La filosofía trascendental es el sistema de todas nuestras cogniciones puras a priori, que podemos llamar ontología. Así, ontología trata con cosas en general, desde abstractas hasta particulares. Abarca todos los conceptos puros de la comprensión y todos los principios de la razón. Las ciencias principales que pertenecen a la metafísica son: ontología, cosmología, y teología. Ontología es una pura doctrina de elemento de toda nuestra cognición al completo, o: contiene la suma de todos nuestros conceptos puros que podemos tener a priori sobre la cosas (Kant, 2001)*

> *"La gente trata con asuntos relacionados con la teoría de entidades desde la antigüedad bajo el título de 'Metafísica' y, especialmente, bajo el título de 'Ontología' como parte de la metafísica; y ellos no han fallado siempre al reconocer las características de la libertad de existencia". (Meinong, 1921)*

Las definiciones de ontologías en Inteligencia Artificial, se han basado en las definiciones filosóficas de este término, y más concretamente a la interpretación del filósofo Quine (Quine, 1961), quien dijo que todo lo que puede ser cuantificado existe. A continuación se presentan las definiciones más importantes y en las que se basa este trabajo.

Una de las definiciones de ontología más extendidas es la dada por Tom Gruber (Gruber, 1993):

> *"Una ontología es una especificación explícita de una conceptualización. El término proviene de la filosofía, donde una ontología es un recuento sistemático de la existencia. En sistemas de Inteligencia Artificial, lo que existe es lo que puede ser representado. Cuando el conocimiento de un dominio se representa mediante un formalismo declarativo, el conjunto de objetos que puede ser representado se llama universo del discurso. Esos conjuntos de objetos, y las relaciones que se establecen entre ellos, son reflejados en un vocabulario con el cual representamos el conocimiento en un sistema basado en conocimiento. Así, en el contexto de IA, podemos describir la ontología de un programa como un conjunto de términos. En tal ontología, las definiciones asocian nombres de entidades del universo del discurso con textos comprensibles por los humanos que describen el significado de los nombres, y axiomas formales que limitan la interpretación y buen uso de dichos términos. Formalmente, una ontología es una teoría lógica"*

Para comprender esta definición, conviene explicar qué es una conceptualización, la cual se puede definir como una interpretación estructurada de una parte del mundo que usan los seres humanos para pensar y comunicar sobre ella. Para un informático, una conceptualización podría ser la clasificación de sistemas informáticos atendiendo a su naturaleza física en sistemas hardware, sistemas software y sistemas firmware. Por explícita entendemos que los conceptos y las restricciones se definen de forma explícita.

La definición de Gruber ha recibido varias críticas. Por ejemplo, Nicola Guarino (Guarino, 1995), tras examinar siete posibles interpretaciones de ontologías, afirmó:

"Un punto de inicio en este esfuerzo clarificador será el cuidadoso análisis de la interpretación dada por Gruber. El problema principal de dicha interpretación es que se basa en la noción de conceptualización. Una conceptualización es un conjunto de relaciones extensionales que describen un estado particular, mientras que la noción que tenemos en mente es intensional, esto es, algo como una rejilla conceptual al que le imponemos varios posibles estados".

Sin embargo, ésta no es la única crítica recibida por aquella definición, puesto que es considerada como demasiado general. Algunas opiniones afirman que algunas ontologías satisfacen dicha definición pero que no son útiles para desarrollar aplicaciones.

Nicola Guarino propuso una definición alternativa de ontología:

"En el sentido filosófico, podemos referirnos a una ontología como un sistema particular de categorías que representa una cierta visión del mundo. Como tal, este sistema no depende de un lenguaje particular: la ontología de Aristóteles es siempre la misma, independientemente del lenguaje usado para describirla. Por otro lado, en su uso más típico en IA, una ontología es un artefacto ingenieril constituido por un vocabulario específico para describir una cierta realidad, más un conjunto de supuestos explícitos concernientes al significado pretendido de las palabras del vocabulario. Este conjunto de supuestos tiene generalmente la forma de teorías lógicas de primer orden, donde las palabras del vocabulario aparecen como predicados unarios o binarios, respectivamente llamados conceptos y relaciones. En el caso más simple, una ontología describe una jerarquía de conceptos relacionados por relaciones de subsunción; en los casos más sofisticados, se añaden axiomas para expresar otras relaciones entre conceptos y restringir la posible interpretación."

Según este autor, las ontologías usadas en aplicaciones reales son realmente adaptaciones ingenieriles de ontologías, no ontologías propiamente dichas. Así, por ejemplo, siguiendo esta perspectiva, "el peso de una persona es una masa" es una afirmación que podría aparecer en una ontología, mientras que "el peso de una persona es un número" puede aparece en una adaptación

de la ontología debido a limitaciones de tiempo y recursos. Sin embargo, esta idea estricta de ontología no es usada ni por los más puristas en IA.

Así, en (Guarino, 1998), el autor establece que:

> *"Una ontología puede especificar una conceptualización en una forma muy indirecta, puesto que i) solo puede aproximar un conjunto de modelos pretendidos; y ii) tal conjunto de modelos pretendidos sólo es una caracterización débil de una conceptualización."*

Otra definición de ontología es la presentada por Borst (Borst, 1997), que refina la definición de Gruber:

> *"Una ontología es una especificación formal de una conceptualización compartida."*

En este contexto, "formal" se refiere a la necesidad de disponer de ontologías comprensibles por las máquinas. Esta definición enfatiza la necesidad de consenso en la conceptualización. Finalmente, "compartida" se refiere al tipo de conocimiento contenido en las ontologías, esto es, conocimiento consensuado y no privado.

Posteriormente, las definiciones de Gruber y Borst fueron explicadas en (Studer *et al*, 1998) de la siguiente forma:

> *"Conceptualización se refiere a un modelo abstracto de algún fenómeno en el mundo a través de la identificación de los conceptos relevantes de dicho fenómeno. Explícita significa que el tipo de conceptos y restricciones usados se definen explícitamente. Formal representa el hecho de que la ontología debería ser entendible por las máquinas. Compartida refleja la noción de que una ontología captura conocimiento consensual, esto es, que no es de un individuo, sino que es aceptado por un grupo"*

Por último cabe destacar, que a día de hoy no existe una definición consensuada de ontología, si no que incluso los autores de las definiciones más reconocidas y aceptadas se las autoplantean. Este es el caso de Gruber, el cual responde en un boletín publicado en Octubre de 2004 (Gruber, 2004) lo siguiente, cuando le preguntan si cambiaría de alguna forma la definición que dio de ontología y que ha sido muchas veces referenciada:

> *"Bien, los componentes más importantes de esa definición de ontología son que la ontología es un artefacto de representación (una especificación), distinta del mundo que modela, y que es un artefacto diseñado, construido para un propósito. Creo que la mayoría de científicos en computación obtienen la diferencia entre una especificación del mundo, incluso para mundos sintéticos. Retrospectivamente, no cambiaría la definición pero intentaría enfatizar que nosotros diseñamos ontologías. La consecuencia de esta vista es que podemos aplicar una disciplina de ingeniería en su diseño y evaluación. Si las ontologías son cosas derivadas de una ingeniería, entonces no tenemos que preocuparnos tanto sobre si son correctas y favorecer el negocio de construirlas para hacer algo útil. Podemos diseñarlas para conocer objetivos funcionales y restricciones. Podemos construir herramientas que nos ayuden a gestionarlas y validarlas. Y podemos tener múltiples ontologías que se coordinen o compitan basadas en un criterio objetivo mas que en una marca de fábrica o una autoridad."*

1.1.2 Tipos de ontologías

En la literatura podemos encontrar diferentes clasificaciones de tipos de ontologías. Principalmente se siguen dos criterios para tales clasificaciones: (a) el tipo de conocimiento que contienen; y (b) la motivación de la ontología.

1.1.2.1 Clasificación por el conocimiento que contienen

Este es el criterio donde existe mayor diversidad, la cual puede ser ilustrada por las dos siguientes clasificaciones de ontologías. La primera de ellas se propuso en (Van Heijst *et al*, 1997), donde se distinguen tres tipos de ontologías:

- **Ontologías terminológicas, lingüísticas**: Especifican los términos usados para representar conocimiento en un dominio determinado. Un ejemplo de ontologías terminológicas es la red semántica UMLS (Unified Medical Language System) (Lindberg *et al*, 1993). Dentro de las ontologías lingüísticas una de la más utilizada a nivel mundial es WordNet (Fellbaum, 1998), que contiene una gran estructura de términos y relaciones Taxonómicas y Partonómicas.

- **Ontologías de información**: Especifican la estructura de los registros de la base de datos. Los esquemas de bases de datos son un ejemplo.

- **Ontologías para modelar conocimiento**: Especifican conceptualizaciones de conocimiento. Estas ontologías tienen una estructura interna mucho más rica que los anteriores tipos de ontologías, y éstas son las ontologías que interesan a los desarrolladores de sistemas basados en conocimiento.

Una clasificación alternativa es la que se puede encontrar en (Mizoguchi *et al*, 1995), donde también se proponen tres categorías:

- **Ontologías del dominio**: Contienen todos los conceptos asociados a un dominio particular.

- **Ontologías de tarea**: Establecen la forma en la cual se puede usar el conocimiento del dominio para realizar tareas específicas. De esta forma, una aplicación podría realizar búsquedas de información mientras otra podría gestionar la asignación de bloques libre de memoria.

- **Ontologías generales**: Contienen descripciones generales sobre objetos, eventos, relaciones temporales, relaciones causales, modelos de comportamiento y funcionalidades.

1.1.2.2 Clasificaciones por motivación

A continuación se van a presentar dos clasificaciones distintas atendiendo a este criterio Según la primera de ellas, se distinguen cuatro tipos de ontologías:

- **Ontologías para la representación de conocimiento**: Permiten explicar las conceptualizaciones que subyacen en los formalismos de representación de conocimiento (Davis *et al*, 1993).

- **Ontologías genéricas**: Definen conceptos considerados genéricos en diferentes áreas. Ejemplos de tales conceptos serían componente, subclase, proceso, estado, etc. Estas ontologías son reutilizables en diferentes dominios. Se llaman también ontologías abstractas o superteorías porque permiten definir conceptos abstractos, y dichas ontologías pueden ser usadas para definir conceptos de forma más específica en diferentes dominios. Como ejemplos podemos ver la taxonomía, la mereología, la topología y la teoría general de sistemas.

- **Ontologías del dominio**: Definen conceptualizaciones específicas del dominio. Las

metodologías actuales de adquisición de conocimiento distinguen entre ontologías y conocimiento del dominio, porque el último describe situaciones factuales del dominio, mientras que las ontologías imponen descripciones sobre la estructura y contenido del conocimiento del dominio.

- **Ontologías de aplicación**: Están ligadas al desarrollo de una aplicación concreta. Tales ontologías cubren los aspectos relacionados con aplicaciones particulares. Típicamente, estas ontologías toman conceptos de ontologías del dominio y genéricas, así como métodos específicos para realizar la tarea, por lo que no son muy adecuadas para ser reutilizadas.

Una clasificación alternativa fue propuesta por Poli (Poli, 2000). En dicha clasificación se identifican los siguientes tipos de ontologías:

- **Ontologías generales:** Tienen que ver con las categorías fundamentales y sus conexiones de dependencia. Con respecto a las categorías fundamentales, los investigadores son cada vez más conscientes de la dificultad de manejar este nivel supremo. Por ello, es de máxima importancia emplear una organización de categorías principales que sea lo más transparente posible. Existen categorías fundamentales que se aplican a todos los niveles ontológicos. Sin embargo, muchas de las categorías top-level pueden tener diferentes valores en niveles diferentes de la ontología, aunque deben tener algo en común.

- **Ontologías categóricas:** Estudian las diversas formas en las que una categoría se da cuenta de los diversos niveles ontológicos, determinando la posible presencia de una teoría general que subsume sus concretizaciones. Mientras que la ontología general está más relacionada con la arquitectura de la teoría, la ontología categórica es más sensible a los detalles de las categorías individuales. Sin embargo, es obvio que ambas son necesarias.

- **Ontologías del dominio:** Se refieren a la estructuración detallada de un contexto de

análisis con respecto a los subdominios que lo componen.

- **Ontologías genéricas:** Aparecen ligadas a corpus lingüísticos y léxicos conceptuales. De hecho, los términos se pueden clasificar en varios niveles. Esto significa que cada término debería ser accesible por defecto únicamente en su sentido genérico, mientras que sus significados especializados quedan para cuando se active una ontología del dominio específica. Por otro lado, la ontología del dominio contiene términos que no tienen correspondencias analíticas en ontologías genéricas. El conocimiento del dominio "satura" el conocimiento genérico.

- **Ontología regional:** Analiza las categorías y sus conexiones de interdependencia para cada nivel ontológico (estrato o capa).

- **Ontología aplicada:** Estas ontologías son la aplicación concreta del entorno ontológico a un objeto específico (por ejemplo, un hospital).

1.1.2.3 Clasificaciones por el grado de formalidad de la ontología

Una tercera clasificación se basa en el grado de formalidad de la ontología. Según este criterio, se distinguen tres tipos de ontologías (ver (Poli, 2002)):

Ontología descriptiva, relacionada con la recolección de información sobre los ítems del dominio analizado. La unidad y variedad del mundo es la salida de las conexiones de dependencia y formas de independencia entre los ítems. Cosas materiales, plantas y animales, así como los productos de los talentos y actividades de animales y humanos, son ítems del mundo. En otras palabras, el mundo no sólo contiene cosas, animadas o no, sino también actividades y procesos, así como los productos derivados de los mismos. Es difícil negar que existen pensamientos, sensaciones y decisiones, así como el completo espectro de actividades mentales, y estamos obligados a admitir la existencia de reglas, lenguajes, sociedades y costumbres (Poli, 2001a).

Ontología formal, que destila, filtra, codifica y organiza los resultados de una ontología descriptiva. Según esta interpretación, la ontología formal lo es en el sentido de Husserl en sus "*Logical Investigations"*. Ser formal en este sentido implica tratar con categorías como cosa, proceso, materia, forma, todo, parte, etc. Estas categorías caracterizan aspectos y tipos de realidad que todavía no han sido utilizados bajo ningún formalismo.

La ontología formal se ha desarrollado de dos maneras principales (Albertrazzi, 1996). El primer enfoque consiste en estudiar la ontología formal como parte de la ontología, y analizarla usando las herramientas de modo que se aproxima a la lógica formal: desde este punto de vista, la ontología formal examina las características lógicas de predicación, así como aquellas de las diferentes teorías de universales. El uso del paradigma específico de la teoría de conjuntos aplicada a predicación condiciona su interpretación. El segundo enfoque vuelve a los orígenes Husserlianos y analiza las categorías fundamentales de objeto, estado, parte, todo, etc, así como las relaciones entre partes y todos y sus leyes de dependencia una vez que los conceptos materiales han sido sustituidos por sus conceptos formales correlativos al "algo" puro. Este tipo de análisis no trata con el problema de relaciones entre ontología formal y ontología material.

1.2 INGENIERÍA ONTOLÓGICA

1.2.1 Elementos de ontologías

Las ontologías proporcionan un vocabulario común de un área y definen, a diferentes niveles de formalismo, el significado de los términos y relaciones entre ellos. El conocimiento en ontologías se formaliza principalmente usando seis tipos de componentes: clases, atributos, relaciones, funciones, axiomas e instancias (Gruber, 1993).

- Las *clases* en la ontología se suelen organizar en taxonomías. Algunas veces, la noción de ontología se diluye en el sentido que las taxonomías se consideran ontologías completas (Studer *et al*., 1998). Se suele usar tanto el término clases como conceptos.

Un concepto puede ser algo sobre lo que se dice algo y, por lo tanto, también podría ser la descripción de una tarea, función, acción, estrategia, proceso de razonamiento, etc.

- Los *atributos* representan la estructura interna de los conceptos. Atendiendo a su origen, los atributos se clasifican en específicos y heredados. Los específicos son los propios del concepto al que pertenecen, mientras que los heredados vienen dados por las relaciones taxonómicas en las que el concepto desempeña el rol de hijo y, por tanto, hereda los atributos del padre. Los atributos se caracterizan por el dominio en el cual pueden tomar valor.

- Las *relaciones* representan un tipo de interacción entre los conceptos del dominio. Se definen formalmente como cualquier subconjunto de un producto de n conjuntos, esto es: R: C1 x C2 x ... x Cn. Como ejemplos de relaciones binarias incluimos: "subclase de" y "conectado a".

- Las *funciones* son un tipo especial de relaciones en las que el n-ésimo elemento de la relación es único para los n-1 precedentes. Formalmente, definimos las funciones F como: F: C1 x C2 x ... x Cn-1 → Cn. Como ejemplos, podemos mencionar las funciones "madre de" y "precio de un coche usado".

- Los *axiomas* son expresiones que son siempre ciertas. Pueden ser incluidas en una ontología con muchos propósitos, tales como definir el significado de los componentes ontológicos, definir restricciones complejas sobre los valores de los atributos, argumentos de relaciones, etc., verificando la corrección de la información especificada en la ontología o deduciendo nueva información. Tales ontologías son llamadas ontologías pesadas, en contraste con las ontologías ligeras que no incluyen axiomas.

- Las *instancias* son las ocurrencias en el mundo real de los conceptos. En una instancia, todos los atributos del concepto tienen asignado un valor concreto.

1.2.1.1 Relaciones

En (Gómez-Pérez *et al*, 2000) se enumeran las relaciones más comunes en dominios reales, a saber: equivalencia, taxonómica, partonómica, dependencia, topológica, causal, funcional, cronológica, similaridad, condicional y propósito. Sin embargo, no todas las relaciones tienen la misma relevancia ni imponen el mismo tipo de propiedades jerárquicas a la ontología.

Taxonomía

La palabra taxonomía tiene su origen en dos términos griegos, a saber, taxis (orden) y nomos (tratado) y esta palabra proviene de la Filosofía. Taxonomía es la ciencia que estudia la división en grupos ordenados o categorías. Desde un punto de vista ontológico, una taxonomía es una organización ontológica basada en una relación de orden parcial llamada IS-A, a través de la cual se agrupan las entidades y son subsumidas por clases de más ato nivel. En general, las taxonomías han sido importantes para modelar esquemas de bases de datos, sistemas basados en conocimiento y vocabularios semánticos (Guarino & Welty, 2001).

A continuación se presentan las propiedades satisfechas por las relaciones taxonómicas. Con este propósito, se usará la notación empleada en (Guarino & Welty, 2001). De esta forma, se dice que un individuo x perteneciente a una clase θ, IS-A individuo de la clase φ si for all x, φ (x) ➔ θ (x). Esto es, existe cierta dependencia entre las clases φ y θ, por lo tanto, cada predicado satisfecho por los individuos de la clase φ también es satisfecho por los individuos de la clase θ. Estas son las propiedades generales taxonómicas:

Asimetría: Esta propiedad significa que la inclusión de una clase de individuos, X, en una clase Y implica la no inclusión de Y en X. Formalmente, esta propiedad garantiza que: *(X IS-A Y)* ➔ *not(Y IS-A X)*.

Transitividad: Sea X incluido en una clase Y, que, a su vez, está incluida en una clase Z, ambas inclusiones a través de relaciones IS-A. Entonces, existe una relación IS-A entre las

clases X y Z. Formalmente, *(X IS-A Y) and (Y IS-A Z)* ➔ *(X IS-A Z)*.

Irreflexividad: Admitir la reflexividad en relaciones taxonómicas solo tendría sentido para modelar tautologías. Una tautología es la expresión de un mismo hecho de distintas maneras. La relación taxonómica se considera no reflexiva. Formalmente, *not(X IS-A X)*.

Existen otras propiedades taxonómicas que están relacionadas con los atributos de los conceptos a través de la taxonomía:

Redefinición: Esta propiedad consiste en cambiar el nombre de una propiedad común a dos conceptos, padre e hijo, y se asigna un nombre diferente al atributo en el hijo.

Herencia múltiple: Esta propiedad está asociada con atributos conceptuales. Un concepto puede tener diferentes padres taxonómicos, así que este concepto heredaría propiedades de todos sus padres.

Además de las propiedades taxonómicas básicas, existen otras condiciones basadas en cuestiones filosóficas relacionadas con taxonomías. Algunas de estas condiciones se señalan en (Guarino & Welty, 2001):

- Identidad
- Unidad
- Esencia
- Dependencia
- Rigidez

Las dos primeras condiciones se enlazan al concepto filosófico de "ser". Según Guarino, las intuiciones tras ambos conceptos requieren, con la finalidad de comprenderlos, hacer una distinción entre ellos. Así, la condición de identidad se relaciona con el problema de distinguir una instancia de su clase específica de instancias de la misma clase, por medio de lo que llamamos "propiedad característica", la cual es única para cada instancia.

Por otro lado, la unidad está relacionada con la distinción de partes en una instancia del resto del mundo. Con este propósito se usa una "relación unificante", que es un nexo entre todas las partes. Sin embargo, el límite entre ambas nociones no es evidente, especialmente cuando se invoca una dimensión temporal. De hecho, el principal problema aparece con la noción de "identidad a través de cambios". Eso nos lleva a admitir que los individuos pueden permanecer invariables aunque varíen algunas de sus propiedades. Estas propiedades invariables se llaman "propiedades esenciales", y constituyen la esencia de la clase.

Análogamente, la rigidez se define a través de "propiedades rígidas". Una propiedad rígida es una propiedad esencial para todas las instancias en las que aparece dicha propiedad. Finalmente, la dependencia ontológica se refiere a la existencia de relaciones particulares entre atributos y conceptos. Concretamente, en (Guarino & Welty, 2000) se distinguen dos categorías, intrínseca y extrínseca, según la dependencia de objetos. En particular, las propiedades extrínsecas no son inherentes a objetos sino que están ligadas a objetos. Finalmente, hay un número de conceptos que están asociados a relaciones IS-A que se han tomado directamente del modelado conceptual de bases de datos o del paradigma orientado a objetos, tales como claves primarias o identificadores únicos.

Sería interesante formalizar las condiciones presentadas anteriormente. Para ello se usará la formalización presentada en (Guarino & Welty, 2000).

Identidad

Supongamos dos individuos taxonómicos, x e y, y una relación θ tal que se cumplan θ(x) y θ(y). Se dice que hay una condición de identidad en uno de ellos si hay una relación φ tal que se cumplen las siguientes condiciones:

1. $\theta(x) \Lambda \theta(y)$
2. $\varphi(x,y)$
3. $x=y$

Rigidez

Según (Guarino, *et al*, 1994; Guarino & Welty, 2000), la rigidez de una propiedad se define formalmente con respecto a la cardinalidad del conjunto de instancias para el que tal propiedad es esencial, de acuerdo a las siguientes definiciones:

Definición 1. Propiedad rígida: Una propiedad rígida es una propiedad esencial para todas sus instancias. Formalmente, se dice que la propiedad θ es rígida si, para todo x que la tiene definida se cumplen:

1. para todo x, θ(x)
2. θ (x)

Definición 2. Propiedad no rígida: Una propiedad es no rígida cuando no es esencial para todas sus instancias. Formalmente, se dice que la propiedad θ es no rígida si, para todo x que la tiene definida se cumplen:

1. existe x, θ(x)

2. No θ (x)

Definición 3. Propiedad anti-rígida: Una propiedad es anti-rígida cuando es no esencial para ninguno de sus instancias en todos los posibles mundos. Formalmente, se dice que la propiedad θ es anti-rígida si, para todo x que la tiene definida se cumplen:

1. para todo x, θ(x)

2. No θ (x)

Diseño y construcción de taxonomías

De acuerdo con (Kremer, 2001), existen tres criterios a tomar en cuenta cuando construimos taxonomías:

- Adecuación descriptiva. La taxonomía debe clasificar la totalidad de los objetos del dominio a modelar (este criterio asume una metodología de construcción bottom-up en términos de nivel de abstracción)

- Simplicidad. La selección de nombres para cada elemento del dominio, así como su clasificación, debería ser una tarea simple.

- Capacidad predictiva. Esta es la propiedad más deseable para buenas taxonomías. Objetos con clasificaciones similares deberían tener propiedades similares.

Una buena taxonomía debe poseer un umbral de discriminación que debe ser lo suficientemente alto para no agrupar diferentes objetos en la misma categoría, y lo

suficientemente bajo para permitir la existencia de un número adecuado de categorías a modelar de forma manejable. Sin embargo, existen diferentes problemas relativos al uso de taxonomías. Los problemas más frecuentes derivados del mal uso de esta relación son discutidos en (Guarino, 1999). En particular se distinguen cinco problemas:

A) Reducción de sentido: Este es el caso de situaciones como "una asociación es un grupo".

B) Sobre-generalización. Este es el caso de situaciones como "una localización es un objeto físico".

C) Confusión de sentido: Este es el caso de situaciones como "una ventana es tanto un artefacto como una localización".

D) Conflicto de sentidos: Este es el caso de "una organización es tanto un ser social como un grupo".

E) Enlace sospechoso tipo-rol: Este es el caso de "una manzana es tanto una fruta como una comida".

Estos problemas han sido extraídos del análisis de taxonomías existentes conocidas. En particular, se han tomado ejemplos de las siguientes ontologías: PANGLOS (Knight & Luk, 1994), WORDNET (Miller, 1995), MICROCOSMOS (Mahesh, 1996), y CYC (Lenat & Guha, 1990).

Mereología

El término mereología es un término griego que significa "estudio de las partes". Este término fue usado por el filósofo polaco Lešniewsky para introducir una teoría formal sobre las partes y sus conceptos asociados. En (Simons, 1987) se presenta la Mereología Clásica Extensional, donde la mereología se ve como un modelo de organización ontológico. En la siguiente tabla (Tabla 1-1), se presenta la notación usada por Simons, que será usada posteriormente en esta sección.

Símbolo	Significado	Símbolo	Significado
$\exists$	Cuantificador Existencial	+	Suma binaria
$\forall$	Cuantificador Universal	-	Diferencia mereológicas
$\langle ... \rangle$	Ámbito de cuantificación	·	Producto binario
=	Igualdad	σ	Suma general
$\approx$	Identidad	π	Producto general
<	Parte de	$\perp$	Conexión
<<	Parte propia de	][	Desconexión
$\oplus$	Superposición	×	Conexión externa
|	Disyunción		

Tabla 1-1. Notación utilizada por Simons

La mereología va más allá del estudio de relaciones parte-de entre elementos de sistemas comunes. La mereología también se ocupa de aquellos elementos cuyas partes y todos son relevantes. Aquellos elementos se identifican como individuos. Con el objeto de realizar un razonamiento mereológico basado en individuos, se deben enunciar algunos principios y propiedades mereológicas.

Parte-de, parte propia e igualdad

En mereología, la relación parte-de se expresa mediante el operador "<". Esto permite establecer dos significados para cada relación: (1) uno no incluye el concepto de igualdad entre individuos, al que nos referimos como parte propia de ("<<"); (2) uno incluyendo la igualdad, referido como parte de ("<"). Ambos aspectos se pueden formalizar como sigue:

Dados dos individuos, x e y, x es una parte de y, denotado por x<y, sí y sólo si:

i) x << y, ó

ii) x = y

donde x<<y significa que "x es una parte propia de y"

Hay algunas propiedades que definen el significado del concepto universal de parte. Tales propiedades son asimetría y transitividad, de las que surge la no reflexividad. Formalicemos dichas propiedades.

Asimetría: $x << y \rightarrow \text{not } (y << x)$
Transitividad: $(x << y) \text{ and } (y << z) \rightarrow x << z$

Definamos ahora otros conceptos relacionados con relaciones parte-de. La superposición es la afirmación de que dos individuos tienen partes comunes. Formalmente, se puede definir como sigue: Dados dos individuos, x e y, $\exists z$ s.t. $\langle z < x \text{ and } z < y \rangle$. Por otro lado, la disjunción es la negación lógica de la superposición, y se define como not $\langle x \oplus y \rangle$. Se puede definir fácilmente que la superposición es reflexiva y simétrica pero no transitiva. Por otro lado, la superposición propia es la superposición que existe entre dos individuos x e y tales que x es parte de y, o ambos tienen una parte propia común.

Principio de suplementación débil: Si un individuo tiene una parte propia, entonces debe tener una parte que sea distinta de la anterior. Formalmente, $x << y \rightarrow \exists z$ t.q. $\langle (z << y)$ and $(z \mid x) \rangle$.

La parte común de los individuos superpuestos se llama límite inferior. La mereología establece que, dados dos individuos superpuestos x e y, existe un individuo que posee como partes todos los límites inferiores entre dichos individuos. Tal individuo es el producto binario, que se basa en la necesidad de diferenciar los conjuntos de partes solapadas entre x e y de los propios individuos. La suma binaria de dos individuos x e y, denotada por x+y, se define como el individuo que está superpuesto a, al menos, uno de ellos. El supuesto de existencia de la suma binaria para cada par de individuos mereológicos es la propiedad más controvertida en mereología clásica extensional.

Dados dos individuos superpuestos, sus partes compartidas las enlazan para formar un individuo simple. La suma es plausible para individuos de tipos similares pero no para aquellos individuos que sean muy diferentes en términos espacio-temporales o que tienen tipos diferentes. Por otro lado, dados dos individuos x e y que cumplen la superposición propia, su diferencia mereológica, denotada por x-y, puede ser calculada como la parte propia de x más grande que no tiene partes comunes con y. Esto sólo existirá si x no es parte de y. La existencia de la suma o producto para cada par de individuos no garantiza la existencia de la suma para cada clase de individuos, ni la existencia del producto para cada clase de individuos con una parte común, porque tales clases pueden ser infinitas (Simons, 1987). Sin embargo, se puede estudiar la suma o producto de individuos que pertenecen a cierta clase y que satisface cierto predicado. Formalmente, la suma general de cada individuo x perteneciente a la clase F, denotado por $\sigma x \langle Fx \rangle$, se define como sigue:

i) Fx, y

ii) $\forall y$ t.q. $\langle x \oplus y \rangle$, y

iii) $\exists z$ t.q. $\langle Fz \rangle$ and $\langle z \oplus y \rangle$

Dado un individuo que satisface un predicado en particular, se puede decir que, para cada individuo que se solapa con aquél, existe un tercer individuo que satisface los mismos predicados que se solapan con el segundo. Esto permite extender el concepto de suma a suma general. De forma similar, la noción de producto binario puede ser extendida a producto general de la forma siguiente. El producto general del individuo x de la clase F, denotado por $\pi x \langle Fx \rangle$, cumple las siguientes condiciones:

i) $\forall y$ t.q. $\langle \langle y < x \rangle ; y$

ii) $\langle \forall z \langle Fz \rangle$ and $\langle y < z \rangle \rangle$

Principio de la Suma General

Este principio permite la completa axiomatización de la mereología clásica extensional. Esto garantiza la existencia de sumas generales. Formalmente:

i) $\exists x \in Fx$, y

ii) $\exists x$ t.q. $\forall y \langle \langle y \oplus x \rangle \rangle$; y

iii) $\exists z$ t.q. $\langle \langle Fz \rangle$ and $\langle y \oplus z \rangle \rangle$

Principio de Extensionalidad

Este axioma se puede derivar de los anteriores, y establece que un individuo sólo puede ser distinguido por sus partes. Ahora entraríamos en la discusión sobre la posibilidad de que dos individuos tengan las mismas partes y que pudieran ser diferentes. Esto no es posible en mereología extensional. Sin embargo, existen distintos tipos de individuos, tales como eventos o clases para los cuales se debería permitir tal supuesto.

Atomismo

En mereología atomista, cada individuo puede ser considerado como construido a partir de individuos menores llamados átomos. Un átomo, denotado por At, es un individuo sin partes propias. Formalmente, las condiciones satisfechas por individuos atómicos son:

i) Atx, y

ii) not $\langle \exists z \text{ t.q. } \langle z << x \rangle \rangle$

Por otro lado, ciertos tipos de individuos son atómicos y otros no. Los intervalos temporales son ejemplos de esta última situación, porque pueden dividirse en intervalos menores. Sin embargo, los componentes de sistemas electrónicos son átomos del sistema. Esta suposición de la atomicidad del individuo simplifica la teoría, porque un individuo sería parte de otro individuo sí y sólo si todos sus átomos son también átomos del último. Esta propiedad también simplifica los axiomas mereológicos. Por ejemplo, la relación de identidad puede ser redefinida como la posesión de los mismos átomos. En resumen, no hay una teoría mereológica única, pero esto ofrece algunos ingredientes que se pueden utilizar para construir nuestras propias propiedades ontológicas. Cada teoría es adecuada para un tipo particular de individuo.

La transitividad de las relaciones parte/todo ha sido discutida por varios autores (Winston *et al*, 1987; Gerstl & Pribbenow,1993). Según (Winston *et al*, 1987), las relaciones

mereológicas pueden ser clasificadas en siete clases diferentes y, en general, no se asume la transitividad entre instancias de diferentes clases. Estas son las siete clases:

- Componente/Objeto: rama/árbol
- Miembro/Colección: árbol/bosque
- Porción/Masa: porción/tarta
- Constituyente/Objeto: aluminio/avión
- Característica/Actividad: pago/compra
- Fase/Proceso: adolescencia/desarrollo
- Lugar/Área: Murcia/España

La mayor parte de la investigación en las relaciones parte-de se ha centrado en el estudio de las partes. Sin embargo, en (Poli, 2001) se presenta una teoría de los todos. En ella se identifican diferentes tipos de todos de acuerdo a las siguientes propiedades:

- Separabilidad de partes y todos.
- Propiedad espacio-temporal de las partes.
- Papel desempeñado por la parte con respecto al todo.
- Homogeneidad de las partes.

Este autor presenta tres tipos de todos: agregados, todos y sistemas. En los agregados existe unidad por proximidad, pero no solidaridad ni unidad dinámica. En los todos podemos encontrar unidad por proximidad y solidaridad, mientras que no existe unidad dinámica. Finalmente, estas tres propiedades se pueden identificar en los sistemas.

1.2.1.2 Otras relaciones semánticas

Relación de Equivalencia

La relación de equivalencia establece la igualdad o equivalencia entre dos expresiones aparentemente diferentes. A continuación se presentan las propiedades satisfechas este tipo de relaciones:

Reflexividad: Esta propiedad significa que es posible la inclusión de una clase de individuos X en sí misma, siendo la inclusión a través de una relación EQUIVALENT. Formalmente, *(X EQUIVALENT X)*.

Simetría: Esta propiedad significa que la inclusión de una clase de individuos, X, en una clase Y, implica la inclusión de Y en X, y viceversa, ambas inclusiones a través de relaciones EQUIVALENT. Formalmente, esta propiedad garantiza que: *(X EQUIVALENT Y)* ⇔ *(Y EQUIVALENT X)*.

Transitividad: Sea X incluido en una clase Y, que, a su vez, está incluida en una clase Z, ambas inclusiones a través de relaciones EQUIVALENT. Entonces, existe una relación EQUIVALENT entre las clases X y Z. Formalmente tenemos lo siguiente, *(X EQUIVALENT Y) and (Y EQUIVALENT Z)* ➔ *(X EQUIVALENT Z)*.

Absorción: Esta propiedad significa que no se puede dar una inclusión de una clase X en el vacío ∅. Formalmente, *not(X EQUIVALENT ∅)*.

Esta relación suele aparecer en el lenguaje natural con expresiones parecidas a las siguientes:

- A *equivale a* B
- A *es igual a* B

Relación de dependencia

La relación de dependencia es un tipo especial de relación de asociación a través de responsabilidades, parentesco, propiedades, etc, y una participación. A continuación se presentan las propiedades satisfechas por este tipo de relaciones:

Reflexividad: Esta propiedad significa que es posible la inclusión de una clase de individuos X en sí misma, siendo la inclusión a través de una relación DEPENDENT. Formalmente, *(X DEPENDENT X)*.

Asimetría: Esta propiedad significa que la inclusión de una clase de individuos, X, en una clase Y, implica la no inclusión de Y en X, ambas inclusiones a través de relaciones DEPENDENT. Formalmente, esta propiedad garantiza que: *(X DEPENDENT Y)* ➔ *not (Y DEPENDENT X)*.

Transitividad: Sea X incluido en una clase Y, que, a su vez, está incluida en una clase Z, ambas inclusiones a través de relaciones DEPENDENT. Entonces, existe una relación DEPENDENT entre las clases X y Z. Formalmente tenemos lo siguiente, *(X DEPENDENT Y) and (Y DEPENDENT Z)* ➔ *(X DEPENDENT Z).*

Mutua dependencia: Sea X incluido en una clase Y, e Y incluida en una clase X, ambas inclusiones a través de relaciones DEPENDENT. Entonces, no podemos concluir la igualdad entre X e Y debido a que puede haber una dependencia mutua. Formalmente, *(X DEPENDENT Y) and (Y DEPENDENT X)* ¬➔ *(X = Y).*

Esta relación suele aparecer en el lenguaje natural en las formas siguientes:

- A *está asociado a* B
- A *depende de* B
- A *es responsable de* B

Relación topológica

La relación topológica describe la distribución espacial de conceptos físicos e interconexiones entre esos conceptos. A continuación se presentan las propiedades satisfechas por este tipo de relaciones:

Reflexividad: Esta propiedad significa que es posible la inclusión de una clase de individuos X en sí misma, siendo la inclusión a través de una relación CONNECTED-TO. Formalmente, *(X CONNECTED-TO X)*.

Simetría: Esta propiedad significa que la inclusión de una clase de individuos, X, en una clase Y, implica la inclusión de Y en X, y viceversa, ambas inclusiones a través de relaciones CONNECTED-TO. Formalmente, esta propiedad garantiza que: *(X CONNECTED-TO Y) ⇔ (Y CONNECTED-TO X)*.

Transitividad: Sea X incluido en una clase Y, que, a su vez, está incluida en una clase Z, ambas inclusiones a través de relaciones CONNECTED-TO. Entonces, existe una relación CONNECTED-TO entre las clases X y Z. Formalmente tenemos lo siguiente, *(X CONNECTED-TO Y) and (Y CONNECTED-TO Z) ➔ (X CONNECTED-TO Z).*

Esta relación suele aparecer en el lenguaje natural en las formas siguientes:

- A *está a la derecha de* B
- A *está encima de* B
- A *está debajo de* B
- A *está dentro de* B
- A *contiene a* B
- A *interseccciona a* B
- A *está conectado a* B
- A *está al lado de* B

Relación causal

Este tipo de relación describe como dados unos estados o acciones se induce a otros estados o acciones. A continuación se presentan las propiedades satisfechas por este tipo de relaciones:

Irreflexividad: La relación causal se considera no reflexiva. Es decir, no es posible la inclusión de una clase de individuos X en sí misma, siendo la inclusión a través de una relación CAUSED-BY. Formalmente, *not(X CAUSED-BY X)*.

Asimetría: Esta propiedad significa que la inclusión de una clase de individuos, X, en una clase Y, implica la no inclusión de Y en X, ambas inclusiones a través de relaciones CAUSED-BY. Formalmente, esta propiedad garantiza que: *(X CAUSED-BY Y)* ➔ *not (Y CAUSED-BY X)*.

No transitividad: Sea X incluido en una clase Y, que, a su vez, está incluida en una clase Z, ambas inclusiones a través de relaciones CAUSED-BY. Entonces, no se puede concluir que existe una relación CAUSED-BY entre las clases X y Z. Formalmente tenemos lo siguiente, *(X CAUSED-BY Y) and (Y CAUSED-BY Z)* ¬➔ *(X CAUSED-BY Z)*.

Relación de Dependencia: Sea X incluido en una clase Y a través de una relación CAUSED-BY. Entonces, esta relación implica una relación de dependencia entre X e Y. Formalmente, *(X CAUSED-BY Y)* ➔ *(X DEPENDENT Y)*.

Esta relación suele aparecer en el lenguaje natural de la siguiente manera:

- A *es la causa de* B
- A *activa a* B
- A *necesita a* B

Relación funcional

La relación funcional describe las condiciones para las acciones y reacciones que tienen lugar y las posibles consecuencias de las acciones. A continuación se presentan las propiedades satisfechas por este tipo de relaciones:

Irreflexividad: La relación funcional se considera no reflexiva. Es decir, no es posible la inclusión de una clase de individuos X en sí misma, siendo la inclusión a través de una relación FUNCTIONAL. Formalmente, *not(X FUNCTIONAL X)*.

No simetría: Sea X incluido en una clase Y, a través de una relación FUNCTIONAL Entonces, no se puede concluir que existe una relación FUNCTIONAL entre las clases Y y X. Formalmente tenemos lo siguiente, *(X FUNCTIONAL Y)* $\neg \rightarrow$ *(X FUNCTIONAL Z).*

Transitividad: Sea X incluido en una clase Y, que, a su vez, está incluida en una clase Z, ambas inclusiones a través de relaciones FUNCTIONAL. Entonces, existe una relación FUNCTIONAL entre las clases X y Z. Formalmente tenemos lo siguiente, *(X FUNCTIONAL Y) and (Y FUNCTIONAL Z)* $\rightarrow$ *(X FUNCTIONAL Z).*

Relación de Dependencia: Sea X incluido en una clase Y a través de una relación FUNCTIONAL. Entonces, esta relación implica una relación de dependencia entre X e Y. Formalmente, *(X FUNCTIONAL Y)* $\rightarrow$ *(X DEPENDENT Y).*

Esta relación suele aparecer en el lenguaje natural en las formas siguientes:

- A *permite a* B
- A *necesita a* B
- A *activa a* B

Relación cronológica

Esta relación se conoce también como relación temporal y describe la secuencia de tiempo en la que ocurren eventos. A continuación se presentan las propiedades satisfechas por este tipo de relaciones:

Irreflexividad: La relación cronológica se considera no reflexiva. Es decir, no es posible la inclusión de una clase de individuos X en sí misma, siendo la inclusión a través de una relación CHRONOLOGICAL. Formalmente, *not(X CHRONOLOGICAL X)*.

Asimetría: Esta propiedad significa que la inclusión de una clase de individuos, X, en una clase Y, implica la no inclusión de Y en X, ambas inclusiones a través de relaciones CHRONOLOGICAL. Formalmente, esta propiedad garantiza que: *(X CHRONOLOGICAL Y) ➔ not (Y CHRONOLOGICAL X)*.

Transitividad: Sea X incluido en una clase Y, que, a su vez, está incluida en una clase Z, ambas inclusiones a través de relaciones CHRONOLOGICAL. Entonces, existe una relación CHRONOLOGICAL entre las clases X y Z. Formalmente tenemos lo siguiente, *(X CHRONOLOGICAL Y) and (Y CHRONOLOGICAL Z) ➔ (X CHRONOLOGICAL Z)*.

Esta relación suele aparecer en el lenguaje natural en las formas siguientes:

- A *ocurre antes de* B
- A *occurre después de* B
- A y B *occurren simultáneamente*
- A *ocurre durante* B
- A *comienza cuando* B *termina*

Relación de similaridad

La relación de similaridad establece que conceptos son iguales o análogos y en que medida. A continuación se presentan las propiedades satisfechas por este tipo de relaciones:

Reflexividad: Esta propiedad significa que es posible inclusión de una clase de individuos X en sí misma, siendo la inclusión a través de una relación SIMILAR. Formalmente, *(X SIMILAR X)*.

Simetría: Esta propiedad significa que la inclusión de una clase de individuos, X, en una clase Y, implica la inclusión de Y en X, y viceversa, ambas inclusiones a través de relaciones *SIMILAR*. Formalmente, esta propiedad garantiza que: *(X SIMILAR Y) ⇔ (Y SIMILAR X)*.

No transitividad: Sea X incluido en una clase Y, que, a su vez, está incluida en una clase Z, ambas inclusiones a través de relaciones SIMILAR. Entonces, no se puede concluir que existe una relación SIMILAR entre las clases X y Z. Formalmente tenemos lo siguiente, *(X SIMILAR Y) and (Y SIMILAR Z) ¬➔ (X SIMILAR Z)*.

Esta relación suele aparecer en el lenguaje natural en la forma siguiente:

- A *es similar a* B

Relación condicional

La relación condicional define las condiciones en las cuales ciertas cosas tienen lugar. A continuación se presentan las propiedades satisfechas por este tipo de relaciones:

Irreflexividad: La relación cronológica se considera no reflexiva. Es decir, no es posible la inclusión de una clase de individuos X en sí misma, siendo la inclusión a través de una relación CONDITIONAL. Formalmente, *not(X CONDITIONAL X)*.

No simetría: Sea X incluido en una clase Y, a través de una relación CONDITIONAL Entonces, no se puede concluir que existe una relación CONDITIONAL entre las clases Y y X. Formalmente tenemos lo siguiente, *(X CONDITIONAL Y)* $\neg\rightarrow$ *(Y CONDITIONAL X).*

Transitividad: Sea X incluido en una clase Y, que, a su vez, está incluida en una clase Z, ambas inclusiones a través de relaciones CONDITIONAL. Entonces, existe una relación CONDITIONAL entre las clases X y Z. Formalmente tenemos lo siguiente, *(X CONDITIONAL Y) and (Y CONDITIONAL Z)* $\rightarrow$ *(X CONDITIONAL Z).*

Relación de Dependencia: Sea X incluido en una clase Y a través de una relación CONDITONAL. Entonces, esta relación implica una relación de dependencia entre X e Y. Formalmente, *(X CONDITIONAL Y)* $\rightarrow$ *(X DEPENDENT Y).*

Esta relación suele aparecer en el lenguaje natural en la forma siguiente:

- A *tiene como condición* B
- A *está condicionada a* B
- Si A entonces B

Relación de propósito

Esta relación establece el porqué y para qué de los conceptos. A continuación se presentan las propiedades satisfechas por este tipo de relaciones:

Irreflexividad: La relación cronológica se considera no reflexiva. Es decir, no es posible la inclusión de una clase de individuos X en sí misma, siendo la inclusión a través de una relación PURPOSE. Formalmente, *not(X PURPOSE X)*.

No simetría: Sea X incluido en una clase Y, a través de una relación PURPOSE. Entonces, no se puede concluir que exista una relación PURPOSE entre las clases Y y X. Formalmente tenemos lo siguiente, *(X PURPOSE Y)* $\neg\rightarrow$ *(X PURPOSE Z).*

Transitividad: Sea X incluido en una clase Y, que, a su vez, está incluida en una clase Z, ambas inclusiones a través de relaciones PURPOSE. Entonces, existe una relación PURPOSE entre las clases X y Z. Formalmente tenemos lo siguiente, *(X PURPOSE Y) and (Y PURPOSE Z)* $\rightarrow$ *(X PURPOSE Z).*

Esta relación suele aparecer en el lenguaje natural en la forma siguiente:

- Los códigos fueron creados por el ser humano *para transmitir* ideas
- A *ocurre con la finalidad de* B
- A *ocurre para* B
- A *nace con el propósito de* B

1.2.2 Ejemplos de ontologías

En las secciones anteriores se presentan diferentes clasificaciones de las ontologías. A continuación describiremos varios ejemplos de ontologías pertenecientes a dichos tipos. En particular, los ejemplos pertenecen a estas categorías:

- **Ontologías para representar conocimiento** (van Heijst *et al*, 1997): Capturan las primitivas de representación usadas para formalizar conocimiento en los paradigmas de representación de conocimiento. El ejemplo más representativo de este tipo es el de la Frame-Ontology (Gruber, 1993), que captura las primitivas de representación (clases, instancias, ranuras, facetas, etc) usadas en lenguajes basados en frames. Esta ontología define los términos que capturan convenciones usadas en sistemas de representación de conocimiento centrados en objetos. Puesto que estos términos se construyen sobre la semántica de KIF, se puede pensar que KIF y la frame-ontology constituyen un lenguaje especializado de representación. Dicha ontología es la base conceptual para los traductores de Ontolingua. Un propósito de esta ontología es permitir que diferentes sistemas de representación puedan compartir ontologías centradas en objetos. Los traductores de ontologías escritos en KIF usando la frame ontology, tales como los proporcionados por Ontolingua permiten trabajar desde un formato fuente común, y todavía lo siguen usando sistemas actuales de representación.

- **Ontologías generales/comunes** (Mizoguchi *et al*, 1995): Incluyen vocabularios relacionados con cosas, eventos, tiempo, espacio, causalidad, comportamiento, función, etc. A continuación, describiremos las características deseables para este tipo de ontologías (http://www.cyc.com/cyc-2-1/cover.html):

  - Universalidad, esto es, cada concepto imaginable puede ser enlazado correctamente en la ontología top-level en lugares adecuados, sin importar lo general o específico, lo prosaico, o el contexto en el que esté.

- Articulación. Las distinciones hechas en la ontología son necesarias y suficientes para la mayoría de propósitos. Existe una justificación teórica y pragmática para cada clase, para cada predicado y función, para cada individuo. Por suficiente entendemos que se hacen bastantes distinciones para permitir y facilitar la compartición del conocimiento, la integración y limpieza de las bases de datos, etc.

Un ejemplo sería la ontología top-level de Sowa (Sowa, 2000), que incluye las categorías y distinciones básicas derivadas de una variedad de fuentes de la Lógica, Lingüística, Filosofía e Inteligencia Artificial. Esta ontología tiene una estructura de enrejado donde el concepto raíz es el "*universal type"*, y el concepto ínfimo es el "*absurd type"*. Como subtipos del universal tenemos los conceptos primitivos (*independent*, *relative*, *mediating*, *continuant*, *occurrent*). Combinando estos conceptos primitivos obtenemos más conceptos de la ontología. Por ejemplo, *history = proposition* ∩ *occurrent*. Siguiendo con este proceso de combinación de conceptos se obtendría el resto de conceptos. Este enrejado quedaría cerrado por el tipo absurdo.

- **Meta-ontologías**, también llamadas ontologías genéricas o centrales (van Heijst *et al*, 1997), que son reusables en varios dominios. El ejemplo más representativo sería una ontología mereológica (Borst, 97) que incluiría el término parte-de. Otro ejemplo sería la ontología Cyc. La metodología Cyc surge de la experiencia del desarrollo de la base de conocimiento Cyc, que contiene una inmensa cantidad de conocimiento común, y que se construye sobre un núcleo de más de 1.000.000 de axiomas introducidos manualmente para capturar una gran parte de lo que la gente considera conocimiento consensuado sobre el mundo. La codificación de esta base de conocimiento fue realizada en CycL, que es un cálculo de predicados de primer orden aumentado con extensiones para manejar igualdad, razonamiento por defecto y algunas características de segundo orden. Esta base de conocimiento puede ser considerada una ontología porque puede usarse como base para construir sistemas inteligentes así como para comunicación e interoperabilidad.

- **Ontologías del dominio** (Mizoguchi *et al*, 1995; van Heijst *et al*, 1997), que son reusables en un dominio determinado. Proporcionan vocabularios sobre los conceptos y relaciones de un dominio, sobre las actividades que se desarrollan en dicho dominio, y sobre las teorías y principios elementales que gobiernan dicho dominio. Por ejemplo, las ontologías médicas se introducen para resolver problemas tales como la petición de reutilización y compartición de datos de los pacientes, su transmisión y necesidad de criterios basados en semántica con propósitos estadísticos. En este sentido, la comunicación no ambigua de conceptos médicos detallados y complejos es crucial hoy en día para sistemas de información médica. A continuación presentamos dos ejemplos, GALEN y UMLS.

    - GALEN (Rector *et al*, 1995) incluye un modelo semánticamente válido de terminología clínica representado en un lenguaje formal, y asociado con el soporte sofisticado para diferentes lenguajes naturales y conversión entre diferentes esquemas de codificación. GALEN se basa en un modelo semántico sólido de terminología clínica llamado GALEN Coding reference (CORE). Este modelo contiene conceptos clínicos elementales (p.ej., fractura, hueso, etc), relaciones que controlan la combinación de conceptos (p.ej., los huesos pueden tener fracturas), y conceptos complejos (p.ej., fractura de la clavícula).

    - El Unified Medical Language System (UMLS) (Pisanelli *et al*, 1998) es una base de datos diseñada para unificar terminologías biomédicas de fuentes dispares tales como terminologías clínicas, fuentes de drogas, o vocabularios en diversos idiomas. Existen tres fuentes de conocimiento UMLS, a saber: (1) metatesauro, que contiene información semántica sobre conceptos biomédicos, sus nombres y relaciones entre ellos;(2) red semántica, que es una red de categorías generales o tipos semánticos a lo que se asignan todos los conceptos del metatesauro; (3) el lexicón especialista, que contiene información sintáctica sobre términos biomédicos y eventualmente cubrirá la mayoría de términos correspondientes a los nombres de conceptos que aparecen en el metatesauro.

- **Ontologías lingüísticas**. La característica principal de este tipo de ontología es que están limitadas a la semántica de las unidades gramaticales. Forman un grupo heterogéneo de recursos usados principalmente en procesamiento del lenguaje natural. La mayoría de ontologías lingüísticas usan palabras como unidades gramaticales. De hecho, entre las ontologías que se han descrito antes, sólo la Generalized Upper Model captura información sobre unidades gramaticales mayores que palabras. Asimismo, algunas de ellas presentan un mapeo uno a uno entre conceptos y palabras en lenguaje natural, mientras que en otras ontologías se facilita un mapeo múltiple. Existen también diferencias con respecto al grado de dependencia del lenguaje. Algunas ontologías lingüísticas dependen totalmente de un idioma en particular; algunas abarcan múltiples lenguajes. Estas ontologías tienen diversos orígenes y motivaciones: bases de datos léxicas on-line; traducción, generación de lenguaje natural, etc. Wordnet (Miller, 1995) es probablemente la ontología lingüística más importante. Es una base de datos léxica para inglés basada en teorías psicolingüísticas. Está organizado en 70.000 conjuntos de sinónimos, llamados synsets, cada uno representando un concepto léxico. Los synsets se enlazan entre sí por medio de relaciones semánticas. Wordnet divide el lexicón en cinco categorías: nombres, verbos, adjetivos, adverbios y palabras con función. Los nombres se organizan en jerarquías temáticas, los verbos a través de relaciones de construcción, y los adjetivos y adverbios en hiperespacios N dimensionales.

1.3 CONSTRUCCIÓN DE ONTOLOGÍAS

1.3.1 Metodologías para Construir Ontologías a partir de cero

En (Lenat *et al*, 1990) se publicaron los pasos generales y algunos puntos interesantes relacionados con el proceso de desarrollo de Cyc. Posteriormente, en (Uschold & King., 1995) se comentó la metodología empleada para la creación de la ontología TOVE (TOronto Virtual Enterprise) para modelar organizaciones. Al año siguiente, estos autores propusieron unas reseñas metodológicas para construir ontologías (Uschold & Groninger, 1996). En (Bernaras *et al*, 1996) se presentó un método para construir ontologías para redes eléctricas como parte del proyecto KACTUS. Methontology (Fernández *et al*., 1997) apareció simultáneamente y fue extendido posteriormente (Fernández-López *et al*, 1999), (Fernández-López *et al*, 2000). En 1997, se propuso otra metodología basada en la ontología SENSUS (Swartout *et al*, 1997). Procedamos a continuación a describir un conjunto de diferentes metodologías para la construcción de ontologías a partir de cero.

Metodología Cyc

La metodología Cyc (Lenat *et al*, 1990) consiste en varios pasos. En primer lugar hay que extraer manualmente el conocimiento común que está implícito en diferentes fuentes. A continuación, una vez que tengamos suficiente conocimiento en nuestra ontología, podemos adquirir nuevo conocimiento común usando herramientas de procesamiento de lenguaje natural o aprendizaje computacional. Así se construyó la ontología Cyc. Esta metodología recomienda los siguientes pasos:

- Codificación manual de conocimiento implícito y explícito extraído de diferentes fuentes

- Codificación de conocimiento usando herramientas software

- Delegación de la mayor parte de la codificación en las herramientas

Metodología de Construcción de Ontologías de Uschold y King

Esta metodología fue presentada en (Uschold & King, 1995) y propone algunos pasos generales para desarrollar ontologías, a saber: (1) identificar el propósito; (2) capturar los conceptos y relaciones entre estos conceptos y los términos utilizados para referirse a estos conceptos y relaciones; (3) codificar la ontología. La ontología debe ser documentada y evaluada, y se pueden usar otras ontologías para crear la nueva. De esta forma se creó la Enterprise Ontology. Esta metodología recomienda los siguientes pasos:

- Identificar propósito
- Capturar la ontología
- Codificación
- Integrar ontologías existentes
- Evaluación
- Documentación

Metodología de Construcción de Ontologías de Grüninger y Fox

En esta metodología, presentada en (Grüninger *et al*, 1995), el primer paso es identificar intuitivamente las aplicaciones posibles en las que se usará la ontología. Posteriormente, se usa un conjunto de preguntas en lenguaje natural, llamadas cuestiones de competencia, para determinar el ámbito de la ontología. Se usan estas preguntas para extraer los conceptos principales, sus propiedades, relaciones y axiomas, los cuales se definen formalmente en Prolog. Por consiguiente, ésta es una metodología muy formal que se aprovecha de la robustez de la lógica clásica y que puede ser usada como guía para transformar escenarios informales en modelos computables. Esta metodología, que se usó para construir la ontología TOVE, recomienda los siguientes pasos:

- Escenarios motivantes
- Cuestiones informales de competencia
- Terminología formal
- Cuestiones formales de competencia
- Axiomas formales
- Teoremas de completitud

Metodología KACTUS

En esta metodología (Bernaras *et al*, 1996) se construye la ontología sobre una base de conocimiento por medio de un proceso de abstracción. Cuantas más aplicaciones se construyen, las ontologías se convierten en más generales y se alejan más de una base de conocimiento. En otras palabras, se propone comenzar por construir una base de conocimiento para una aplicación específica. A continuación, cuando se necesita una nueva base de conocimiento en un dominio parecido, se generaliza la primera base de conocimiento en una ontología y se adapta para las dos aplicaciones, y así sucesivamente. De esta forma, la ontología representaría el conocimiento consensuado necesario para todas las aplicaciones. Esta metodología ha sido utilizada para construir una ontología para diagnosticar fallos, y recomienda seguir los siguientes pasos:

- Especificación de la aplicación
- Diseño preliminar basado en categorías ontológicas top-level relevantes
- Refinamiento y estructuración de la ontología

METHONTOLOGY

Methontology es una metodología para construir ontologías tanto partiendo desde cero como reusando otras ontologías, o a través de un proceso de reingeniería. Este entorno permite la construcción de ontologías a nivel de conocimiento, e incluye: (1) identificación del proceso de desarrollo de la ontología donde se incluyen las principales actividades (evaluación, gestión de configuración, conceptualización, integración, implementación, etc); (2) un ciclo de vida basado en prototipos evolucionados; y (3) la metodología propiamente dicha, que especifica los pasos a ejecutar en cada actividad, las técnicas usadas, los productos a obtener y cómo deben ser evaluados. Esta metodología está parcialmente soportada por el entorno de desarrollo ontológico WebODE. Esta metodología ha sido usada en la construcción de múltiples ontologías, como una ontología química, ontologías hardware y software, etc. Se proponen los siguientes pasos:

- Especificación
- Conceptualización
- Formalización
- Implementación
- Mantenimiento

Metodología SENSUS

La metodología basada en Sensus (Swartout *et al*, 1997) es un enfoque top-down para derivar ontologías específicas del dominio a partir de grandes ontologías. Los autores proponen identificar un conjunto de términos semilla que son relevantes en un dominio particular. Tales términos se enlazan manualmente a una ontología de amplia cobertura. Los usuarios seleccionan automáticamente los términos relevantes para describir el dominio y acotar la ontología Sensus. Consecuentemente, el algoritmo devuelve el conjunto de términos estructurados jerárquicamente para describir un dominio, que puede ser usado como esqueleto para la base de conocimiento. Esta metodología sirvió para construir la ontología Sensus y recomienda los siguientes pasos:

- Tomar una serie de términos como semillas.
- Enlazarlos manualmente.
- Incluir todos los conceptos en el camino que va de la raíz de Sensus a los conceptos semilla.

- Añadir nuevos términos relevantes del dominio.

- Opcionalmente, añadir, para aquellos nodos por los que pasan más caminos, su subárbol inferior.

Metodología On-To-Knowledge

El proyecto OTK (Staab *et al*, 2001) aplica ontologías a la información disponible electrónicamente para mejorar la calidad de la gestión de conocimiento en organizaciones grandes y distribuidas. La metodología proporciona guías para introducir conceptos y herramientas de gestión de conocimiento en empresas, ayudando a los proveedores y buscadores de conocimiento a presentar éste de forma eficiente y efectiva. Esta metodología incluye la identificación de metas que deberían ser conseguidas por herramientas de gestión de conocimiento y está basada en el análisis de escenarios de uso y en los diferentes papeles desempeñados por trabajadores de conocimiento y accionistas en las organizaciones. Cada una de las herramientas de la arquitectura de OKT se centra en el desarrollo de aplicaciones dirigidas por ontologías y, finalmente, describe el uso y la evaluación de la metodología mediante casos de estudio como, por ejemplo, la ontología Proper o AIFB. Esta metodología recomienda los siguientes pasos:

- Estudio de viabilidad

- Comienzo

- Refinamiento

- Evaluación

TERMINAE

Terminae (Aussenac-Gilles *et al*, 2002) aporta tanto una metodología como una herramienta para la construcción de ontologías a partir de textos. Se basa en un análisis lingüístico de los textos, el cual se realiza mediante la aplicación de diferentes herramientas para el procesamiento del lenguaje natural. En particular, se usan dos herramientas: (1) Syntex para identificar términos y relaciones; y (2) Caméléon para identificar roles o relaciones. Estas herramientas se basan en la misma hipótesis lingüística: el significado de las frases y las palabras es específico para un dominio y puede ser inferido de la observación de regularidades en documentos. La metodología funciona como sigue. Mediante la aplicación de Syntex obtenemos una lista de posibles palabras y frases del texto y algunas dependencias sintácticas y gramaticales entre ellas. Estos datos se usan como entrada para el proceso de modelado junto con el texto original. De esta forma, la identificación de conocimiento se basa en dos tareas que se realizan alternativamente:

- Explorar los resultados Syntex para identificar conocimiento importante o decidir cómo representar alguna información de acuerdo al uso de las palabras en el texto.

- Extraer sistemáticamente del texto tanto conocimiento como sea posible.

Cada pieza de conocimiento puede ser representada en el modelo de conocimiento de Terminae, cuyo lenguaje de representación de conocimiento posee las siguientes primitivas: fichero terminológico (términos), conceptos genéricos (clases), conceptos primitivos (instancias), y roles (relaciones). El siguiente paso es normalizar el conocimiento para obtener una ontología bien estructurada, donde cada concepto quede justificado por sus relaciones con otros conceptos. Esta metodología sugiere aplicar criterios diferenciadores para hacer explícitas las propiedades comunes y diferentes de un concepto con sus respectivos conceptos padre y hermanos debidas a sus roles. La última etapa es la formalización de la ontología en el lenguaje formal Terminae, que es un tipo de lógica descriptiva. Una función de clasificación sirve para comprobar la corrección de las definiciones de conceptos genéricos, ya que sólo pueden ser definidos si tienen roles diferenciados.

1.3.2 Extracción de ontologías a partir de texto

La extracción de conocimiento directamente de textos en lenguaje natural es una tarea muy importante dentro de la inteligencia artificial y la ingeniería de conocimiento, ya que se podrían simplificar los procesos de adquisición de conocimiento de tal forma que los ingenieros de conocimiento no fuesen necesarios para dicha actividad y el conocimiento pudiese ser extraído directamente por los expertos a partir de estos textos en lenguaje natural.

Diferentes disciplinas como la lingüística computacional, la recuperación de información, el aprendizaje computacional o la ingeniería del software, han investigado y diseñado multitud de técnicas que podemos utilizar en la extracción de conocimiento a partir de texto y, más concretamente, en la construcción de ontologías (ontology learning).

Dentro del aprendizaje de ontologías una actividad muy importante es la construcción de ontologías a partir de documentos Web (Alani *et al*, 2003; Davulcu *et al*, 2003) para la consecución de la Web Semántica (Berners-Lee *et al*, 2001). Esto es porque se necesita enriquecer la Web con grandes cantidades de ontologías que capturen el conocimiento del dominio y la construcción manual requiere un enorme esfuerzo humano, siendo la adquisición de ontologías un cuello de botella para la consecución de la Web Semántica (Omelayenko 2001).

La mayoria de las técnicas de ontology learning sólo generan jerarquías de conceptos (Taxonomías) o utilizan un conjunto muy reducido de relaciones (Omelayenko, 2001; Maedche and Staab, 2001). Además, la mayoría de técnicas son semi-automáticas y necesitan de un ingeniero de conocimiento que guíe el proceso de adquisición de conocimiento.

Las aproximaciones de ontology learning pueden dividirse en tres grupos principales según el método utilizado: aproximaciones simbólicas, aproximaciones estadísticas y aproximaciones basadas en machine learning. Además, cada una de éstas puede dividirse según el tipo de ontología que se pretende obtener: ontologías de lenguaje natural (Natural Language Ontologies (NLO)), ontologías del dominio e instancias de ontologías (Omelayenko, 2001).

Dentro de cada una de estas clasificaciones, el proceso de ontology learning puede descomponerse en dos grandes grupos de tareas: adquisición y mantenimiento de ontologías:

Adquisición de ontologías:

- Creación de ontologías: En esta tarea, el sistema asiste al ingeniero de conocimiento sugiriéndole las relaciones más importantes en el dominio o chequeando y verificando las bases de conocimiento construidas.
- Extracción de esquemas de ontologías de documentos Web: Los sistemas toman los datos y el meta-conocimiento (como una meta-ontología) como entrada y generan una ontología de salida lista para ser usada, con la posible ayuda del ingeniero de conocimiento.
- Extracción de instancias de ontologías: Dado un esquema de ontología, se extraen las instancias de la ontología presentes en documentos Web.

Mantenimiento y refinamiento de ontologías:

- Integración y navegación de ontologías: Trata la reconstrucción y navegación en grandes bases de conocimiento construidas a mano o con técnicas de machine learning. Por ejemplo, la tarea puede consistir en el cambio de los formatos de bases de conocimiento.
- Actualización de ontologías: Esta tarea actualiza algunas partes de la ontología que son diseñadas para ser actualizadas (por ejemplo, los tags de formateo que hacen cambios en los layouts de página).
- Enriquecimiento de ontologías: Incluye modificaciones automáticas de relaciones menores dentro de una ontología existente. No cambia conceptos mayores ni las estructuras, pero hace la ontología más precisa.

1.3.2.1 Aproximaciones Simbólicas

Extracción de ontologías

La idea de utilizar patrones léxico-sintácticos en la forma de expresiones regulares para la extracción de relaciones semánticas y, en particular, relaciones taxonómicas fueron introducidas por Hearst (Hearst, 1992). Estas aproximaciones se basan en métodos heurísticos que utilizan expresiones regulares que originalmente han sido aplicadas en el área de la extracción de información (Hoobs, 1993). En estas aproximaciones de ontology learning, se procesa el texto buscando ocurrencias de estos patrones léxico-sintácticos que indican un tipo de relación (como pueden ser las relaciones taxonómicas). Por eso, la idea principal es bastante sencilla: definir una expresión regular que capture expresiones que vuelvan a ocurrir y mapear los resultados de la expresión asociada a una estructura semántica, como relaciones taxonómicas entre conceptos.

Por ejemplo, en (Hearst, 1992) se considera el siguiente patrón lexico-sintáctico:

…NP {, NP}*{,} u otros(as) NP…

Cuando aplicamos este patrón a una frase puede inferir que los NPs de la izquierda de la expresión "u otros(as)" son subconceptos del NP de la izquierda. Por ejemplo en la frase:

"Las contusiones, las heridas, la rotura de huesos u otras lesiones son comunes en los maltratos".

Se extraerían las relaciones taxonómicas siguientes:

CONTUSIÓN **IS A** LESIÓN
HERIDA **IS A** LESIÓN
ROTURA DE HUESOS **IS A** LESIÓN

En (Hearst, 1992), los patrones se definen manualmente, lo cual es una tarea que necesita mucho tiempo y está sujeta a errores. En (Morin, 1999) se extiende el trabajo propuesto por (Hearst, 1992) usando una herramienta simbólica de machine learning para refinar los patrones léxico sintácticos. En este contexto, el sistema PROMETHEE soporta adquisición de relaciones semánticas semi-automática y el refinamiento de estos patrones léxico-sintácticos.

El trabajo de Assadi (Assadi, 1999) muestra un experimento práctico de la construcción de una ontología en el campo de la planificación de redes eléctricas. Describe una primera aproximación de clustering que combina criterios lingüísticos y conceptuales. Como un ejemplo, el autor da el patrón <NP, line> que da como resultado dos categorizaciones por modificadores. La primera categorización está motivada por los modificadores "función de estructura", resultando un clustering de "connection line", "dispatching line" y "transport line" (ver tabla 1-2).

A proposal categorization	The other candidate terms
Connection line	Mountain line
Dispatching line	Telecommunication line
Transport line	Input line

Tabla 1.2. Ejemplo de Categorización

Faure y Nedellec (Faure and Nedellec, 1998) presentaron un sistema cooperativo de machine learning llamado ASIUM que es capaz de adquirir relaciones taxonómicas mediante un análisis sintáctico. Este sistema está basado en un algoritmo de clustering conceptual, donde los grupos básicos están formados por "head words" que aparecen con el mismo verbo después de la misma preposición. ASIUM traduce los grupos a nuevos conceptos y la jerarquía de conceptos formando la ontología.

Un sistema de ontology learning donde se aplican diferentes técnicas a definiciones de diccionario en el contexto de los dominios de la seguridad y las telecomunicaciones se describen

en (Kietz, *et al*, 2000; Maedche & Staab, 2000). Un aspecto importante en este sistema es que los conceptos existentes se incluyen en el proceso completo. Por eso, al contrario que (Hearst, 1992; Morin, 1999) las operaciones de extracción son realizadas en el nivel de los conceptos. Por tanto, por eso los patrones son directamente mapeados en conceptos, lo cual hace que el sistema sea capaz de, además de extraer relaciones taxonómicas, refinar las relaciones existentes y relacionarlas con los conceptos existentes.

Mantenimiento y refinamiento de ontologías.

Hahn y Schnattinger (Hahn & Schnattinger, 1998) introdujeron una metodología para el mantenimiento y refinamiento de taxonomías específicas del dominio. Una ontología se actualiza incrementalmente con nuevos conceptos adquiridos desde textos. El proceso de adquisición se centra en la "calidad" lingüística y conceptual de varias formas de evidencia subyacentes en la generación y refinamiento de hipótesis de conceptos. En particular, consideran conflictos semánticos y estructuras semánticas análogas de una base de conocimiento en la ontología con la finalidad de determinar la calidad de una propuesta particular. Para ello, refinan y extienden una ontología existente con conceptos y relaciones taxonómicas entre conceptos.

El sistema Camille (Contextual Acquisition Mechanism for Incremental Lexeme Learning) fue desarrollado como un sistema de comprensión del lenguaje natural. Cuando el analizador se encuentra con palabras que no conoce, Camille intenta inferir lo que pueda sobre el significado de la palabra desconocida (Hastings, 1994). Si la palabra desconocida es un nombre, hay unas restricciones semánticas útiles que limitan el significado del nombre. El aprendizaje de verbos desconocidos es más difícil, por lo cual la adquisición de verbos ha sido el objetivo principal de las investigaciones de Camille, el cual ha sido probado en muchos dominios del mundo real.

1.3.2.2 Aproximaciones estadísticas

En este apartado se consideran algunas aproximaciones a la construcción de taxonomías que infieren nuevos conceptos y sus relaciones, basándose en datos estadísticos sobre la concurrencia de palabras que expresan estos conceptos. La idea principal que tienen en común todas estas aproximaciones es que la identidad semántica de una palabra se refleja en su distribución sobre diferentes contextos, por lo que el significado de una palabra se representa en términos de la concurrencia de palabras con su frecuencia de ocurrencia. Esta manera de representar la semántica obvia la necesidad de preparar recursos especiales para procesarla, como patrones léxico-sintácticos, y pretende hacer el proceso de la extracción de taxonomías completamente automático. Sin embargo, el problema principal de estas aproximaciones es la escasez de datos notorios, por ejemplo el hecho de que los corpus disponibles sobre palabras de interés puedan no ser muy indicativos del significado del concepto en este dominio.

Extracción de ontologías

Los datos distributivos sobre palabras pueden usarse para construir conceptos y la jerarquía formada por estos desde cero. El clustering puede definirse como el proceso de organizar objetos dentro de grupos en los que los miembros de cada grupo son similares de alguna forma (Kaufman & Rousseeuw, 1990). En general, hay dos tipos de clustering: **clustering no jerárquico,** en el que cada objeto se asigna exactamente a un grupo, y **clustering jerárquico**, en el que cada grupo de tamaño mayor que uno está compuesto de grupos más pequeños. Los algoritmos de clustering jerárquico son preferibles para un análisis de datos detallado. Producen jerarquías de grupos y, además, contienen más información que los algoritmos no jerárquicos. Sin embargo, son menos eficientes con respecto al tiempo y al espacio que el clustering no jerárquico. (Manning & Schuetze, 1999) identifica dos usos principales para el clustering en el procesamiento del lenguaje natural:

- El uso de clustering para el análisis de datos exploratorio

- El uso de clustering para la generalización.

Uno de los de los primeros trabajos en esta área, denominado también *clustering distributivo* de palabras inglesas, se describe en (Pereira *et al*, 1993). Su trabajo se centra en la construcción de clases basadas en modelos de ocurrencia de palabras.

La herramienta Mo'K (Bisson *et al*, 2000) soporta el desarrollo de métodos de clustering conceptual para la construcción de ontologías. Este trabajo se centra en la elaboración de métodos de clustering para realizar la creación de jerarquías de conceptos (Taxonomías) asistidas por los ingenieros de conocimiento. La entrada de los métodos de clustering viene dada por las clases (nombres) y sus atributos (relaciones gramaticales) obtenidas a partir de un análisis sintáctico del corpus, junto con la frecuencia en la que aparecen en el corpus.

El algoritmo utiliza un método de clustering para agrupar los objetos 'similares' creando determinadas clases y la jerarquía entre éstas. El usuario puede ajustar varios parámetros del proceso para mejorar el rendimiento, seleccionando ejemplos de entrada con sus atributos, el nivel de poda, y las funciones de evaluación de distancia. El artículo presenta un estudio experimental que ilustra cómo la calidad del aprendizaje depende de las diferentes combinaciones de los parámetros.

Mantenimiento y refinamiento de ontologías

Los datos estadísticos sobre conceptos obtenidos desde el texto también se utilizan para ampliar una ontología existente con nuevos conceptos. En este caso, se aplica un método de clasificación automática para determinar un lugar apropiado para los nuevos conceptos en la taxonomía.

Anteriores aproximaciones para la aplicación de una ontología automáticamente utilizaban diversas técnicas de clasificaciones que pueden resumirse en los siguientes métodos: el método de los k vecinos (kNN), el método basado en categorías y el método basado en centroides. Todos ellos operan sobre la representación semántica basada en vectores que describen el significado de una palabra de interés en términos de cuentas de su ocurrencia con otras palabras en el contexto, por ejemplo, de palabras que aparecen dentro de algún contexto

que rodea a la palabra objetivo. Las diferencias clave entre los métodos provienen de diferentes ideas subyacentes sobre cómo se representa una clase (semántica) de palabras. El método kNN se basa en la suposición de que los miembros de una clase se definen porque la nueva instancia es similar a uno o más individuos de la clase. De ese modo, se define la similaridad por una puntuación como, por ejemplo, el coseno entre dos vectores de ocurrencia. Para clasificar una nueva instancia, primero se determina el conjunto de las k instancias de entrenamiento más similares a la nueva instancia. La nueva instancia se asigna a la clase que tenga el mayor número de miembros en el conjunto de los vecinos más cercanos. Además, la decisión de clasificación se puede basar en la similaridad de la medida entre la nueva instancia y sus vecinos: cada vecino puede votar por su clase con un peso a la nueva instancia. Cuando este método se aplica a la ampliación de un thesauro, la clase de instancias de entrenamiento se toman normalmente para ser constituidas por palabras a lo largo del mismo conjunto de sinónimos, por ejemplo, lexicalizando el mismo concepto (Marti *et al*, 1993).

La mayor desventaja de aplicar este método a menudo es que requiere un gasto computacional muy significativo para calcular la similaridad entre las nuevas instancias y cada instancia del conjunto de entrenamiento. Otro método alternativo con un menor costo computacional es el método basado en categorías (Resnik & Smith, 1993). Aquí, la suposición es que la clase de los miembros se define por la similaridad del nuevo ítem con una representación generalizada de la clase. La representación generalizada se construye añadiendo todos los vectores que constituyen una clase y normalizando el vector resultante, es decir, se calcula un vector probabilístico que representa la clase. Para determinar la clase de una nueva palabra, este vector unitario se compara con cada vector que representa una clase. Es por esto por lo que el número de operaciones de cálculo se reduce al número de clases. De ese modo, una representación de clase puede derivar desde un conjunto de vectores correspondientes a un conjunto de sinónimos a un conjunto de vectores correspondientes a un conjunto de sinónimos y alguno o todos los conjuntos de sinónimos subordinados. En el caso de kNN, una clase se representa por su conjunto de sinónimos. El otro extremo es representar la clase en la misma manera que se utiliza, por ejemplo en (Resnik & Smith, 1993), donde se representan los conceptos de un tesauro. Aquí un vector de clase se construye desde los datos de todos los sinónimos del concepto correspondiente; las nuevas palabras que resulten similares a este vector se asignan a la clase que lexicalice el concepto correspondiente. Otra forma de preparar la

representación de una clase de palabras es lo que se puede llamar una aproximación basada en centroides (Pereira *et al*, 1993). Es como el método basado en categorías, con la única diferencia que el vector que representa una clase se calcula de una manera distinta. Todos los n vectores correspondientes a los miembros de la clase se suman y el vector resultante se divide por n para calcular el centroide entre los n vectores.

Otro sistema estadístico para enriquecer una ontología es el presentado en (Agirre *et al*, 2000), que trata de enriquecer los conceptos de la ontología WordNet (Millar, 1990; Fellbaum, 1998), proponiendo un método para construir listas de palabras relacionadas con cada concepto existente en WordNet, donde cada significado de cada palabra tiene una lista asociada de palabras relacionadas. Por ejemplo, la palabra "mano" tiene, al menos, dos posibles significados: La mano que forma una parte del cuerpo (palabras relacionadas: mano-cuerpo, brazo, hombre) o una mano de pintura (palabras relacionadas; mano-pintura, pintor, brocha). El sistema entonces busca por documentos Web, utilizando el motor de búsqueda AltaVista, relacionados con cada concepto de WordNet y construye una lista de palabras asociadas con el tópico. Estas búsquedas se realizan consultando en los documentos que contengan las palabras asociadas al significado, pero que no contengan las palabras asociadas a otros significados. Por ejemplo, una consulta podría ser algo como "mano Y (cuerpo O brazo) Y NO (pintura O brocha)" para obtener los documentos Web relacionados con el significado del concepto mano como parte del cuerpo.

1.3.2.3 Aproximaciones de machine learning

Las primeras aproximaciones de ontology learning utilizando técnicas de ML se basaron en el descubrimiento de relaciones jerárquicas entre clases utilizando Ripple-Down Rules (Suryanto & Compton, 2000). En este trabajo se empezaba con el descubrimiento de relaciones de clase en reglas de clasificación. Se consideraban tres relaciones básicas: intersección de clases, exclusividad mutua y similaridad. Para cada posible relación definen una medida para evaluar el grado de intersección, de exclusividad mutua y similaridad entre las clases. Una vez que se han hallado las medidas, utilizan técnicas simples para crear las relaciones jerárquicas (Taxonómicas).

En (Omelayenko, 2001) se realiza una clasificación de los algoritmos de machine learning más potenciales para su aplicación para el aprendizaje de ontologías. Esta clasificación se divide en cuatro grandes grupos:

Aprendizaje de reglas proposicionales. Estos algoritmos aprenden reglas de asociación y reglas atributo-valor. Los algoritmos de aprendizaje se basan en árboles de decisión, representados normalmente con C4.5 (Quinlan 1993), y sus modificaciones se utilizan para proporcionar estas reglas de asociación a partir de un conjunto de instancias para el entrenamiento del algoritmo.

Aprendizaje Bayesiano: Se suele representar por los clasificadores nativos de Bayes. Se basan en el teorema de Bayes y genera reglas probabilísticas atributo-valor basada en la suposición de independencia condicional entre atributos de las instancias de entrenamiento.

Aprendizaje de reglas de lógica de primer orden: Los algoritmos suelen pertenecer a la familia de algoritmos FOIL.

Los algoritmos de Clustering agrupan las instancias basadas en la similaridad o la distancia de medidas entre un par de instancias definidas en términos de los valores de los atributos. La aplicación iterativa del algoritmo produce estructuras jerárquicas de los conceptos

Así, el autor realiza un pequeño estudio sobre varios trabajos y los clasifica según el tipo de ontología, la tarea de aprendizaje y el algoritmo de machine learning utilizado.

En (Maedche & Staab, 2001), se presenta una arquitectura de ontology learning para la Web Semántica. Esta arquitectura incluye un sistema para la construcción de ontologías a partir de texto denominada Text-to-Onto (Maedche & Staab, 2000). Este sistema utiliza un algoritmo de aprendizaje de reglas de asociación (association-rule-learning) para descubrir relaciones entre conceptos dejando la decisión final de la creación de los nuevos conceptos y la inserción de las relaciones al ingeniero de conocimiento. El sistema utiliza un algoritmo para descubrir reglas de asociación generalizadas. Los datos de entrada están formados por frases y el conjunto de conceptos

que aparecen juntos en la frase. El algoritmo extrae las reglas de asociación representadas por conjuntos de ítems que aparecen juntos a menudo y le presenta las reglas al ingeniero de conocimiento.

En (Navigli *et al,* 2003; Navigli & Velardi, 2004) se presenta una arquitectura para el aprendizaje automático de ontologías del dominio a partir de texto, llamada OntoLearn. El primer paso consiste en extraer la terminología a partir de un corpus de textos del dominio, después se filtra la terminología utilizando un procesamiento del lenguaje natural y técnicas estadísticas que permiten contrastar corpus de diferentes dominios. Después hace uso de la base de conocimientos léxica WordNet para interpretar los términos e identificar relaciones taxonómicas y otras relaciones semánticas (similaridad, partonómica). También utilizan un aprendizaje basado en reglas inductivas para extraer otras relaciones entre conceptos.

Muchas arquitecturas de extracción de conocimiento a partir de texto hacen uso de bases de conocimiento léxico-semánticas como WordNet (Millar, 1990) para inglés y EuroWordNet para otros idiomas europeos como italiano y español. Estas arquitecturas utilizan estas bases de conocimiento para identificar y verificar relaciones semánticas entre los términos adquiridos durante el proceso (Alani *et al*, 2003; Navigli *et al,* 2003)

Por último, cabe destacar que la integración de técnicas de machine learning puede reducir el tiempo y mejorar el desarrollo de ontologías del dominio por parte de los expertos (Webb *et al*, 1999)

1.4 PROBLEMA A RESOLVER EN ESTE TRABAJO

Dentro de las tecnologías del conocimiento, las ontologías se están erigiendo en la tecnología más importante para representar el conocimiento. Una muestra de la importancia actual de las ontologías es la cantidad de metodologías que han surgido para su creación, como se ha podido ver en las secciones anteriores, así como los estudios en creación de ontologías a partir de texto de una manera automática (Ontology Learning). La construcción de ontologías a partir de texto se está convirtiendo en una actividad importante con diversas finalidades. El creciente interés en esta actividad se debe principalmente a dos motivos: (1) la necesidad de sistemas para la gestión de conocimiento; y (2) el desarrollo de la Web Semántica.

Por lo que respecta a los sistemas de gestión de conocimiento, y en el marco de una organización, éste reside tanto en documentos como en mentes humanas y sistemas informáticos. En cuanto a la Web Semántica, el conocimiento se encuentra distribuido principalmente en páginas web, las cuales pueden adoptar múltiples formatos. Uno de los formatos más comunes en los que se redactan los documentos es lenguaje natural. Además, la construcción manual de ontologías es un cuello de botella que hay que intentar solventar mediante sistemas que ayuden a la construcción de ontologías directamente por los expertos sin necesitar la ayuda de un ingeniero de conocimiento.

Sin embargo, si analizamos las metodologías disponibles de construcción de ontologías a partir de texto, nos encontramos con diversos problemas que impiden la implantación de éstas como soluciones plausibles. El principal problema es que la mayoría de metodologías de construcción de ontologías utiliza relaciones taxonómicas exclusivamente, dejando de lado el resto de relaciones semánticas interconceptuales. Asimismo, en el caso de metodologías que permiten el uso de diferentes relaciones, su semántica no se expresa de manera formal. En esta investigación, se ha considerado que para que un modelo de representación posea aplicabilidad real, debe abarcar una pluralidad de relaciones interconceptuales, y no solamente taxonómicas.

En base a las carencias detectadas en las metodologías y sistemas de ontology learning, el

trabajo presentado en esta tesis se planteó con el fin de dar cuenta de los siguientes objetivos:

- Definición y formalización de una metodología para la extracción de ontologías a partir de texto que sea independiente del idioma, en la que se incluya la extracción de las relaciones semánticas descritas en este capítulo subsanando las carencias detectadas en los sistemas actuales de ontology learning. Más concretamente, se pretende la simplificación de los procesos de adquisición de conocimiento a partir de expertos, así como ofrecer una herramienta para solventar el problema caudado por el cuello de botella que supone actualmente la construcción de ontologías.

- Diseño e implementación de una aplicación software para la extracción de ontologías a partir de textos. Este sistema se diseñará para cumplir con los requisitos que implica el entorno de ontology learning descrito anteriormente.

- Validación del entorno y sistema anteriores. La metodología de validación se puede resumir a continuación: (1) validación de la metodología de construcción de ontologías en el dominio de la Web Semántica; (2) validación del sistema en el reconocimiento de órdenes en lenguaje natural.

1.5 RESUMEN

En este capítulo se ha descrito el estado del arte en ingeniería ontológica. La primera parte se centra en el origen de la noción de ontología. Se han presentado varias definiciones desde el punto de vista de la Inteligencia Artificial (IA), donde se distinguen dos corrientes principales. Por un lado, se considera que las ontologías definen los términos básicos y relaciones de un dominio particular. Por otro lado, la comunidad filosófica de la IA defiende la noción filosófica original de ontología y usan las ontologías de manera más formal. Actualmente, las ontologías son la tecnología más extendida para representar conocimiento en IA, porque proporcionan una representación estructurada y formal del conocimiento, y las representaciones ontológicas son compartibles y reutilizables.

Asimismo, se han propuesto varias clasificaciones de ontología siguiendo diferentes criterios. Una de tales clasificaciones distingue entre ontología formal y descriptiva. Mientras que las ontologías descriptivas están relacionadas con la recogida de información, las ontologías formales filtran, codifican y organizan los resultados de las ontologías descriptivas. En IA, las ontologías son pesadas cuando incluyen axiomas, y son ligeras en otro caso. Los axiomas son uno de los elementos usados para especificar ontologías. Otros elementos ontológicos importantes son: clases, relaciones, funciones e instancias. Se han propuesto diferentes modelos ontológicos. Dichos modelos se diferencian fundamentalmente en dos aspectos: (1) el tipo de ontología a modelar; y (2) la riqueza del modelo ontológico. Las relaciones son un elemento importante del modelo ontológico puesto que son instrumentos para relacionar entidades ontológicas. Entre las posibles relaciones, hemos distinguido la taxonomía (teoría de clasificación) y mereología (teoría de partes), y las relaciones de equivalencia, dependencia, topología, causalidad, funcional, cronológica, condicional y propósito.

Debido a la creciente importancia de las ontologías, se han propuesto diferentes mecanismos y metodologías para diseñar y construir ontologías. Algunos de ellos se han desarrollado para obtener ontologías particulares, mientras que otras metodologías tenían un propósito más general. Asimismo, se han tratado metodologías para la construcción automática

de ontologías a partir de textos que ayuden a la consecución de la Web Semántica e intenten salvar el gran problema del cuello de botella que supone la construcción manual de ontologías.

Con todo, uno de los objetivos de esta tesis es proporcionar una metodología para la construcción de ontologías de manera semi-automática (ontology learning) de modo que se alivie el problema del cuello de botella que supone la construcción manual de ontologías.

CAPÍTULO 2

PART-OF-SPEECH TAGGERS

2.1 INTRODUCCIÓN

La ingeniería lingüística es la aplicación del conocimiento de la lengua al desarrollo de sistemas informáticos capaces de reconocer, comprender, interpretar y generar lenguaje humano en todas sus formas. La ingeniería lingüística comprende una amplia gama de métodos, técnicas y herramientas; hace uso de una gran cantidad de recursos, lingüísticos y de otros tipos, y se utiliza de forma creciente en un gran número de aplicaciones.

A un nivel muy general, podemos considerar que una aplicación de ingeniería lingüística incluye alguna forma de comprensión del lenguaje humano, algún tipo de procesamiento de la información adquirida, y, eventualmente, una comunicación del resultado del proceso, a veces también utilizando alguna forma de lenguaje humano. Un ejemplo claro es el de un *sistema de traducción automática*, donde un texto en una lengua fuente es procesado hasta lograr un nivel de comprensión suficiente. La información extraída, expresada en algún lenguaje de representación adecuado es transferida al sistema de representación equivalente en la lengua objetivo, y, desde aquí, un proceso de generación logrará la traducción final en dicha lengua objetivo.

Por supuesto, no todas las aplicaciones de la ingeniería lingüística incluyen estos elementos. Así, un sistema de generación de cartas personalizado no precisa ningún tratamiento de comprensión, o un sistema de identificación de la lengua (o un detector de errores ortográficos) no necesitan generar lenguaje humano. La mayoría de las aplicaciones incluyen, sin embargo, alguna forma más o menos precisa de comprensión. Así, un sistema de consulta en lenguaje humano a una base de datos precisa un nivel muy alto de comprensión de las expresiones del interlocutor humano para que la respuesta del sistema sea de utilidad. En cambio, en un sistema de traducción o de resumen automáticos se pueden lograr niveles de corrección muy notables con niveles de comprensión bajos. Es decir, no es preciso comprender totalmente una oración para ser capaz de traducirla correctamente.

La inmensa mayoría de los sistemas que incluyen alguna forma de tratamiento de la lengua requiere la actuación de una serie de procesadores lingüísticos. La descripción lingüística del material a tratar, sea textual u oral, suele organizarse en forma estratificada: nivel fonológico, nivel fonémico, nivel textual, nivel morfológico, nivel léxico, nivel sintáctico, nivel lógico, nivel semántico, nivel pragmático, etc., y a cada uno de ellos se le suele asociar una colección de tales procesadores que deben resolver, en forma aislada o colaborativa, los problemas de tratamiento de la información propia del nivel. Así, en un sistema de comprensión de un documento escrito, el nivel textual del documento está constituido por los caracteres que aparecen en el mismo, y las tareas a llevar a cabo incluyen la separación del texto de la metainformación eventualmente presente, la segmentación del texto en unidades a tratar (esto es, párrafos, oraciones, etc.), la localización de las palabras ortográficas, etc. En el nivel sintáctico, los elementos a tratar son palabras dotadas ya de información morfosintáctica (categoría, propiedades morfosintácticas) y lo que se pretende es explicitar las relaciones sintácticas subyacentes.

Aunque, como se ha indicado antes, la complejidad del tratamiento, y, por lo tanto, la de los procesadores implicados, depende de la aplicación concreta, es evidente que los procesos iniciales, a saber, tratamiento textual, consulta en diccionarios, análisis y desambiguación morfosintáctica (POS-Tagging), etc., aparecen en prácticamente todos los sistemas. De estos procesos iniciales del tratamiento de los textos, previos al análisis sintáctico o a la interpretación semántica, nos ocupamos en este trabajo, que, además, queda limitado al ámbito del tratamiento de textos escritos.

Tradicionalmente, las aproximaciones utilizadas para el tratamiento de la lengua se basaban en el conocimiento lingüístico. Recientemente, sin embargo, han resurgido con éxito los sistemas empíricos basados, no en el conocimiento lingüístico. sino en la modelización del comportamiento lingüístico a partir de su constatación empírica. Este resurgir es debido a factores tales como la disponibilidad de recursos, por ejemplo los corpus lingüísticos, la capacidad de proceso de los sistemas informáticos y la necesidad de aplicaciones de tratamiento de textos no restringidos. La influencia del desarrollo de Internet ha sido decisiva a este respecto. Dentro de los métodos empíricos, cabe mencionar los de base estadística. Y. últimamente los basados en técnicas de aprendizaje automático (*Machine Learning*).

2.2 DEFINICIÓN DE POS-TAGGING

Uno de los elementos más importantes en todo sistema de procesamiento de lenguaje natural es el Part of Speech (POS) tagging (Morales-Carrasco and Gelbukh, 2003). Una de las posibles traducciones al español de Part of Speech Tagging es el de desambiguación morfosintáctica o análisis morfológico. El resultado del análisis morfológico es el de asignar a cada una de las unidades léxicas presentes (palabras dentro de una oración) el conjunto de sus categorías gramaticales posibles. Luego el objetivo de un POS tagging consiste en la asignación automática de descriptores, etiquetas, o tags, a cada palabra dentro de una oración, donde estas etiquetas corresponden a la categoría gramatical asociada a cada palabra.

Las categorías gramaticales han estado presentes en Lingüística durante mucho tiempo. Thrax (c.100 A.C) distinguió entre ocho tipos de palabras, usando las siguiente categorías: Nombre, verbo, participio, artículo (incluyendo los pronombres relativos), pronombre, preposición, adverbio y conjunción.

El problema del etiquetado se puede definir de la siguiente manera (Merialdo, 1994):

Suponemos que el usuario ha definido un conjunto de tags (asociadas a palabras). Consideramos una frase $P= p_1p_2...p_n$, *y una secuencia de tags* $T= t_1t_2...t_n$ *de la misma longitud. Llamamos al par (P, T) un alineamiento. Decimos que a la palabra* p_i *se le ha asignado el tag* t_i *en este alineamiento.*

Suponemos que los tags tienen algún significado lingüístico para el usuario, de modo que de todos los posibles alineamientos para una frase, sólo hay uno correcto desde un punto de vista gramatical.

Un procedimiento de tagging es un procedimiento ϕ que selecciona una secuencia de tags (define un alineamiento) por cada sentencia

$$\phi : P \longrightarrow T = \phi(P)$$

Existen (al menos) dos medidas para la calidad de un procesamiento de tagging:

A nivel de sentencia

$perf_s(\phi)$ = porcentaje de frases etiquetadas correctamente (percentage of sentences correctly tagged)

A nivel de palabra

$perw_s(\phi)$ = porcentaje de palabras etiquetadas correctamente (percentage of words correctly tagged)

En la práctica, el rendimiento a nivel de frase es generalmente menor que el rendimiento a nivel de palabra, dado que todas las palabras tienen que ser etiquetadas correctamente para la sentencia que se ha de etiquetar correctamente. La medida estándar usada en la literatura es el rendimiento a nivel de palabra.

Para la obtención de todas las palabras del texto objeto de tratamiento se hace uso de la tokenización, proceso que se podría definir como la segmentación de un texto en palabras-tokens individuales. Normalmente, en la práctica, dicho proceso suele realizarse suponiendo que una palabra ortográfica (separada por espacios, o signos de puntuación de las palabras adyacentes) es la unidad apropiada para el etiquetado de palabras. Sin embargo, hay excepciones a estos casos. Por ejemplo, en diversos idiomas, una palabra ortográfica puede estar formada por más de una palabra gramatical. En el caso del idioma inglés, las contracciones de

verbos (como en *she's*, *they'll*, *we're*) y las contracciones negativas (*don't*, *isn't*, *won't*) tienen asignadas dos palabras gramaticales en la misma palabra ortográfica. Además es bastante frecuente el caso contrario, donde dos o más palabras ortográficas tienen una sola palabra gramatical. Por ejemplo, en el idioma inglés, los adverbios formados por varias palabras *of course* e *in short*, y las preposiciones formadas por varias palabras como *instead of* y *up to*.

En la lengua castellana también ocurre que una palabra ortográfica pueda tener asociada más de una palabra gramatical, por ejemplo, palabras formadas por verbos y pronombres como *cógelo*, que tendría asociado dos palabras gramaticales, y *dímelo*, que tendría asociada tres. También ocurre lo contrario, que varias palabras ortográficas solo tengan una palabra gramatical asociada, por ejemplo, la conjunción *sin embargo*.

El uso de *lexicones computacionales* soluciona estos problemas. Con este término nos referimos a aquellos repositorios de información léxica elaborados con el objeto de servir de soporte representacional a diversas aplicaciones en el ámbito de las tecnologías del lenguaje humano, así como al trabajo lexicográfico tradicional, es decir, a la elaboración de diccionarios destinados a la consulta por un usuario humano.

La idea básica que subyace a dicho concepto es la de que cualquier tarea relativa al lenguaje natural que se pretenda llevar a cabo con el ánimo de conseguir resultados aceptables habrá de basarse y hacer uso extenso, indefectiblemente, de un lexicón correctamente diseñado, estructurado e implementado, rico en contenido de información léxica y que permita un fácil y rápido acceso a dicha información, tanto de forma directa por el usuario en el caso de las aplicaciones lexicográficas, como de forma interna, por una aplicación de traducción automática.

2.3 POS-TAGGING AUTOMÁTICO – ESTADO DEL ARTE.

Los tagger automáticos empezaron a crearse a finales de los 50, y la investigación en este área ha sido intensa durante las dos últimas décadas. Los problemas relacionados con la tokenización y el análisis léxico han sido importantes, pero la resolución de la ambigüedad es, para muchos autores, el subproblema más difícil en el análisis.

El problema de la desambiguación morfosintáctica consiste en que las palabras tomadas en forma aislada son ambiguas respecto a su categoría. Podemos considerar el siguiente ejemplo en inglés:

- The **light** is on. Aquí podemos observar que *light* es un nombre (luz).

- This is a **light** case. Sin embargo, en esta otra vemos que en este caso esta palabra lo que representa es un adjetivo (ligero).

- **Light** the fire. Por último, en este caso esta palabra representa un verbo (encender).

Como podemos observar en el siguiente ejemplo, lo mismo es aplicable a cualquier otro idioma, como español:

- Hoy **como** pollo. Aquí '**como**' actúa como **verbo**.

- La vida es **como** una caja de bombones. Y en cambio aquí '**como**' se comporta con un **adverbio**.

Un ejemplo más extenso de ambiguedad podemos verlo en el siguiente ejemplo Tabla 2-1, extraído de (Márquez *et al,* 2000):

The_DT first_JJ time_NN he_PRP was_VBD shot_VBN in_IN the_DT hand_NN as_IN he_PRP chased_VBD the_DT robbers_NNS outside_RB ._.							
First	time	shot	in	hand	as	chased	Outside
JJ	NN	NN	IN	NN	IN	JJ	IN
RB	VB	VBD	RB	VB	RB	VBD	JJ
		VBN	RP			VBN	NN
							RB

Tabla 2-1. Una frase y la ambigüedad. Los tags utilizados provienen del corpus Penn Treebank (Marcus *et al,* 1993).

Afortunadamente la categoría de la mayoría de las palabras no es ambigua dentro de un contexto, como puede verse en el ejemplo anterior. Así, con la ayuda de las palabras que no son ambiguas, se podrá inferir la categoría más probable de las ambiguas.

El objetivo de un desambiguador (también llamado *etiquetador morfosintáctico*) es el de asignar a cada palabra la categoría más *apropiada*, dentro de un contexto. Es decir, dada una secuencia de palabras, dotada cada una de un conjunto de etiquetas posibles, el desambiguador debe devolver una secuencia de etiquetas que sea la más *verosímil* dado el contexto. Por supuesto, la calidad del desambiguador depende del grado de precisión (la *granularidad*) del etiquetado, del contexto considerado y de la información de que disponga el desambiguador para considerar *apropiada* una etiqueta o *verosímil* una secuencia de etiquetas.

A veces, los desambiguadores no resuelven totalmente el problema de la ambigüedad gramatical y se limitan a eliminar las opciones imposibles o menos probables.

Existen tres grandes familias de etiquetadores: lingüísticos, estadísticos y los basados en Machine Learning.

Taggers Lingüísticos

Los primeros taggers automáticos que se desarrollaron contienen reglas creadas manualmente para el etiquetado, a veces con ayuda de un corpus ya etiquetado. Estos taggers utilizan conocimiento lingüístico, generalmente expresado en forma de reglas o restricciones para establecer las combinaciones de etiquetas aceptables o prohibidas. Las reglas suelen construirse manualmente, responden a criterios lingüísticos y se representan de forma explícita. Estas reglas de desambiguación se programan utilizando autómatas finitos o reglas ordenadas de patrones, en los que se realizaba un análisis de las posibles categorías gramaticales candidatas de la palabra actual, descartando todas sus alternativas hasta quedarse con la adecuada.

Dentro de estos sistemas podemos distinguir el sistema TAGGIT (Greene & Rubin, 1971), que se basa en reglas de patrones de contexto. TAGGIT usa un conjunto de etiquetas gramaticales con 71 elementos y una gramática de desambiguación de 3,300 reglas.

Los actuales taggers lingüísticos todavía representan el conocimiento como un conjunto de reglas (restricciones) escritas por lingüistas y obtenidas por introspección y herramientas estadísticas disponibles para el estudio del corpus. Sin embargo, los modelos actuales son mucho más expresivos, fáciles de comprender y son usados en algoritmos de desambiguación muy eficientes. Se trata de sistemas de muy alta precisión que contienen miles de reglas y normalmente requieren años de construcción y, por lo tanto un coste de desarrollo muy alto. Por ejemplo, el tagger EngCG (Karlsson *et al*, 1995), implementa las denominadas *Constraint Grammars* para el inglés, y comprende un millar de reglas, superando el 99,5 % de acierto.

Taggers estadísticos

A partir de los años 70 se fue dejando de lado la construcción de POS-taggers que hacían uso de reglas de desambiguación, dando paso a los taggers estadísticos, que obtienen los modelos del lenguaje y generalizaciones en que basan su actuación automática a partir de la evidencia empírica obtenida de corpus lingüísticos voluminosos. El coste de desarrollo de estos taggers es, por ello, mucho menor, aunque también es menor su grado de precisión, superior en cualquier caso al 97 %, lo cual es suficiente en algunas aplicaciones. Los sistemas son independientes de la lengua y fácilmente transportables a lenguas y dominios diversos con un coste limitado. El problema de estos sistemas reside en el aprendizaje (estimación) de los parámetros del modelo estadístico utilizado.

La desambiguación en el tagger CLAWS (Garside, Leech, & Sampson, 1987) se basa en elegir la categoría correcta en una base de evidencias estadísticas del corpus. En este sistema, la desambiguación se lleva a cabo por un módulo llamado (CHAINPROBS), que elige la etiqueta más probable de una palabra ambigua utilizando probabilidades de la ocurrencia de esta palabra con la categoría gramatical asociada en un contexto local dado. Este tagger fue la versión probabilística de TAGGIT y, posteriormente, fue mejorado por DeRose (1988) utilizando programación dinámica.

Dentro de los modelos probabilísticos se han utilizado los bigramas en los que la probabilidad de una etiqueta se estima simplemente con el contexto de la etiqueta anterior, y los trigramas en los que el contexto abarca a las dos etiquetas precedentes, como utiliza el tagger realizado por Church (1988).

En la actualidad existen taggers estadísticos utilizados ampliamente en todo el mundo como puede ser TnT Tagger (Brants, 2000), que hace uso de Hidden Markov Models (HMM) y de probabilidades léxicas o el conocido MXPOST (Ratnaparkhi, 1996), basado en un modelo de máxima entropía y diversas medidas contextuales.

Taggers basados en Machine Learning

Debido a que la mayoría de taggers estadísticos necesitan un entrenamiento, ya sea supervisado o no supervisado, se podrían clasificar como un tipo de "machine learning". Se clasifican dentro de esta categoría, aquellos taggers que incorporan métodos que adquieren información más sofisticada y que utilizan los paradigmas del "machine learning".

Se han utilizado técnicas de aprendizaje supervisado partiendo de corpus desambiguados manualmente y técnicas de aprendizaje no supervisado en las que no es precisa (o está limitada) esa intervención manual.

Dentro de las técnicas de machine learning utilizadas, se pueden destacar la inducción de reglas implementada en el tagger en (Brill 1992, Brill & Resnik. 1994, Brill 1995) que aprende automáticamente una serie de reglas de trasformación que puedan reparar mejor los errores cometidos por un etiquetado basado en asignar las etiquetas mas frecuentes de esas palabras.

Otra técnica ampliamente empleada son los árboles de decisión, utilizados, por parte, en (Márquez *et al*, 2000) y TreeTagger (Schmid, 1994b), los cuales hacen uso de árboles de decisión C4.5 (Quinlan 1993). También se ha utilizado el aprendizaje basado en casos (Daelemans *et al*, 1996; Zavrel & Walter, 1999), y redes neuronales (Schmid 1994a, Federici & Pirrelli, 1994).

2.4 ETIKET@

Dentro del idioma español existen muy pocos taggers específicos, utilizándose en la mayoría de los casos algoritmos que son independientes del idioma y a los cuales se les introduce un corpus de entrenamiento. Entre estos algoritmos se encuentran todos los anteriormente citados para el Inglés y, más concretamente, en (Morales-Carrasco & Gelbukh 2003) se hace un experimento en el cual se prueba el TnT tagger en el idioma español dando un nivel suficiente de acierto para tareas de reconocimiento del lenguaje natural.

Los corpus de entrenamiento para el idioma español de los taggers independientes del idioma no están disponibles para su uso, y crear estos corpus según el formato introducido es una tarea compleja y manual.

Es por eso por lo que se ideó la realización de un POS Tagger independiente del idioma, para su uso en tareas de procesamiento del lenguaje natural, y más concretamente para su uso en el sistema de construcción de ontologías que se ha perseguido durante el desarrollo de esta tesis.

2.5 ALGORITMO QUE UTILIZA ETIKET@

El funcionamiento del tagger desarrollado utiliza árboles de decisión C4.5 (Quinlan 1993) para la desambiguación morfológica y su funcionamiento es bastante similar a TreeTagger (Schmid, 1994b) y a la aproximación basada en árboles de decisión presentada en (Márquez *et al*, 2000).

El tagger en cuestión se basa en dos fases:

- Una primera fase (fase de búsqueda), en la que se obtiene la categoría gramatical más probable de cada palabra utilizando un lexicón.

- Una segunda fase (desambiguación morfológica) para asegurar que la categoría gramatical de esa palabra es la correcta en ese contexto.

2.5.1 Fase de búsqueda

Muchos tagger utilizan un lexicón para obtener a priori la categoría gramatical más probable para cada palabra de cada frase (Schmid, 1994b; Cutting *et al,* 1992)

eTiKeTa utiliza un lexicón para el mismo propósito, basado en una tabla dentro de una base de datos que se va actualizando y ampliando conforme va entrenándose el tagger. Está formado por una base de datos en la cual se indica la palabra, la categoría gramatical asociada, y la frecuencia de aparición de esa palabra con esa categoría gramatical.

El lexicón utilizado por TreeTagger (Schmid 1994b) está formado por tres partes: un lexicón de formas completas (fullform lexicon), un lexicón de sufijos (suffix lexicon) y una

entrada por defecto (defaul Entry). TreeTagger consulta en primer lugar el primer lexicón y, si no encuentra nada, consulta el segundo, por si se pudiese determinar la categoría de la palabra por su sufijo. Por último, devuelve el valor de la categoría más probable.

Por el contrario, el lexicón descrito en esta memoria, sólo está formado por uno similar al primer lexicón de TreeTagger. En caso de que no encuentre la palabra en el lexicón, se le asocia la categoría gramatical **Other**.

Dentro del lexicón utilizado por eTiKeT@ existen palabras que solo pueden tomar una categoría gramatical y que suelen aparecer con mucha frecuencia en todos los textos. Estas palabras se denominan **Stopwords** y suelen ser determinantes, artículos, pronombres, conjunciones, interjecciones, preposiciones y partículas. Por ejemplo **The** es una Stopword para el idioma inglés que sólo puede tomar la categoría gramática de determinante, así como su homóloga en español **el**.

2.5.2 Fase de desambiguación

Una vez que la fase de búsqueda ha finalizado, cada palabra tiene asociada la categoría gramatical, con la que suele aparecer más en los textos, pero no tiene por qué ser la correcta. Es por eso por lo que esta fase es necesaria para la correcta categorización de las palabras.

Esta fase hace uso de un árbol de decisión (ver punto siguiente) que devuelve la categoría gramatical asociada a la palabra según el contexto donde aparece.

En este momento pueden suceder cuatro cosas:

- *La palabra es una Stopword*: Debido a que estas palabras poseen una única categoría, prima la categoría gramatical obtenida en la fase de búsqueda.

- *La categoría gramatical en la fase de búsqueda es la categoría Other:* Como esta palabra no aparece en el lexicón, es etiquetada en base a la solución obtenida por el árbol de decisión.

- *La categoría gramatical es la misma que la de la fase de búsqueda*: En este caso, la palabra queda clasificada con la categoría gramatical correspondiente.

- *La categoría gramatical es distinta*: En este caso, se consulta de nuevo al lexicón para ver si la palabra en cuestión puede aparecer con la categoría inferida por el árbol de decisión.

    - *Si aparece en el lexicón con la nueva categoría*, la categoría asignada es la inferida por el árbol de decisión.

    - *En caso contrario*, se asigna la categoría de la fase de búsqueda.

2.5.3 Construcción del árbol de decisión.

A diferencia de TreeTagger y otros algoritmos que utilizan ngramas, eTiKeT@ no sólo hace uso de las categorías gramaticales de las palabras precedentes a las que se quiere evaluar, sino que también tiene en cuenta las categorías de las palabras que siguen a aquellas, como también se hace en (Márquez *et al,* 2000). En particular, eTiKeT@ tiene en cuenta el contexto (categorías gramaticales de las palabras anteriores y posteriores) en el que se encuentra la palabra para decidir la categoría gramatical de ésta. Este contexto es configurable en la aplicación que implementa eTiKeT@ en base a dos parámetros *nprew* (number of precedent Words) y *nposw* (number of posterior words).

Así el ngrama se define de la siguiente manera:

$$Tag_{-i} = t,\ i \in \{1 \ldots nprew\};\ t \in T$$

$$Tag_{j} = t,\ j \in \{1 \ldots nposw\};\ t \in T$$

donde *T* representa el conjunto de Tags

El árbol de decisión C4.5 se construye a partir de los ngramas introducidos en el sistema durante la fase de entrenamiento. A cada ngrama va asociado un peso que indica su frecuencia de aparición y la categoría gramatical de la solución.

La figura 2-1 muestra un ejemplo descriptivo de un árbol de decisión. Este árbol consta de dos niveles. En el primer nivel se pregunta por la categoría de la palabra anterior a la actual y en el segundo por la categoría de la palabra posterior. Como se puede observar, este árbol concluye con la categoría más probable a partir de las categorías de las palabras precedentes y posteriores, y no con probabilidades (como es el caso de TreeTagger, por ejemplo). Una conclusión de este árbol es que si la categoría de la palabra anterior es un Determinante, entonces la categoría de la palabra actual debe ser un nombre.

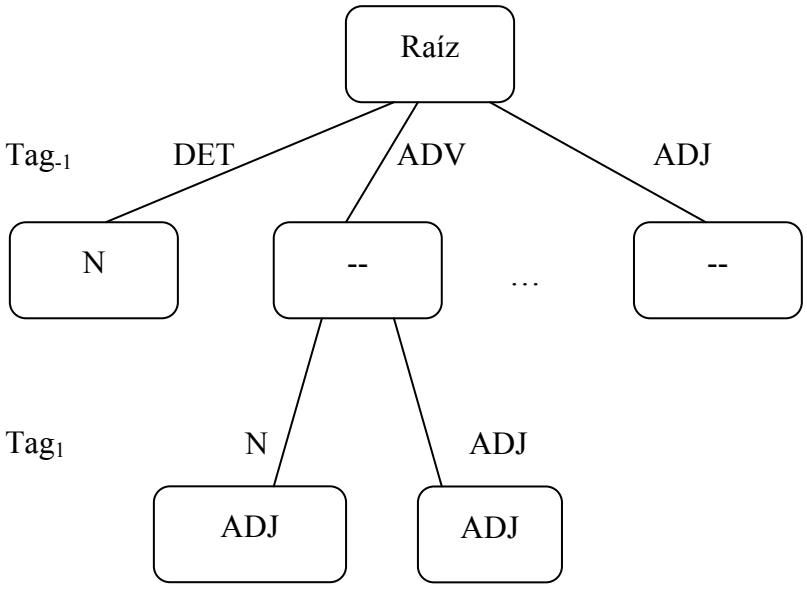

Figura 2-1. Un ejemplo de un árbol de decisión utilizado por eTiKeT@

2.5.4 Modos de funcionamiento

Como se ha comentando anteriormente, eTiKeT@ nació con la idea de simplificar las tareas de entrenamiento de los tagger disponibles, sin tener necesidad de utilizar corpus ya etiquetados para su entrenamiento, pudiendo entrenarse el sistema mediante textos, donde el usuario es el que va diciendo al sistema la categoría gramatical de cada frase. Es por eso por lo que eTiKeT@ tiene dos modos de funcionamiento:

Modo Entrenamiento

En este modo la aplicación va seleccionando automáticamente frase a frase en un texto dado para que el usuario le indique las categorías gramaticales de las palabras de cada frase con lo que se puede ir entrenando al tagger de forma **interactiva**. En este modo de ejecución, la aplicación induce las categorías gramaticales de cada palabra, y, cuando no puede inferir alguna, la categoriza como **Other**. El usuario puede ir modificando las categorías inducidas, o bien, aceptar como buena esa palabra.

En la figura 2-2 se muestra un volcado de pantalla en la que aparecen las frases etiquetadas con opción de modificar las categorías gramaticales de cualquier palabra.

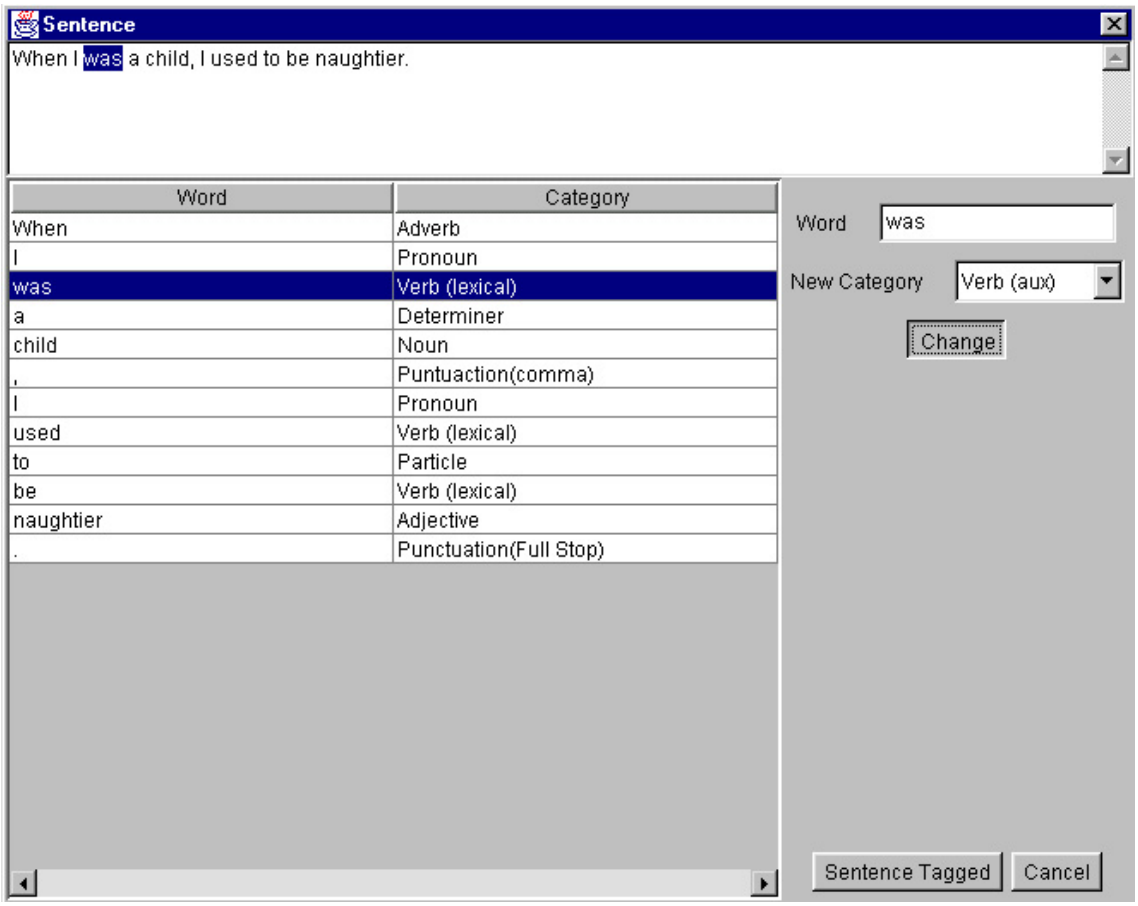

Figura 2-2. Etiquetado morfológico de una frase con eTiKeT@.

Modo normal de uso

En este modo de utilización, se le pasa a la aplicación como parámetro un fichero de texto con las frases a etiquetar y este devolverá el texto categorizado.

Por ejemplo, si el texto está formado por la siguiente frase:

El cáncer de piel es una enfermedad producida por el desarrollo de células cancerosas en las capas exteriores de la piel.

El sistema devuelve la siguiente categorización.

El [det] cáncer [nombre] de [prep] piel [nombre] es [verbo (auxiliar)] una [det] enfermedad [nombre] producida [verbo (léxico)] por [prep] el [det] desarrollo [nombre] de [prep] células [nombre] cancerosas [adj] en [prep] las [det] capas [nombre] exteriores [adj] de [prep] la [det] piel [nombre]

2.6 RESULTADOS

Como se ha indicado anteriormente, la medida de rendimiento más utilizada en la literatura sobre el tema es el rendimiento a nivel de palabra, esto es, el porcentaje de palabras etiquetadas correctamente.

El corpus utilizado para el entrenamiento se ha extraído de diferentes artículos dentro del dominio de la oncología. Este corpus está formado por 6400 tokens y ha sido etiquetado a través de diferentes sesiones de entrenamiento, obteniendo al final sobre un 85,23 % de rendimiento con los parámetros *nprew* = 2 y *nposw* = 2.

En la figura 2-3 podemos observar cómo el rendimiento varía conforme aumenta el número de tokens con los que el sistema se entrena.

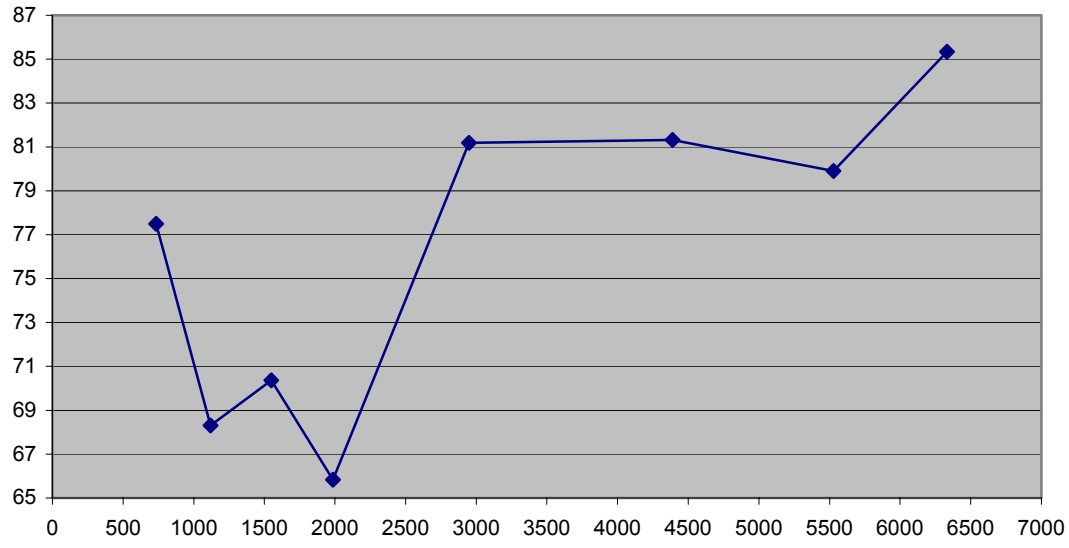

Figura 2-3. Gráfica del ratio de acierto respecto de las palabras analizadas.

Muchos taggers han obtenido resultados superiores al 90%, como se muestra en (Brill and Wu, 1998), donde se comparan los resultados entre un tagger unigrama, un trigrama, el

transformation-based tagger (Brill, 1995) y el MXPOST (Ratnaparkhi 1996) entrenándolos con 1,1 millón de palabras (ver Tabla 2-2).

Tagger	Accuracy (%)
Unigram	93,26
Trigram	96,36
Transform.	96,61
Max. Ent.	96,83

Tabla 2-2. Resultados de eficiencia de diferentes taggers (tabla extraída de Brill & Wu,1998)

Debido a la cantidad (1,1 millón) de palabras con las que se han entrenado los taggers para la obtención de los resultados mostrados en la tabla 2-2, los resultados no pueden compararse con el tagger descrito en este capítulo.

Las principales ventajas de eTiKeT@ frente a los tagger convencionales son las siguientes:

- Que obtiene un ratio de acierto bastante alto con corpus pequeños. Esto es muy interesante porque permite entrenar el tagger en dominios específicos con pocas palabras y que obtenga un ratio de acierto aceptable.

- Debido a su modo de entrenamiento amigable, se puede considerar como un generador de tagger aplicable a dominios específicos de una manera fácil y amigable para cualquier usuario. Además, gracias a la metodología seguida por este sistema, se pueden crear lexicones de dominios específicos (con la fase de entrenamiento del sistema) que después pueden ser importados en otras aplicaciones de procesamiento de lenguaje natural.

2.7 RESUMEN

En este capítulo se ha descrito la importancia de la ingeniería lingüística, y más concretamente de los POS-Taggers, en las tareas de reconocimiento de lenguaje natural, describiendo el estado del arte en POS-Tagging. Una posible clasificación de estos taggers es la siguiente:

- Taggers Lingüísticos: Fueron los primeros taggers que se crearon y se basan en el uso de un conocimiento lingüístico generalmente expresado en forma de reglas creadas manualmente.

- Taggers Estadísticos: Obtienen los modelos del lenguaje y las generalizaciones en que basan su actuación automática a partir de la evidencia empírica obtenido de corpus lingüísticos voluminosos. Actualmente, estos tagger son utilizados ampliamente en todo el mundo.

- Taggers basados en Machine Learning: Incorporan métodos de adquisición de información más sofisticada que utilizan los paradigmas de "machine learning", tanto supervisado como no supervisado.

A continuación, se describe un tagger basado en Machine Learning que utiliza el paradigma de los árboles de decisión C4.5. Este tagger fue desarrollado debido básicamente a dos razones. En primer lugar, la constatación de la existencia de un número muy reducido de taggers específicos para español. En segundo lugar, se ha comprobado tras el análisis del estado del arte, que los algoritmos independientes del idioma necesitan corpus de entrenamiento que no están disponibles para su uso.

Por último, se muestra una pequeña validación del tagger desarrollado.

CAPÍTULO 3

UN ENTORNO PARA ONTOLOGY LEARNING

3.1 INTRODUCCIÓN

En este capítulo se presenta una metodología para generar conocimiento a partir de texto. Más concretamente, se presenta un sistema de ontology learning basado en técnicas de machine learning para la extracción de ontologías del dominio.

Este sistema trata de descubrir términos que representen conceptos mediante la búsqueda en una base de conocimiento que relacione estos conceptos con las expresiones lingüísticas que puedan representar esos conceptos. Además, el sistema permite identificar relaciones entre los conceptos, así como inferir nuevas entidades de conocimiento. La idea principal en la que se basa este trabajo es el hecho que las relaciones entre las entidades de conocimiento (sobre todo entre conceptos) están asociadas normalmente a expresiones verbales. Es por eso por lo que el sistema que se presenta permite almacenar las expresiones verbales que representan relaciones entre conceptos, con la finalidad de ser capaz de identificar automáticamente este conocimiento cuando reaparezca.

El sistema utiliza dos tecnologías de conocimiento, así como las categorías gramaticales de las palabras de la frase actual para inferir las entidades de conocimiento (conceptos, atributos y valores) que aparezcan en esta frase y construir una ontología del dominio a partir de un fragmento de texto. Las tecnologías de conocimiento utilizadas son Razonamiento Basado en Casos (CBR en inglés), que consiste en la utilización de situaciones anteriores para resolver los problemas actuales y ontologías. En particular, en lo que respecta a CBR, se hace uso de RDR(Compton *et al*, 1989). La mayoría de las metodologías CBR presentan problemas de mantenimiento. Sin embargo, RDR soluciona este problema y permite a los expertos construir y mantener bases de conocimiento de manera sencilla. MCRDR (Kang, 1996) es una extensión de RDR que permite trabajar con conclusiones múltiples.

El subsistema MCRDR permite la obtención de una especie de patrones léxico-sintácticos para traducir el conocimiento a partir de una frase de manera automática. Como se ha mencionado en el primer capítulo, en (Hearst, 1992) se utilizan patrones léxico-sintácticos

manuales para la extracción de relaciones taxonómicas. A través de nuestra aproximación, se pretenden obtener de manera automática estos patrones para la detección de diferentes relaciones semánticas, no sólo las taxonómicas.

3.2 SUPUESTOS DE PARTIDA

En este capítulo se presenta una metodología para la construcción de ontologías a partir de texto en lenguaje natural. Esta metodología se basa en los siguientes supuestos:

- Las ontologías pueden utilizarse para representar conocimiento. Como el texto completo puede ser muy largo será conveniente dividirlo en fragmentos más pequeños para facilitar su procesamiento. Además, este trabajo se basa, como se ha dicho antes, en la idea que las relaciones suelen estar asociadas a expresiones verbales en lenguaje natural, por lo que es conveniente dividir el texto en frases.

- Los expertos en un dominio pueden extraer conocimiento a partir de un texto de ese dominio.

- Los textos contienen conocimiento. Pueden aparecer entidades de conocimiento a lo largo del texto de una manera explícita, aunque a veces el conocimiento puede estar expresado de manera implícita. (La metodología propuesta aquí tratará de encontrar sólo el conocimiento explícito en el texto).

3.3 LA METODOLOGÍA MCRDR

3.3.1 Introducción

Ripple-Down Rules (RDR) (Compton *et al*, 1989; Compton *et al*, 1998) es una metodología utilizada para construir sistemas basados en conocimiento y sistemas expertos tiene su origen en los problemas (de mantenimiento) encontrados en el GARVAN-ES1, uno de los primeros sistemas expertos médicos usados en el mundo (Buchanan, 1986). GARVAN-ES1 se usaba para dar una interpretación clínica según los informes sobre mediciones de tiroxina en la sangre (Horn *et al*, 1985). El dominio manejado por GARVAN-ES1 lo abarca actualmente PEIRS (*Pathology Expert Interpretive Reporting Systems*), que es un sistema de patología clínica más genérico. En este sentido, PEIRS añade comentarios a los informes. El sistema da como conclusión distintas patologías interrogando a los pacientes, de modo que la mayoría de los informes no requieren ningún comentario. Esto representa una mejora importante, ya que el sistema incrementa la calidad de la información de salida del laboratorio y también reduce la carga de expertos, porque comprobar una interpretación resulta mucho más sencillo que componerla y realizarla manualmente.

Hay dos versiones de PEIRS: una interactiva, que es utilizada por el experto que añade reglas, y otra (no interactiva) que funciona como un proceso por lotes. Ambos sistemas usan la misma base de conocimiento.

Entre las limitaciones de estos sistemas, está que el refinamiento es local y no se hace ningún esfuerzo por contrastar el orden en el que los casos son añadidos al sistema, pudiendo haber muchas repeticiones locales de conocimiento. PEIRS no ha sido todavía evaluado, pero con el desarrollo de GARVAN-ES1/RDR se comprobó que la repetición no era un problema significativo, ya que el conocimiento repetido suponía un porcentaje muy bajo dentro del total.

3.3.2 Desarrollo de RDR

La metodología RDR se desarrolló a partir de la experiencia con los problemas de mantenimiento de GARVAN-ES1. La principal observación que se obtuvo del mantenimiento de GARVAN-ES1 es que el experto no decía cómo llegaba a esa conclusión, ni explicaba su proceso de razonamiento, sino que más bien indicaba por qué una conclusión era correcta. La justificación que se daba dependía de un contexto en particular. Sin embargo, esta metodología es especialmente buena distinguiendo un caso de otro y justificando por qué la clasificación no es la misma. Si un experto dice que el sistema RDR ha producido una clasificación errónea para un caso, el sistema encuentra el caso motivante del error, esto es, el que provocó la regla, digamos R1, fuera añadida, siendo dicha regla la causante de la conclusión errónea para el caso actual. Este caso se llama *caso angular* de la regla R1, y lo codificaremos como CA1.

RDR muestra la diferencia al experto entre dos casos, el caso CA1 y el caso actual, escrito CA, y le pide a éste que seleccione algunas condiciones de la lista de diferencias que formarán parte del antecedente de la nueva regla y que justifican la nueva conclusión. El sistema añade una nueva regla R2 (en la base de conocimiento), que se evalúa sólo si se satisface CA1. RDR añade R2 como un refinamiento de la regla previa R1. Este refinamiento en particular se usa sólo bajo las circunstancias en las cuales la nueva conclusión se añadió. Así pues, la base de conocimiento tiene una estructura binaria, ya que el conocimiento se añade como una excepción a la regla anterior sin alterar el conocimiento de la regla previa.

RDR no distingue la fase de adquisición de conocimiento inicial de la fase de mantenimiento, pues esta metodología parte de una base de conocimiento vacía a la que el experto va añadiendo reglas cada vez que éste encuentra un caso que no es clasificado correctamente. Dicho de otro modo, RDR simplemente elimina el período de adquisición inicial y usa la fase de evaluación para construir la base de conocimiento.

3.3.2.1 Estructura de la base de conocimiento generada por RDR

La estructura de la base de conocimiento puede representarse como un árbol binario donde cada regla es del tipo *if-then* y cada nodo se almacena junto al caso angular que fue el causante de que la regla se añadiera. Las condiciones en la regla se unen mediante la conjunción de éstas (Figura 3-1).

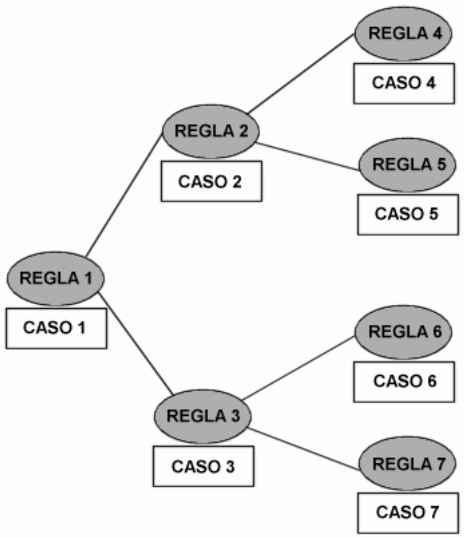

Figura 3-1. Estructura de RDR

3.3.2.2 Inferencia en RDR

El proceso de inferencia comienza por el nodo raíz, de forma que cada regla transversal se evaluará por medio del caso de entrada que se le dé al sistema (*caso actual*).Como la estructura es de un árbol binario, habrá sólo dos ramas. Si el caso actual satisface la regla, entonces se sigue el proceso de inferencia por la rama que indique *verdadero*. De otro modo, se iría por la rama que indica *falso*. Cuando el proceso de inferencia se completa (no hay más nodos transversales), la conclusión de la última regla satisfecha es la que se le da al sistema como conclusión del caso.

3.3.2.3 Creación de la base de conocimiento basada en casos en RDR

Una de las principales ventajas de RDR es que corregir un error en la base de conocimiento se hace solamente añadiendo una regla como una excepción más bien que como una regla de cambio de conclusión en la base del conocimiento. Cuando el sistema añade una nueva regla para corregir la base del conocimiento, el sistema necesita averiguar dos cosas: las condiciones del antecedente y su conclusión, por una parte, y, por otra, una localización en la base de conocimiento para la nueva regla.

Cuando el sistema proporciona la clasificación para el caso actual, hay un trazo de reglas que muestra por qué proporciona la clasificación dada para el caso en cuestión, y el sistema también tiene el caso angular, que se usa para añadir la regla a la base de conocimiento. El trazo de reglas abarca no sólo las reglas que fueron satisfechas por el caso, sino el conjunto de secuencias de reglas que se evaluaron y si la evaluación fue verdadera o falsa. Así pues, el trazo de reglas proporciona la localización para la nueva regla y las diferencias entre el caso actual y el caso angular de la última regla evaluada como verdadera en el trazo de reglas, ayudando así al experto a seleccionar las condiciones para la nueva regla.

En un sistema convencional, el experto en el dominio decide la localización de la regla, y no es aconsejado de las condiciones necesarias que tiene que incluir en la regla para prevenir conflictos. Añadir la nueva regla al final del trazo de reglas actual posibilita al sistema para que se use la regla en el mismo contexto en el que fue suministrada. Con todo, el experto proporciona la regla, la cual sólo se evaluará para otros casos que aparezcan bajo el mismo contexto, si se alcanza la misma secuencia de reglas evaluadas de nuevo. Para componer dicha regla, el sistema pide al experto que seleccione las condiciones de la lista de diferencias entre el caso actual y el caso angular. El caso actual tiene el mismo trazo de reglas que el caso angular, así que hay que identificar, de alguna manera, lo que distingue a ambos casos. Las condiciones comunes también pueden incluirse en la nueva regla como elemento clarificador. La lista de diferencias se construye de la siguiente manera:

{x | (x ∈ caso de entrada) & (x ∉ caso angular)} ∪ {NOT (x)| (x ∉ caso de entrada) & (x ∈ caso angular)}

Finalmente, el sistema añade una regla al final del trazo de reglas con las condiciones seleccionadas por el experto de la lista de diferencias junto con algunas condiciones comunes de la intersección.

Los siguientes pasos detallan cómo el sistema RDR es usado para la adquisición de conocimiento cuando el experto no está de acuerdo con la clasificación dada por el sistema.

1. El experto ejecuta el caso para el cual se desea cambiar la clasificación del sistema.

2. El sistema recupera información sobre dos casos después de la inferencia: uno es el caso actual y otro es el caso angular. También se conoce el trazo de reglas.

3. El experto introduce la clasificación correcta.

4. El sistema muestra las diferencias entre los casos. El experto escogerá las condiciones para justificar la nueva clasificación. El experto puede elegir también condiciones para la nueva regla de entre las condiciones comunes a los dos casos.

5. El sistema compone la nueva regla a partir de las condiciones y la clasificación.

6. El sistema sitúa la nueva regla al final del trazo de reglas.

7. El sistema añade el caso actual como caso angular. Este nuevo caso es un caso angular de la nueva regla.

Por ejemplo, en la figura 3-2, antes de añadir la regla 120, la clasificación del caso actual proviene de la regla 20, ya que es la última regla activada en el trazo de reglas. La nueva regla es añadida al final de trazo de reglas con condiciones que el experto selecciona de la lista de diferencias. Si el experto selecciona, al menos, una de las condiciones que están en el caso actual, pero no en el caso angular, la nueva regla será satisfecha por el caso actual pero no por el caso angular. De otra manera, una de las condiciones será una condición negada que no existe en el caso actual pero que existe en el caso angular 20.

En la tabla 3-1 se muestra la lista de diferencias entre el caso angular de la última regla activada y el caso actual.

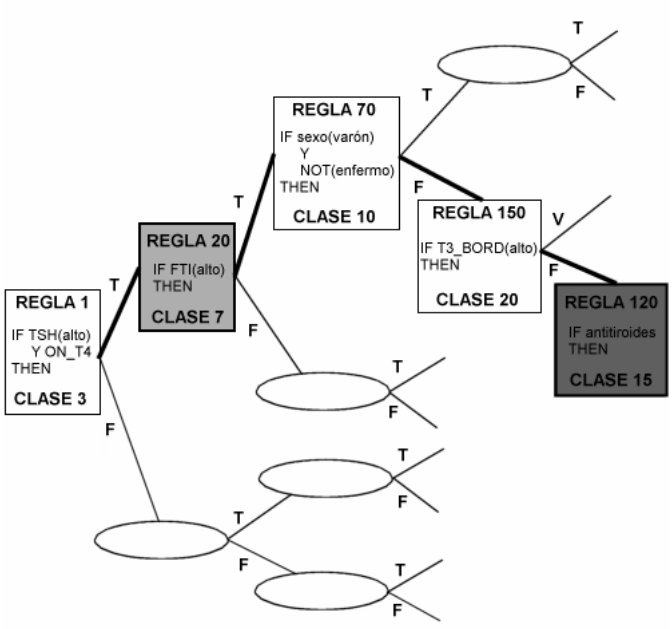

Figura 3-2. Proceso para añadir la regla 120

Caso Actual	Caso Angular 20	Lista de Diferencias
ON_T4	ON_T4	
ENFERMO	ENFERMO	
ANTITIROIDES		ANTITIROIDES
Sexo: varón	Sexo: varón	
TSH: alto	TSH: alto	
TSH_BORD: bajo	TSH_BORD: bajo	
T3: normal		T3:normal
T3_BORD: bajo		T3_BORD: bajo
	T3: no aparece	NOT (T3: no aparece)
	T3_BORD: no aparece	NOT(T3_BORD:no aparece)
FTI: alto	FTI: alto	
FTI_BORD: alto	FTI_BORD: alto	

Tabla 3-1: Lista de diferencias entre el caso angular de la última regla activada y el caso actual

3.3.3 MCRDR

La filosofía MCRDR se ha desarrollado para manejar problemas de clasificación múltiple. El objetivo de MCRDR es conservar las ventajas y la estrategia esencial de RDR pero manejando múltiples clasificaciones independientes. Así, MCRDR, al igual que RDR, se basa en la suposición de que el conocimiento que el experto proporciona es la justificación de una conclusión en un contexto particular. La componente principal del contexto es el caso para el cual se da una clasificación errónea y cómo difiere éste de los otros casos para los cuales la clasificación es correcta.

3.3.3.1 Estructura e inferencia en MCRDR

La operación de inferencia mediante RDR está basada en la búsqueda de bases de conocimiento representadas como una lista de decisión en la que cada decisión posible se refina de nuevo por medio de otra lista de decisión. En contraste, MCRDR evalúa todas las reglas en el primer nivel de la base de conocimiento. Después, se evalúan las reglas en el siguiente nivel de refinamiento para cada regla que se satisfizo en el nivel superior, y así en adelante. El proceso se detiene cuando no hay más hijos para evaluar o cuando ninguna de estas reglas puede ser satisfecha por el caso actual. De este modo, se obtienen múltiples trazos de reglas (en los que cada trazo representa una secuencia de refinamiento particular) y, por lo tanto, múltiples conclusiones.

La estructura de una base de conocimiento MCRDR puede representarse como un árbol n-ario, en el cual, cada nodo representa una regla (Figura 3-3).

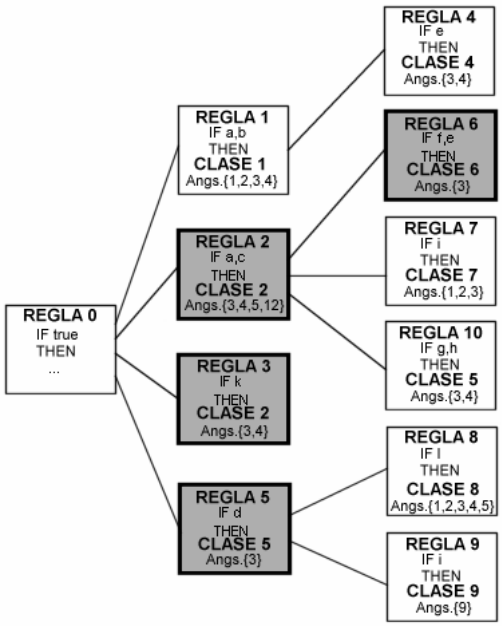

Figura 3-3. Base de conocimiento MCRDR. Las reglas marcadas son satisfechas por el caso {a,c,d,e,f,h,k}.

3.3.3.2 Adquisición de conocimiento usando MCRDR

Cuando un caso se ha clasificado incorrectamente o su clasificación es indeterminada (se asume que ninguna clasificación se hace por defecto), entonces la clasificación debe corregirse. La clasificación de conocimiento puede dividirse en tres partes:

1. El sistema consigue la clasificación correcta del experto.

2. El sistema decide sobre la localización de la nueva regla.

3. El sistema adquiere reglas nuevas del experto y las añade a la base de conocimiento en la posición correcta.

No obstante, es posible que el experto encuentre el sistema más natural si el orden de los pasos 2 y 3 se invierte. El orden no es crucial en términos del algoritmo.

3.3.3.3 Adquisición de una nueva clasificación

Adquirir una nueva clasificación es trivial. Por ejemplo, si el sistema produce las clasificaciones *clase 2*, *clase 5*, *clase 6* para un caso actual determinado, el experto puede decidir que la clasificación *clase 6* no necesita cambiarse, pero que las clasificaciones *clase 2* y *clase 5* deberían borrarse y añadir las clasificaciones *clase 7* y *clase 9* en su lugar. A pesar de que esto es conceptualmente fácil, en un dominio donde una clasificación se representa normalmente en una cadena de texto independiente y es posible un gran número de tales clasificaciones, resulta técnicamente problemático dar a un experto un soporte para encontrar y rehusar clasificaciones.

Es de gran utilidad para nuestro sistema que las clasificaciones sean reutilizables, de manera que el sistema pueda identificar cuándo dos o más clasificaciones idénticas se producen y sólo se informa de una de ellas.

3.3.3.4 Localización de reglas

El sistema debería encontrar la localización para las nuevas reglas de manera que este pueda proporcionar como resultado las nuevas clasificaciones. No podemos asumir que la localización correcta para una nueva regla sea un refinamiento de una de las reglas que dan la clasificación errónea, ya que puede ser una clasificación independiente y la clasificación de la regla sea simplemente incorrecta. Las posibilidades en cuanto a reclasificación aparecen en la tabla 3-2.

Clasificaciones erróneas	***Para corregir la base de conocimiento***
La clasificación errónea debe pararse	Añadir la regla (*regla de parada*) al nodo hoja en concreto para prevenir la clasificación
La clasificación errónea sustituida por una nueva clasificación	Se añade la regla al final del nodo en cuestión (como nodo hijo) para dar la nueva clasificación
Una nueva clasificación independiente	Añadir una regla en el nivel superior para dar la nueva clasificación

Tabla 3-2: Acciones posibles para reclasificar

Cabe destacar la idea de las *reglas de parada*, reglas de las cuales no se obtiene ninguna conclusión, o más bien dan una clasificación nula. Las reglas de parada juegan un papel importante en MCRDR en la prevención de clasificaciones erróneas para un caso actual dado.

Si hay una estrategia que tiende a añadir reglas al nivel superior, la base de conocimiento cubrirá el dominio más rápidamente, pero con una mayor posibilidad de error. Si la estrategia, en cambio, tiende a añadir reglas de manera horizontal, el dominio se cubrirá más lentamente, pero con menos posibilidad de error.

Los expertos deben poder tener la libertad de tomar cualquier tipo de decisión a este respecto, pero la interfaz debe diseñarse de manera que el experto tenga mayor posibilidad de añadir reglas de manera horizontal o vertical.

3.3.3.5 Adquisición de condiciones y validación de reglas

La técnica convencional para la evaluación de la representación de un sistema basado en conocimiento es usar una base de casos estándar. En esta situación, se depende de los casos que sean representativos y que se puedan manejar por el sistema. Cualquier regla que se añade debe sólo disparar los casos relevantes en la base de casos, y cualquier regla que cambie debe también posibilitar que la regla se continúe disparando en los casos apropiados. En general, cierto número de casos debe ser considerado.

Con MCRDR, al igual que con RDR, debe evitarse que una nueva regla se dispare en casos correctamente manejados por sus reglas padre. Sin embargo, a diferencia de RDR, MCRDR tiene múltiples casos angulares para una regla, lo cual puede ser un obstáculo para satisfacer la nueva regla.

Conforme el árbol se desarrolla, las reglas en niveles inferiores tendrán menos casos angulares asociados con ellas. El propósito de esto es hacer una nueva regla lo suficientemente precisa, de manera que satisfaga sólo el caso que se añade y no otros casos anteriormente almacenados, excepto que satisfaga casos que incluyen la misma clasificación. El algoritmo para seleccionar las condiciones del antecedente de manera que la regla sea lo suficientemente precisa es muy sencillo. Por su parte, el algoritmo que se usa para añadir una regla es el siguiente:

1. El sistema forma una lista de casos angulares que pueden alcanzar la nueva regla, de modo que se obtienen las diferencias con el caso presente. Se le pide al experto que seleccione algún elemento de la lista de diferencias entre el caso presente y uno de los casos angulares de la lista de casos que se consideran. El sistema entonces prueba todos los casos angulares de la lista por medio de las condiciones seleccionadas y borra de la lista aquellos casos angulares que no satisfacen las condiciones seleccionadas. Se le pide después al experto que elija algunas condiciones de entre las del caso actual y otro de los casos angulares que permanecen en la lista. Las condiciones seleccionadas se añaden al antecedente de la regla por medio de la conjunción de éstas. El sistema repite este proceso hasta que no haya casos angulares en la lista que satisfagan la regla.

2. Después, el sistema añade una regla con las condiciones seleccionadas y prueba el resto de los casos angulares asociados con la regla padre. De entre éstos, los casos que puedan satisfacer la nueva regla, acaban siendo casos angulares de la nueva regla.

3. Finalmente, el caso actual se añade a la lista de casos angulares asociados a las otras reglas que dieron una clasificación correcta para dicho caso.

4. El sistema está ahora preparado para clasificar otros casos.

3.3.4 Ontology Learning con RDR

Una de las primeras aproximaciones de ontology learning a partir de bases de conocimiento, se basa en el descubrimiento de relaciones jerárquicas entre clases en bases de conocimiento creadas con el paradigma RDR (Suryanto & Compton, 2000). En este trabajo se presenta una técnica para la obtención de ontologías, a partir de bases de conocimiento ya creadas y a una identificación previa de los atributos y valores más relevantes en la ontología.

Actualmente, no existe ninguna aproximación basada en RDR para ontology learning a partir de texto en lenguaje natural.

3.4 EL PROCESO DE ADQUISICIÓN DE CONOCIMIENTO.

El proceso de extracción de conocimiento a partir del texto se divide principalmente en tres fases secuenciales (Figura 3-4): POS-Tagging, fase de búsqueda de conceptos, fase de inferencia (basada en MCRDR).

Como se puede ver en la Figura 3-4, el sistema presentado es un sistema supervisado por un experto o un ingeniero de conocimiento para la construcción semiautomática de ontologías. El sistema va mostrando al usuario el conocimiento inferido de la frase actual y el usuario acepta o modifica ese conocimiento generando de manera semi-supervisada una ontología del dominio.

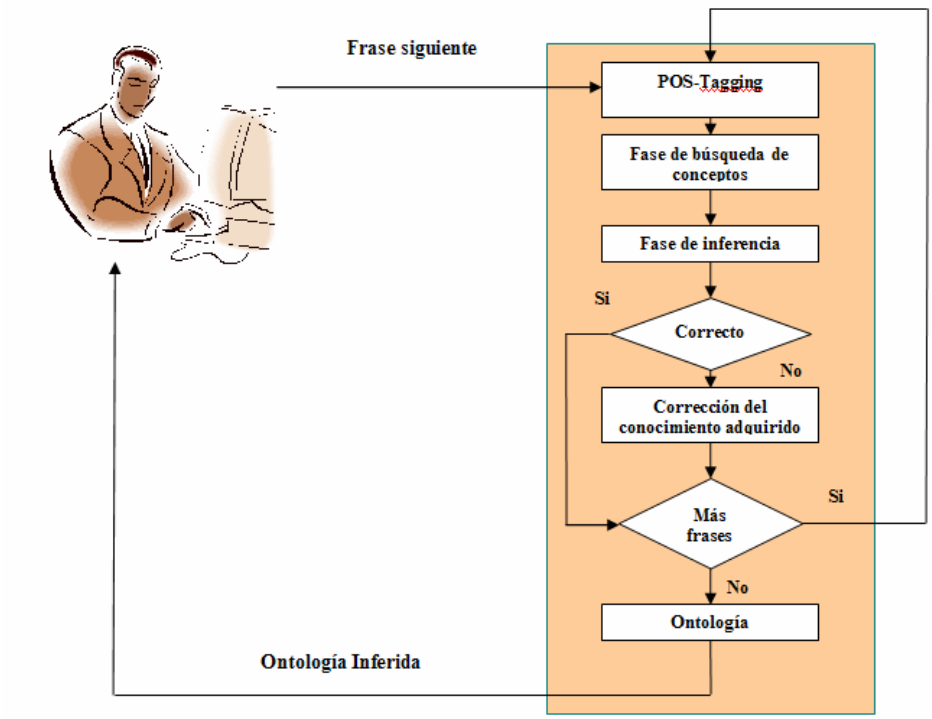

Figura 3-4. El sistema de adquisición de conocimiento

3.5 POS-TAGGING

El objetivo principal de esta fase, como se ha explicado en el capítulo anterior es obtener la categoría gramatical de cada palabra en la frase actual. Es por eso por lo que el sistema debe utilizar un POS-tagger para obtener la frase etiquetada con las categorías gramaticales correspondientes.

Nuestro sistema es independiente del idioma, y por eso hemos utilizado como tagger el descrito en el capítulo anterior.

3.6 FASE DE BÚSQUEDA DE CONCEPTOS

3.6.1 Introducción a la adquisición de términos.

En la actualidad existen muchas técnicas para la adquisición de términos en corpus lingüísticos (Jacquemin, 2001). La entrada usual a estos sistemas son corpus etiquetados y la salida es una lista con los términos candidatos que aparecen en el corpus.

ACABIT (Daille, 1996) utiliza un analizador híbrido compuesto de un analizador sintáctico y un filtro estadístico a partir de textos anteriormente etiquetados. El analizador es una máquina finita de estados que utiliza las categorías gramaticales del texto etiquetado y otras características morfológicas.

ANA (Enguehard & Pantera, 1995) se basa en una simplificación de textos y la observación de patrones recurrentes. Es un sistema incremental debido a que se enriquece con los nuevos términos descubiertos. Este sistema tiene tres conjuntos iniciales de datos: un

conjunto de *stop words*, un conjunto de *términos semilla* que son introducidos manualmente y contienen los conceptos más importantes en el dominio del corpus y un conjunto de *esquemas de palabras* utilizados para construir términos más complejos a partir de términos de una sola palabra.

LEXTER (Bourigault, 1996) utiliza una máquina finita de estados para obtener los sintagmas nominales máximos. Una vez extraídos estos sintagmas, los divide en sub-sintagmas para obtener los que aparecen en el corpus en situaciones no ambiguas.

TERMINO (David & Plante, 1990) está formado por tres fases secuenciales: filtrado de texto, análisis de sintagmas nominales basados en gramática, y construcción del conjunto de términos. El analizador morfológico utilizado se basa en reglas gramaticales en vez de diccionarios y se centra en el análisis de los sintagmas nominales. A diferencia de *ACABIT* y *LEXTER*, *TERMINO* posee su propio analizador morfológico (POS-Tagger) y permite ambigüedades gramaticales en el análisis morfológico.

TERMS (Justeson & Katz, 1995) se basa en la idea que los términos suelen repetirse en documentos técnicos con más frecuencia que sintagmas nominales no terminológicos y que los sintagmas nominales que representan términos tienen una estructura diferente que los sintagmas nominales que no los representan. Normalmente los sintagmas que representan términos no suelen incluir modificadores ni determinantes, al contrario que los sintagmas nominales comunes. Así, *TERMS* utiliza un filtro estadístico que rechaza las cadenas que aparecen sólo una vez en el documento.

3.6.2 Descripción de la fase de búsqueda de conceptos.

El objetivo principal de esta fase es encontrar expresiones lingüísticas que representen conceptos. La asociación entre expresiones lingüísticas y los conceptos que representan son almacenadas en una base de conocimiento que llamamos *base de conocimiento de conceptos*.

El proceso de búsqueda es bastante simple y la salida de este proceso es una lista que contiene las expresiones del fragmento de texto actual que aparecen en la base de conocimiento. Este proceso se basa en la idea que incluye el sistema *TERMS*, en el que un término o concepto aparece dentro de un mismo dominio utilizando expresiones lingüísticas similares.

Este proceso asume que existen palabras que no contienen significado léxico. Estas palabras suelen tener las siguientes categorías gramaticales: Preposición, Conjunción, Interjección, Partícula, Pronombre y Determinante. Es por eso por lo que el sistema sólo buscará sintagmas nominales que comiencen con Nombres, Adjetivos y Adverbios. Los verbos son palabras con significado léxico, pero la finalidad de esta fase es buscar sintagmas nominales que representen conceptos dentro de una ontología.

El sistema funciona de la siguiente manera. Primero, obtiene cada palabra que esté etiquetada con la categoría gramatical de nombre, adjetivo y adverbio, en la frase actual, y busca si existen expresiones lingüísticas ***similares*** en la base de conocimiento de conceptos. Entonces, para cada expresión similar a la palabra actual, se considera si es una expresión **aceptable** y, si es así, se añade esta expresión a la lista de posibles conceptos candidatos en la frase actual, con el conocimiento que representa. En el caso que el experto no encuentre ninguna opción válida inferida por el sistema, se le permite la posibilidad de definir los conceptos de la frase actual.

Dos expresiones lingüísticas distintas pueden representar el mismo concepto. Teniendo esto en cuenta, se incluye dentro de la base de conocimiento esta asociación, de modo que, después, se pueda identificar el concepto asociado a esta expresión. Por ejemplo, las expresiones "perro", "chucho", "can" representan al mismo concepto "can".

La función ***similar*** se encarga de identificar qué expresiones dentro de la base de conocimiento son "similares" a la palabra actual en un fragmento. En el caso más simple, se podría aplicar una función de "igualdad". Sin embargo, esta función no puede tratar expresiones compuestas por más de una palabra. Además, es más recomendable utilizar una función del tipo "esPrefijo". La función "esPrefijo" chequea si la palabra actual es un substring de otra expresión lingüística o no.

Sería conveniente que esta función pudiera tratar con familias de palabras (búsquedas de lema/lexema) y otras peculiaridades del lenguaje. Por ejemplo, si la expresión "causa" existe ya en la base de conocimiento y el fragmento actual contiene la palabra "causado", sería deseable que el sistema se diese cuenta que ambas palabras representan el mismo verbo (lema). Este tratamiento está parcialmente implementado usando part-of-speech taggers y analizadores morfológicos. En este trabajo, una expresión lingüística es "similar" a la expresión actual, si empieza con ésta.

La función ***aceptable*** es una extensión de la ***similar***. Como la función *similar* puede ser muy permisiva, la función *aceptable* se utiliza para determinar si la expresión lingüística actual y la expresión similar no son similares por casualidad.

La función "esPrefijo" tiene una desventaja importante: si la palabra actual es, por ejemplo, el sustantivo "cáncer", cualquier expresión que empiece con "cáncer", como "cáncer de pecho", "cáncer de mama" o "cancerígeno" son considerados como similares.

Además, esta función *aceptable* limita el número de palabras similares. Esta función se ha diseñado con requerimientos fuertes: una expresión lingüística es *aceptable* si aparece en el fragmento actual.

Las palabras actuales en un fragmento de texto son siempre simples. Sin embargo, las expresiones contenidas en la base de conocimiento pueden contener mas de una palabra (expresiones compuestas). Si una palabra es aceptable, entonces el fragmento actual ha de contener todas las palabras que aparecen en la expresión de la base de conocimiento.

Si hay varias asociaciones de conocimiento posibles en la base de datos, el sistema las ordena y las muestra. La posible existencia de más de una asociación se debe a las siguientes razones:

- Dependencia del dominio: los diferentes significados dados a un término pueden variar con respecto al dominio en el que se utiliza. Por ejemplo la palabra "banco"

tiene varios significados según el dominio en que se encuentre y no representan, por tanto, el mismo concepto.

- Dependencia de la persona: los expertos pueden asignar diferentes significados a la misma expresión.

- Localización espacial: si una expresión ha sido usada recientemente, en términos de la distancia en palabras hasta la expresión actual, con un significado específico y aparece la misma expresión, es muy probable que las dos expresiones lingüísticas tengan el mismo significado.

Cuando existen diferentes posibilidades para una expresión dada, el sistema las ordena de acuerdo con los tres factores anteriores. Entre esos factores, la localización espacial interactúa de dos maneras diferentes. El sistema considera si una expresión se ha utilizado ya en el mismo texto y/o en el fragmento actual de texto (en este caso, se le da la mayor prioridad).

Los diferentes criterios de ordenación se caracterizan por tres parametros: (1) quién es el que reconoce el conocimiento, (2) el tipo del dominio donde se encuentra y (3) si la expresión ha aparecido ya en el texto o no. En particular, hay cinco criterios diferentes de ordenación, que se muestran en la tabla 3-3.

Una vez la ordenación ha concluido, la fase de búsqueda de conceptos termina. En este punto, el sistema es capaz de obtener los conceptos que, anteriormente, se han introducido en la base de conocimiento de conceptos. Además, el conocimiento inferido se ordena con los criterios nombrados anteriormente para evitar la ambigüedad.

Criterio de ordenación	Experto	Domino	Texto	Fragmento
1	Mismo	Mismo	Mismo	Mismo
2	Mismo	Mismo	Mismo	Distinto
3	Mismo	Mismo	Distinto	Distinto
4	Distinto	Mismo	Distinto	Distinto
5	Mismo	Distinto	Distinto	Distinto

Tabla 3-3. Criterios de ordenación posibles

3.7 FASE DE INFERENCIA (BASADA EN MCRDR)

Como se ha mencionado antes, en lenguaje natural, las relaciones entre entidades de conocimiento, y más concretamente entre conceptos, suelen estar asociadas a verbos. Es por eso por lo que esta fase se centra principalmente en la obtención de relaciones entre conceptos. Sin embargo, el sistema puede inferir también otras categorías de conocimiento como conceptos que no se hayan inferido en la fase anterior, atributos y valores.

Esta fase utiliza una base de conocimiento que contiene todas las expresiones lingüísticas que representan relaciones y un subsistema MCRDR encargado de inferir los participantes de las relaciones.

El sistema obtiene la palabra o palabras etiquetadas como verbo en la frase actual y busca expresiones lingüísticas dentro de la base de conocimiento de relaciones para obtener el tipo de relación asociada a esta expresión (una expresión verbal en la frase actual). Esta búsqueda en la *base de conocimiento de relaciones* utiliza los criterios *similar* y *aceptable* que se han definido

en la fase anterior. Obviamente, puede haber casos en los que no se encuentre ninguna buena opción para determinar una expresión lingüística que represente una relación. En este caso, el usuario debe poder definir nuevo conocimiento asociado a la expresión.

Una vez el sistema ha encontrado la relación asociada a la expresión verbal principal en la frase actual, el sistema utiliza el subsistema MCRDR para adquirir conocimiento por medio de las categorías de las palabras y su posición en la frase actual con respecto a la expresión verbal, y la relación asociada a un verbo, si existe.

El sistema crea, entonces, un caso formado por la relación, que representa la expresión verbal, y la categoría de las otras palabras de la frase. Por ejemplo, en la frase siguiente "El brazo es una parte del cuerpo", si el sistema identifica que la expresión verbal "es una parte del" representa a una relación PART OF (Mereológica), buscando en la base de conocimiento de relaciones, se crea, entonces, un caso, como se muestra en la tabla 3-4, en el que aparece la relación y las categorías de las palabras en la frase. Categoría_i representa a la categoría de la palabra que está en la posición *i* con respecto a la expresión verbal. Así, "El" está en la posición -2, "brazo" en la -1 y "cuerpo" en la posición "+1".

Frase:	El brazo **es una parte del** cuerpo
Expresión verbal:	es una parte del
Caso generado:	relación: PART OF categoría_{-1}: Nombre categoría_{-2}: Determinante categoría_{+1}: Nombre

Tabla 3-4. Un ejemplo de un caso generado por el subsistema MCRDR

Este caso se introducirá en el sistema y dará lugar a una conclusión. Si esta conclusión es correcta, el sistema de reglas en que se basa MCRDR no se verá modificado. Si no, se obtendrán las diferencias del caso actual con el anterior y el sistema elige las diferencias para crear una nueva regla dentro del sistema.

Por ejemplo, si suponemos que el sistema MCRDR está vacío, con el caso mostrado en la tabla 3-4 no obtendrá ninguna conclusión, al no existir posibles reglas que se activen. Supongamos que el experto identifica qué brazo y cuerpo son conceptos y que existe una relación **PART OF** entre ellos: brazo **PART OF** cuerpo.

El sistema, entonces, creará una nueva conclusión a incluir en la regla que se va a crear y que indica que, si nos encontramos con una estructura sintáctica similar, infiere el conocimiento. En la tabla 3-5, la función $palabra_i$ devuelve la palabra en la posición i, relativa a la expresión verbal. Se muestra la conclusión en la que, si se encuentra un caso similar al dado, se infieren dos nuevos conceptos dados por las palabras situadas en las posiciones -1 y +1 y una relación **PART OF** entre estos conceptos.

Además, el sistema obtendrá la lista de diferencias, que estará formada por todos los atributos del caso, como mostramos en la tabla 3-5. De este conjunto de diferencias, formadas por las categorías de palabras que forman la frase y la relación, sólo se toman las categorías de las palabras que contienen el conocimiento adquirido. En el ejemplo actual, la palabra "El" no forma parte del conocimiento adquirido, así que no debería incluirse para la generación de reglas.

Una vez seleccionadas las diferencias entre casos, se actualizará el árbol de reglas del subsistema MCRDR, dando lugar a una nueva regla insertada en el sistema:

if (relacion=PART OF & categoría$_{-1}$=Nombre & categoría$_{+1}$= Nombre) **then**

concepto$_1$=new Concepto(palabra$_{-1}$())

concepto$_2$=new Concepto(palabra$_{+1}$())

relacion$_1$=new Relacion(PART OF, concepto$_1$, concepto$_2$);

Frase:	El brazo **es una parte del** cuerpo
Diferencias:	**relación**: PART OF **categoría**$_{-1}$: Nombre categoría$_{-2}$: Determinante **categoría**$_{+1}$: Nombre
Conclusión	**concepto**$_1$: **new Concepto**(palabra$_{-1}$()) **concepto**$_2$: **new Concepto**(palabra$_{+1}$()) **New Relacion(PART OF, concepto**$_1$**, concepto**$_2$**)**

Tabla 3-5. Diferencias y conclusión que obtiene el sistema.

Ahora el sistema sería capaz de inferir expresiones del tipo siguiente:

Nombre *expresión verbal asociada a una relación PART OF* Nombre

En el capítulo siguiente se muestran ejemplos más extensos del modo de funcionamiento del sistema de adquisición de conocimiento.

3.8 RESUMEN

En este capítulo se ha descrito un entorno de ontology learning basado en MCRDR, que es una metodología utilizada para construir sistemas basados en conocimiento y sistemas expertos.

El proceso de adquisición de conocimiento mediante dicho entorno se compone de tres fases secuenciales:

- POS-Tagging: En esta fase se obtiene la categoría gramatical de cada palabra en la frase actual. Para ello, utiliza el tagger descrito en el capítulo 2 de este trabajo, donde se presenta un POS-Tagger basado en árboles C4.5.

- Fase de búsqueda de conceptos: Con la realización de esta fase se identifican las expresiones lingüísticas que representan conceptos del dominio. La asociación entre las expresiones lingüísticas y los conceptos asociados se almacenan en la base de conocimiento de conceptos.

- Fase de inferencia: El conjunto de actividades comprendidas en esta fase se basa en la hipótesis de que en lenguaje natural, las relaciones entre conceptos (nivel semántico) suelen estar expresadas mediante verbos (nivel simbólico). Aunque esta fase sirve principalmente para obtener relaciones entre conceptos, puede utilizarse también para adquirir otras entidades de conocimiento como conceptos, atributos y valores. En esta fase se utiliza una base de conocimiento que contiene las expresiones lingüísticas que representan las relaciones genéricas entre conceptos (descritas en este trabajo), y por un subsistema de inferencia basado en reglas que obtiene los participantes en esas relaciones.

CAPÍTULO 4

EJEMPLOS DE LA METODOLOGÍA DE ONTOLOGY LEARNING

4.1 INTRODUCCIÓN

En este capítulo, se muestra cómo funciona el sistema a través de una serie de ejemplos, en los que unas veces se parte de bases de conocimiento y bases de reglas MCRDR vacías; y otras, sólo están vacías la base de conocimiento de relaciones y el subsistema de reglas, que utiliza el módulo de inferencia basado en MCRDR.

4.2 EJEMPLO 1

A continuación, se describe un ejemplo simple del dominio de componentes de ordenador y que muestra el funcionamiento del sistema propuesto en este trabajo. Inicialmente, se supone que las bases de conocimiento están vacías. Para empezar asumiremos que el sistema obtiene la siguiente frase: “Una impresora es un periférico”.

POS-Tagging

El sistema primero analiza la frase con el POS-Tagger para obtener las categorías gramaticales de cada palabra dentro de la frase. El resultado del analizador es el siguiente:

Una (Determinante) impresora (Nombre) es (Verbo) un (Determinante) periférico (Nombre).

Fase de búsqueda de conceptos

Como la base de conocimiento de conceptos está vacía, el sistema no infiere ningún concepto.

Fase de inferencia (basada en MCRDR)

Como la base de conocimiento de relaciones y el subsistema MCRDR están inicialmente vacíos, el sistema no infiere ningún conocimiento.

Como no se ha obtenido ningún conocimiento de la frase, el sistema permite al experto que identifique el conocimiento existente en la frase. Supongamos que el experto identifica que la expresión "es un" está asociada a una relación "IS-A" (Taxonómica) entre los conceptos "impresora" y "periférico".

A continuación, el sistema almacenará que la expresión "impresora" está asociada al concepto "impresora" y que la expresión periférico está asociada al concepto "periférico" en la base de conocimiento de conceptos, para que la próxima vez que el sistema se encuentre con estas expresiones, o unas parecidas, pueda identificar que representan estos conceptos. En la tabla 4-1 se muestra cómo queda la base de conocimiento de conceptos.

Expresiones lingüísticas	Concepto asociado
Impresora	Impresora
Periférico	Periférico

Tabla 4-1. Base de conocimiento de conceptos

Además, almacenará en la base de conocimiento de relaciones que la expresión "es un" representa una relación taxonómica (tabla 4-2), y almacenará un nuevo caso en el sub-sistema MCRDR, que creará la siguiente regla:

If relation=”IS-A” and category (pos(verb)-1)=Noun and category(pos(verb)+1)=Noun **then** {

Concept(word(pos(verb)-1))

Concept(word(pos(verb)+1))

Relation(IS-A, word(pos(verb)-1),word(pos(verb)+1)

}

Aquí “*pos(verb)*” es una función que devuelve la posición de la expresión lingüística que representa la relación en la frase actual. Esta función se utiliza para obtener las posiciones relativas de las palabras con respecto a la posición de la expresión lingüística. La función “*category(i)*” devuelve la categoría de la palabra que se encuentra en la posición i y la función “*word(i)*” devuelve la palabra de dicha posición.

Expresiones lingüísticas	Relación asociada
es un	IS A

Tabla 4-2. Base de conocimiento de relaciones

Supongamos que ahora el sistema obtiene la frase siguiente: “La Cannon-1000 es un tipo de impresora”.

POS-Tagging

La(Determinante) Cannon-1000 (Nombre) es(Verbo) un(Determinante) tipo(Nombre) de(Preposición) impresora(Nombre)”.

Fase de búsqueda de conceptos

Como la base de conocimiento de conceptos ya no está vacía, el sistema buscará si dentro de la frase hay conceptos que pertenezcan a esta base de conocimiento (Tabla 4-1). El sistema entonces inferirá que la expresión "impresora" representa al concepto *impresora*.

Fase de inferencia (basada en MCRDR)

Dada esta situación, en esta fase el sistema buscará por expresiones que representen alguna relación. Así, el sistema busca las expresiones lingüísticas **similares** al verbo principal de la frase, "es", dentro de la base de conocimiento de relaciones (Tabla 4-2). Entonces el sistema obtiene que la expresión "es un", que representa una relación IS A, es similar a la expresión "es". Ahora el sistema obtiene alguna expresión **aceptable** entre todas las similares (si la hay). En este caso podemos ver que la expresión "es un" es una expresión **aceptable** para la frase actual.

Luego el sistema infiere que la expresión "es un" representa una relación IS-A, y en el sistema de reglas se activa la regla introducida anteriormente, infiriendo "Cannon-1000 ***IS-A*** Tipo".

Como podemos ver, el conocimiento inferido no es correcto. Supongamos entonces que el experto realiza la corrección: "Canon-1000 ***IS-A*** impresora", identificando que la expresión "es un tipo de" representa la relación "IS A".

Entonces el sistema almacenará las expresiones que representan los conceptos en esta nueva frase en la base de conocimiento de conceptos. La base de conocimiento de conceptos quedaría según muestra la tabla 4-3.

Expresiones lingüísticas	Concepto asociado
Canon-1000	Canon-1000
Impresora	Impresora
Periférico	Periférico

Tabla 4-3. Base de conocimiento de conceptos

Además almacenará en la base de conocimiento de relaciones que la expresión "es un tipo de" representa una relación taxonómica (tabla 4-4).

Expresiones lingüísticas	Relación asociada
es un es un tipo de	IS A IS A

Tabla 4-4. Base de conocimiento de relaciones

Ahora el sistema obtiene el caso generado para el sistema de inferencia y obtiene las diferencias entre los dos casos. Como se puede ver, los dos casos son conceptualmente el mismo caso. Entonces, el sistema no actualizará las reglas en el sub-sistema MCRDR, por lo que el subsistema MCRDR seguirá teniendo la misma regla.

En la figura 4-1 se muestra la ontología obtenida en el procesamiento de estas frases simples.

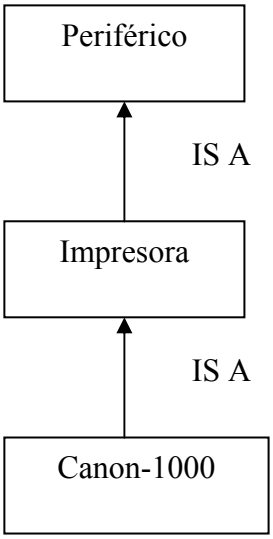

Figura 4-1. Ontología parcial obtenida por el procesamiento del ejemplo 1

El sistema podrá inferir correctamente el conocimiento contenido en expresiones como "Nombre" es un "Nombre" y "Nombre" es un tipo de "Nombre".

Obsérvese que si el subsistema MCRDR contiene bastantes casos y la base de conocimiento de relaciones contiene bastantes entradas, el sistema será capaz de inferir muchas relaciones y otras entidades de conocimiento con sólo conocer su categoría gramatical.

4.3 EJEMPLO 2

El ejemplo que se muestra a continuación presenta frases obtenidas de un corpus relacionado con enfermedades de corazón. Este ejemplo se presenta en inglés para que se pueda observar que el funcionamiento del sistema es independiente del idioma, como se ha señalado anteriormente.

En este ejemplo, suponemos que la base de conocimiento de conceptos no está vacía inicialmente y está formada por cuatro expresiones lingüísticas que se muestran en la tabla 4-5. Además, se parte de que la base de conocimiento de relaciones y el sistema MCRDR están vacíos.

Expresiones lingüísticas	Concepto asociado
Cardiac disease	Cardiac disease
Cardiac glycosides	Cardiac glycoside
Treatment	Treatment
Treatment of cardiac disease	Treatment of cardiac disease

Tabla 4-5. Base de conocimiento de conceptos

Supongamos que el sistema obtiene la siguiente frase: “Cardiac glycosides have been used in the treatment of cardiac disease for more than 200 years”

POS-Tagging

"Cardiac [Adjective] glycosides [Noun] have [Verb auxiliary] been [Verb auxiliary] used [Verb lexical] in [Preposition] the [Determiner] treatment [Noun] of [Preposition] cardiac [Adjective] disease [Noun] for [Preposition] more [Adverb] than [Conjunction] 200 [Numeral] years [Noun]"

Fase de búsqueda de conceptos

Debido a la existencia de palabras sin conocimiento léxico, que suelen tener las categorías gramaticales de Preposición, conjunción, interjección, partícula, pronombre y determinante, el sistema sólo buscará los conceptos asociados a nombres, adjetivos y adverbios. Con esto, el sistema de búsqueda de conceptos será más eficiente, centrándose sólo en aquellas palabras que pueden ser candidatas a ser conceptos.

Es por eso por lo que el sistema obtiene cada palabra de la frase actual con categoría gramatical de nombre, adjetivo o adverbio y busca por palabras **similares** en la base de conocimiento de conceptos. Entonces, para cada una de estas expresiones similares, se mira si se considera una expresión **aceptable**, como se ha indicado en el capítulo anterior.

Así, en el ejemplo anterior, la primera palabra en la frase es "Cardiac", que tiene asociada la categoría gramatical de adjetivo, por lo que el sistema buscará las expresiones lingüísticas **similares** en la base de conocimiento de conceptos. El resultado de las palabras similares es el siguiente:

{cardiac disease, cardiac glycoside}.

Entonces, el sistema ejecutará la función **aceptable**, para encontrar si alguna de las expresiones **similares** obtenidas arriba cumple con el criterio de aceptabilidad. De esas dos

expresiones, se puede ver que "Cardiac glycoside" es una expresión **aceptable** en esta frase, así que se marca como un concepto.

La siguiente palabra a procesar es "treatment", porque todas las palabras anteriores no tienen una categoría gramatical de nombre, adjetivo ni adverbio.

El sistema busca en la base de conocimiento de conceptos, obteniendo los siguientes resultados:

Similares: {treatment, treatment of the cardiac disease}

Aceptable: {treatment of the cardiac disease}

La siguiente palabra es "years", que no tiene ninguna palabra **similar** dentro de la base de conocimiento de conceptos.

El sistema encuentra entonces los dos conceptos siguientes:

"cardiac glycosides" y "treatment of the cardiac disease".

Fase de inferencia (MCRDR)

Como la base de conocimiento de relaciones y el subsistema MCRDR están inicialmente vacíos, el sistema no infiere ningún conocimiento (en esta fase).

Supongamos que el experto identifica que la expresión "has been used in the" está asociada a una relación "IS-USED-BY" entre los conceptos "Cardiac glycosides" y "treatment of cardiac disease".

Como las expresiones lingüísticas que representan los conceptos extraídos ya se encuentran en la base de conocimiento del sistema, éste no deberá almacenarlas.

Lo que si se almacenará en la base de conocimiento de relaciones es que la expresión "has been used in the" representa una relación "IS-USED-BY" (tabla 4-6).

Expresiones lingüísticas	Relación asociada
Has been used in the	IS USED BY

Tabla 4-6. Base de conocimiento de relaciones

Además, se almacenará un nuevo caso en el sub-sistema MCRDR, que creará la siguiente regla:

If relation="IS-USED-BY" and category (pos(verb)-1)=Noun and category(pos(verb-2)=Adjective and category(pos(verb)+1) and category(pos(verb)+2)=prep and (category(pos(verb)+3)=adjective) and (category(pos(verb)+4)=Noun) **then** {

concept1=**Concept**(word(pos(verb)-2) + word(pos(verb)-1))

concept2=**Concept**(word(pos(verb)+1) + word(pos(verb)+2) + word(pos(verb)+3) + word(pos(verb)+4))

Relation(IS-USED-BY, concept1, concept2)

}

Aquí, el operador + concatena dos expresiones lingüísticas.

Supongamos que ahora el sistema se encuentra con la frase siguiente: "Ultrasonography is a method that sends high frequency sound waves into the breast".

POS-Tagging

Ultrasonography[Noun] is[Verb auxiliary] a[Preposition] method[Noun] that[Pronoun] sends[Verb lexical] high[Adjective] frequency[Noun] sound[Noun] waves[Noun] into[Preposition] the[Determiner] breast[Noun].

Fase de búsqueda de conceptos

Como la base de conocimiento de conceptos ya no está vacía, el sistema buscará si dentro de la frase hay conceptos que pertenezcan a esta base de conocimiento (Tabla 4-5). El sistema entonces no inferirá ningún concepto por no hallar ninguna expresión **similar** en la frase actual.

Fase de inferencia (MCRDR)

Debido a que no existe ninguna expresión que represente una relación incluida en la base de conocimiento de relaciones, el sistema no lanzará ninguna regla y no inferirá ningún conocimiento.

Supongamos que el experto identifica las siguientes entidades de conocimiento en la frase actual.

Conceptos: Ultrasonography, method, breast

Relaciones: Ultrasonography ***IS A*** method

El sistema almacenará en la base de conocimiento de conceptos las tres nuevas expresiones, quedando ésta como se indica en la tabla 4-7.

Expresiones lingüísticas	Concepto asociado
Breast	Breast
Cardiac disease	Cardiac disease
Cardiac glycosides	Cardiac glycoside
Method	Method
Treatment	Treatment
Treatment of cardiac disease	Treatment of cardiac disease
Ultrasonography	Ultrasonography

Tabla 4-7. Base de conocimiento de conceptos

Además, se introducirá en la base de conocimiento de relaciones una nueva entrada, como se puede ver en la tabla 4-8.

Expresiones lingüísticas	Relación asociada
Has been used in the	IS USED BY
Is a	IS A

Tabla 4-8. Base de conocimiento de relaciones

Y el subsistema MCRDR añadirá un nuevo caso, creando la regla siguiente

If relation=”IS-A” and category (pos(verb)-1)=Noun and category(pos(verb)+1)=Noun **then** {

Concept(word(pos(verb)-1)),

Concept(word(pos(verb)+1)),

Relation(IS-A, word(pos(verb)-1),word(pos(verb)+1)

}

El sistema continuará analizando el texto. Supongamos que el sistema obtiene ahora la siguiente frase “Breast cancer is a disease in which cancer cells are found in the tissues of the breast”

POS-Tagging

Breast[Noun] cancer[Noun] is[Verb auxiliary] a[Preposition] disease[Noun] in[Preposition] which[Pronoun] cancer[Noun] cells[Noun] are[Verb auxiliary] found[Verb lexical] in[Preposition] the[Determiner] tissues[Noun] of[Preposition] the[Determiner] breast[Noun]

Fase de búsqueda de conceptos

Como la base de conocimiento de conceptos ya no está vacía, el sistema buscará si dentro de la frase hay conceptos que pertenezcan a esta base de conocimiento (Tabla 4-8). El sistema entonces encontrará el concepto “Breast”.

Fase de inferencia (MCRDR)

Dada esta situación, el sistema buscará en la base de conocimiento de relaciones una expresión similar al verbo "is". Hasta ese instante, sólo hay una expresión similar a la expresión "is", que es "is a" (tabla 4-7). Ahora, el sistema ha de obtener las expresiones **aceptables** de las **similares**. Podemos ver que "is a" es una expresión aceptable en la frase actual, por lo que el sistema infiere que la expresión 'is a' representa una relación "IS-A", y se activa una regla infiriendo la siguiente relación:

'cancer IS-A disease'

Supongamos que ahora el experto se da cuenta que el conocimiento extraído es incorrecto y lo corrige a la siguiente relación

'Breast cancer IS-A disease'

Ante esta situación, el sistema actualizará su base de conocimiento de conceptos (tabla 4-9), mientras la base de conocimientos de relaciones no se verá alterada. Sin embargo, el subsistema MCRDR obtendrá el caso generado y las diferencias entre estos dos casos. Como se puede observar, sólo hay una diferencia consistente en que el primer concepto está formado por un Adjetivo y un Nombre, no sólo por un nombre, por lo que el sistema actualizará las reglas en el subsistema MCRDR, insertando la siguiente regla:

If relation="IS-A" and category (pos(verb)-1)=Noun and category(pos(verb)-2)=Noun and category(pos(verb)+1)=Noun **then** {

concept1=**Concept**(word(pos(verb)-2) + word(pos(verb)-1))

concept2=**Concept**(word(pos(verb)+1))

Relation(IS-A, concept1, concept2)

}

Expresiones lingüísticas	**Concepto asociado**
Breast	Breast
Cardiac disease	Cardiac disease
Cardiac glycosides	Cardiac glycoside
Method	Method
Treatment	Treatment
Treatment of cardiac disease	Treatment of cardiac disease
Ultrasonography	Ultrasonography

Tabla 4-9. Base de conocimiento de conceptos

Después de procesar las tres frases anteriores podemos ver los componentes ontológicos obtenidos en la figura 4-2.

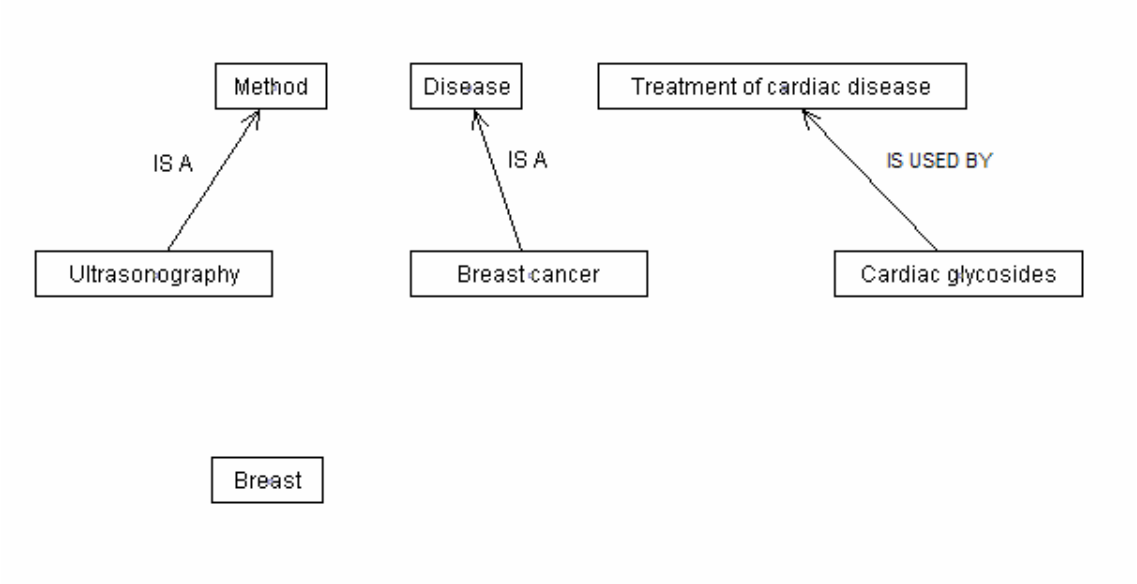

Figura 4-2. Componentes ontológicos obtenidos por el procesamiento del ejemplo 2

4.4 EJEMPLO 3.

Supongamos que las bases de conocimiento de relaciones y el sistema de inferencia están inicialmente vacíos. Supongamos también que la base de conocimiento de conceptos está formada por los conceptos mostrados en la tabla 4-10 y que el sistema obtiene la siguiente frase "El cáncer de piel es una enfermedad producida por el desarrollo de células cancerosas en las capas exteriores de la piel."

Expresiones Lingüísticas	Concepto asociado
Cáncer	Cáncer
Cáncer de mama	Cáncer de mama
Cáncer de piel	Cáncer de piel
Enfermedad	Enfermedad

Tabla 4-10. Base de conocimiento de conceptos

POS-Tagging

El [det] cáncer [noun] de [prep] piel [noun] es [verb (auxiliary)] una [det] enfermedad [noun] producida [Verb (lexical)] por [prep] el [det] desarrollo [noun] de [prep] células [noun] cancerosas [adj] en [prep] las [det] capas [noun] exteriores [adj] de [prep] la [det] piel [noun]

Fase de búsqueda de conceptos

Como el sistema sólo busca conceptos asociados a nombres, adjetivos, y adverbios, la primera palabra a procesar en la sentencia será "cáncer", que tiene asociada la categoría gramatical de nombre. El sistema busca por expresiones similares en la base de conocimiento de conceptos. El resultado de las palabras similares es:

Similar: {cáncer, cáncer de mama, cancer de piel}

Luego la función aceptable obtendrá que la única expresión lingüísitca que cumple los criterios es "Cáncer de piel", así que el sistema obtiene que la expresión "Cáncer de piel" representa al concepto "Cáncer de piel".

La siguiente palabra para procesar es "enfermedad", porque todas las palabras anteriores no están etiquetadas como nombre, adjetivo, ni adverbio.

Similar:{enfermedad}

Aceptable: {enfermedad}

El sistema procederá análogamente con el resto de palabras en la frase, obteniendo los siguientes conceptos:

Conceptos: {cáncer de piel, enfermedad}

Fase de inferencia

Como la base de conocimiento de relaciones y el subsistema MCRDR están todavía vacíos, no se infiere conocimiento alguno en esta fase.

Supongamos que el experto identifica las siguientes entidades de conocimiento en la frase actual.

Conceptos: {cáncer de piel, enfermedad} (ya inferidos por el sistema)

Relaciones: Cáncer de piel ***IS A*** enfermedad

Como los conceptos identificados ya se encuentran en la base de conocimiento de conceptos, ésta no sufrirá ninguna actualización.

Como la base de conocimiento de relaciones está vacía, se añade una nueva entrada, quedando como se muestra en la tabla 4-11.

Expresiones lingüísticas	Relación asociada
es una	IS A

Tabla 4-11. Base de conocimiento de relaciones

Y el subsistema MCRDR añadirá un nuevo caso creando la regla siguiente

R1: If relation="IS-A" and category (pos(verb)-1)=Noun and category (pos(verb)-2)=prep and category(pos(verb)-3)=Noun and category(pos(verb)+2)=Noun **then** {

$concept_1$=**Concept**(word(pos(verb)-3)+ word(pos(verb)-2)+ word(pos(verb)-1)),

$concept_2$=**Concept**(word(pos(verb) +1)),

Relation(IS-A, $concept_1$, $concept_2$)

}

Supongamos que el sistema se encuentra ahora con la siguiente frase: ”El cáncer de próstata se trata de un tumor maligno que se desarrolla en la glándula prostática”

POS-Tagging

“El [det] cáncer[noun] de[prep] próstata[noun] se[Pro] trata[Verb(lexical)] de[prep] un[det] tumor[noun] maligno[adj] que[pro] se[Pro] desarrolla[Verb(lexical)] en[prep] la[adj] glándula[noun] prostática[adj]”

Fase de búsqueda de conceptos

El sistema busca si dentro de la frase hay conceptos que pertenezcan a esta base de conocimiento (Tabla 4-10). El sistema entonces encuentra los conceptos siguientes:

Conceptos: {cáncer}

Fase de inferencia

El sistema busca en la base de conocimiento de relaciones expresiones similares al verbo “trata”, pero no encuentra ninguna expresión similar y, por lo tanto, no infiere ningún conocimiento.

Supongamos que el experto identifica que la expresión “se trata de un” está asociada a una relación taxonómica IS-A. Después de esto, se activa la regla **R1** del sub-sistema MCRDR obteniendo las siguientes entidades de conocimiento en la frase actual.

Conceptos: {cáncer de próstata, tumor}

Relaciones: Cáncer de próstata ***IS A*** tumor

El experto aceptará el conocimiento inferido y el sistema introducirá los conceptos inferidos en la base de conocimiento de conceptos para que se puedan identificar en un futuro. Esta base de conocimiento quedaría como se muestra en la tabla 4-12.

Expresiones Lingüísticas	Concepto asociado
Cáncer	Cáncer
Cáncer de mama	Cáncer de mama
Cáncer de piel	Cáncer de piel
Cáncer de próstata	Cáncer de próstata
Enfermedad	Enfermedad
Tumor	Tumor

Tabla 4-12. Base de conocimiento de conceptos

Además, se añade una nueva entrada en la base de conocimiento de relaciones, quedando como muestra la tabla 4-13.

Expresiones lingüísticas	Relación asociada
es una	IS A
se trata de un	IS A

Tabla 4-13. Base de conocimiento de relaciones

La siguiente frase a analizar es "Una exposición excesiva al sol puede producir cáncer de piel sin el uso de cremas protectoras solares"

POS-Tagging

"Una [det] exposición[noun] excesiva[adj] al[det] sol[noun] puede[Verb(auxiliary)] producir[Verb(lexical)] cáncer[noun] de[prep] piel[noun] sin[adv] el[det] uso[noun] de[prep] cremas[noun] protectoras[adj] solares[adj]"

Fase de búsqueda de conceptos

El sistema busca si dentro de la frase hay conceptos que pertenezcan a esta base de conocimiento (Tabla 4-12). El sistema entonces encuentra los conceptos siguientes:

Conceptos: {cáncer de piel}

Fase de inferencia

El sistema busca en la base de conocimiento de relaciones expresiones similares al verbo "puede", pero no encuentra ninguna expresión similar y, por lo tanto, no infiere ningún conocimiento.

El experto asocia a la expresión "puede producir" una relación CAUSE e identifica el siguiente conocimiento en la frase actual:

Conceptos: {sol, cáncer de piel}

Relaciones: Sol ***CAUSE*** Cáncer de piel

Se actualizan entonces las bases de conocimiento de conceptos y de relaciones quedando como muestran las tablas 4-14 y 4-15.

Expresiones Lingüísticas	Concepto asociado
Cáncer	Cáncer
Cáncer de mama	Cáncer de mama
Cáncer de piel	Cáncer de piel
Cáncer de próstata	Cáncer de próstata
Enfermedad	Enfermedad
Sol	Sol
Tumor	Tumor

Tabla 4-14. Base de conocimiento de conceptos

Expresiones lingüísticas	Relación asociada
Es una	IS A
Puede producir	CAUSE
se trata de un	IS A

Tabla 4-15. Base de conocimiento de relaciones

Además, se crea la siguiente regla en el sub-sistema MCRDR:

R2: If relation="CAUSE" and category (pos(verb)-1)=Noun and category(pos(verb)+1)=noun and category(pos(verb)+2)=prep and category(pos(verb)+3)=noun **then** {

$concept_1$=**Concept**(word(pos(verb)-1))

$concept_2$=**Concept**(word(pos(verb)+1)+word(pos(verb)+2)+ word(pos(verb)+3))

Relation(CAUSE, $concept_1$, $concept_2$)

}

La siguiente frase a procesar es: "Actualmente, el cáncer de páncreas es una clase de tumor muy difícil de controlar incluso con los tratamientos disponibles".

POS-Tagging

Actualmente[adv], el[det] cáncer[noun] de[prep] páncreas[noun] es[verb(auxiliary)] una[det] clase[noun] de[prep] tumor[noun] muy[adv] difícil[adj] de[prep] controlar[verb(lexical)] incluso[adv] con[prep] los[det] tratamientos[noun] disponibles[adj].

Fase de búsqueda de conceptos

Conceptos encontrados: {cáncer de páncreas, tumor}

Fase de inferencia

El sistema busca expresiones similares al verbo "es" en la base de conocimiento de relaciones. Así, el sistema obtiene que la expresión "es una" es una expresión similar y aceptable y el sistema activa la regla **R1**, obteniendo el siguiente conocimiento:

Conceptos: {cáncer de páncreas, clase}

Relaciones: Cáncer de páncreas ***IS A*** clase

Como podemos ver, este conocimiento es erróneo, así que el experto, al darse cuenta del error identifica que la expresión "es una clase de" representa la relación **IS-A**. Ahora, el sistema volverá a activar la regla **R1**, infiriendo el siguiente conocimiento ya correcto.

Conceptos: {cáncer de páncreas, tumor}

Relaciones: Cáncer de páncreas ***IS A*** tumor

El sistema ahora sólo actualiza la base de conocimiento de relaciones (tabla 4-16), ya que los conceptos identificados en esta frase ya se encuentran en su respectiva base de conocimiento y el sub-sistema MCRDR no cambia, al haberse aplicado bien la regla e inferir correctamente las entidades de conocimiento.

Expresiones lingüísticas	Relación asociada
Es una	IS A
Es una clase de	IS A
Puede producir	CAUSE
se trata de un	IS A

Tabla 4-16. Base de conocimiento de relaciones

Siguiente frase: "El hábito del tabaco es la causa principal de cáncer de pulmón en el 90% de los casos de los varones y en el 70% de las mujeres.

POS-Tagging

El[det] hábito[noun] del[prep] tabaco[noun] es[verb(auxiliary)] la[det] causa[noun] principal[adj] de[prep] cáncer[noun] de[prep] pulmón[noun] en[prep] el[det] 90%[num] de[prep] los[det] casos[noun] de[prep] los[det] varones[noun] y[conj] en[prep] el[det] 70%[num] de[prep] las[det] mujeres[noun]

Fase de búsqueda de conceptos

Conceptos encontrados: {cáncer}

Fase de inferencia

En esta fase se encuentra sólo una expresión similar al verbo "es" ("es una"), pero no es una expresión aceptable, por lo que el sistema no infiere nada.

El experto identifica que la expresión "es la causa principal" está asociada a una relación **CAUSE**. Luego se activa la regla **R2**, obteniendo el siguiente conocimiento:

Conceptos: {tabaco, cáncer de pulmón}

Relaciones: tabaco ***CAUSE*** cáncer de pulmón

A continuación, el sistema se actualizará introduciendo las expresiones verbales que representan conceptos, así como la que representa la relación **CAUSE** en sus respectivas bases de conocimiento. Las tablas 4-17 y 4-18 muestran cómo quedarían las bases de conocimiento después de procesar estas frases.

Expresiones Lingüísticas	Concepto asociado
Cáncer	Cáncer
Cáncer de mama	Cáncer de mama
Cáncer de piel	Cáncer de piel
Cáncer de próstata	Cáncer de próstata
Cáncer de pulmón	Cáncer de pulmón
Enfermedad	Enfermedad
Sol	Sol
Tabaco	Tabaco
Tumor	Tumor

Tabla 4-17. Base de conocimiento de conceptos

Expresiones lingüísticas	Relación asociada
Es la causa principal	CAUSE
Es una	IS A
Es una clase de	IS A
Puede producir	CAUSE
se trata de un	IS A

Tabla 4-18. Base de conocimiento de relaciones

En la figura 4-3 podemos ver los componentes ontológicos obtenidos después de procesar el ejemplo en el sistema.

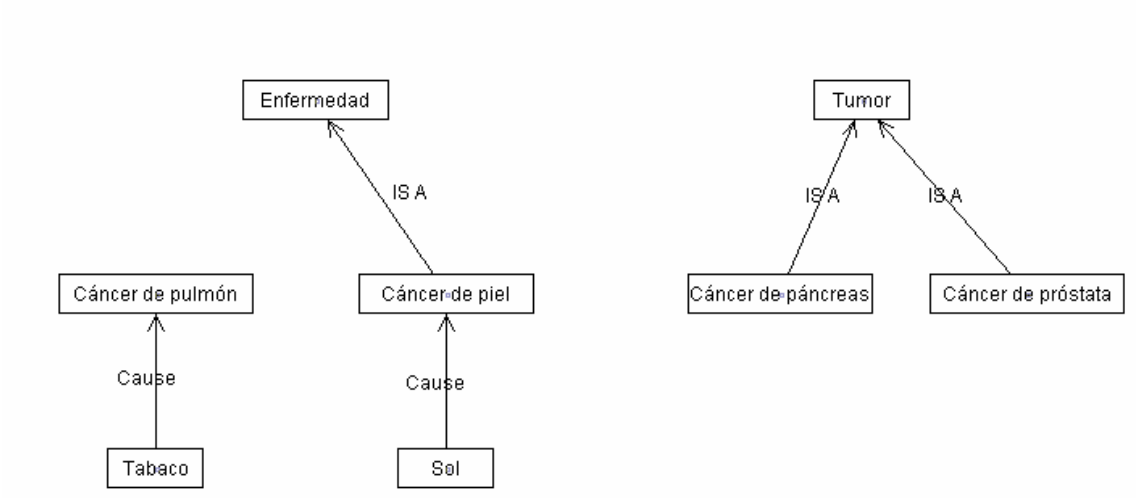

Figura 4-3. Componentes ontológicos obtenidos por el procesamiento del ejemplo 3

4.4 RESUMEN

En este capítulo se han mostrado tres ejemplos prácticos detallados que describen el modus operandi del entorno de ontology learning. A través de estos ejemplos se muestra el proceso de adquisición del conocimiento en las tres fases en que se basa (POS-Tagging, Fase de búsqueda de conceptos, y Fase de Inferencia) a través de diferentes textos de distintos dominos.

Más concretamente, el primer ejemplo es un ejemplo sencillo en el dominio de los componentes de los ordenadores. El segundo trata sobre el dominio de las enfermedades de corazón, y el idioma del texto utilizado en este ejemplo es el inglés. El último ejemplo trata el dominio de los tipos de cáncer, siendo en esta ocasión castellano el idioma del texto.

A través de este capítulo, se ha ilustrado el entorno de extracción de conocimiento a partir de texto del capítulo 3 y cómo es válido tanto para castellano como para inglés, introduciendo dos ejemplos en castellano y uno en inglés. Sin embargo, la validación de dicho entorno no ha sido abordada. En el próximo capítulo se describe una validación de aquel en diferentes dominios.

CAPÍTULO 5

APLICACIÓN DE LA METODOLOGÍA DE ONTOLOGY LEARNING

5.1 INTRODUCCIÓN

En este capítulo se pretende mostrar cómo se ha aplicado la metodología de ontology learning explicada en los capítulos anteriores en dos grandes áreas de aplicación: la Web Semántica y el reconocimiento de órdenes en lenguaje natural aplicado a la robótica médica.

5.2 APLICACIÓN EN LA WEB SEMÁNTICA.

5.2.1 Introducción

La World Wide Web se ha convertido en una fuente muy atractiva de información, debido al tamaño y a la diversidad de material textual. Más concretamente, se ha utilizado, por ejemplo, para la adquisición de información léxica y corpora de manera automática (Santamaría *et al*, 2003):

Extracción de corpora paralelos. Los corpora paralelos son textos que aparecen en dos idiomas distintos, de modo que uno es la traducción del otro. Según Resnik & Smith (2003), la Web está repleta de páginas que están disponible en dos o más idiomas y gracias a esto, es posible la construcción de corpora paralelos de dos idiomas determinados, y lo demuestran presentando en ese trabajo un sistema para la construcción de corpora paralelos en la Web.

Extracción de información léxica. En Aguirre *et al* (2000), se utilizan los motores de búsqueda y la Web para obtener conceptos dentro de los documentos Web candidatos para enriquecer la ontología WordNet, mientras que en Fuji & Ishikawa (1999) utilizan la Web como un recurso para obtener frases descriptivas o definiciones de términos técnicos.

5.2.2 Ontology learning en la Web Semántica

Dentro del aprendizaje de ontologías, una actividad muy importante es la construcción de ontologías a partir de documentos Web (Alani *et al*, 2003; Davulcu *et al*, 2003) para la consecución de una Web que mejore las prestaciones de la Web actual, a saber, la Web Semántica.

La Web Semántica es una extensión de la Web actual donde la información viene dotada de significado bien definido, y permite a las computadoras y a la gente trabajar en cooperación (Berners-Lee et. al, 2001).

El proyecto de la Web Semántica trata de crear un medio universal para el intercambio de información, dotando a los contenidos de los documentos Web de conocimiento de forma que éste sea comprensible por las máquinas.

Las ontologías se consideran necesarias para el desarrollo de la Web Semántica, puesto que pueden mejorar el funcionamiento de la web de varias formas. Las ontologías pueden ser usadas para mejorar la efectividad de búsquedas web, porque el programa de búsqueda puede buscar sólo páginas referidas a conceptos específicos. Las páginas se enlazan a páginas de ontologías que definen información sobre el dominio. Toda la información la procesa un ordenador y podría ser usada para responder a preguntas que implicaran que el usuario rastreara el contenido de varias páginas devueltas por el motor de búsqueda. El uso de ontologías para la Web Semántica también es promovido en Fensel & Musen (2001), donde se considera que las ontologías son una tecnología clave para la Web Semántica.

En Maedche & Staab (2001), los autores establecen que la definición de teorías del dominio compartidas y comunes puede ayudar a personas y máquinas a comunicar de forma concisa, facilitando el intercambio semántico. Así, la construcción rápida y barata de ontologías específicas del dominio es crucial para el éxito y la popularidad de la Web Semántica.

La necesidad de enriquecer la Web con grandes cantidades de ontologías que capturen el conocimiento del dominio ha dado lugar a diversos estudios para la construcción automática de ontologías, ya que la construcción manual requiere un enorme esfuerzo humano. Se piensa que la adquisición de ontologías es un cuello de botella para la consecución de la Web Semántica (Omelayenko, 2001).

5.2.3 Validación en varios dominios.

5.2.3.1 Hipótesis de validación

Se ha realizado una pequeña validación de la metodología descrita en dos dominios médicos (cáncer de mama y leucemia), y uno informático (Cifrado de datos). Aunque en este trabajo no se ha realizado un experimento estadísticamente significante se han realizado tres casos de estudio para comprobar si el planteamiento de la metodología iba por buen camino.

El objetivo principal de estos experimentos es el de analizar cómo el sistema es capaz de aprender y sugerir el conocimiento correcto a través de diferentes sesiones.

El estudio de los resultados se ha basado en una medida que hemos denominado *exactitud* del conocimiento propuesto por el sistema. La medida de exactitud es el resultado de la división entre la cantidad de entidades de conocimiento sugeridas por el sistema y que fueron aceptadas por el usuario experto y el la cantidad total de entidades de conocimiento existentes en el texto.

Exactitud: (Conocimiento Aceptado por el Experto/Conocimiento Total)*100

En base a esta medida, se han formulado las hipótesis siguientes:

Hipótesis 1 (Similaridad): "El sistema obtendrá una exactitud similar para cada experto". Los expertos tienen una experiencia similar en el tópico y han trabajado en los

mismos textos, luego se espera que obtengan resultados similares a través de los experimentos. En términos cuantitativos nos referimos a que obtendrán una exactitud del sistema similar.

Hipótesis 2 (Utilidad): "El sistema es útil para la construcción de la ontología". El sistema es capaz de sugerir entidades de conocimiento útiles para la construcción de ontologías, esto es, entidades de conocimiento relevantes dentro del dominio en cuestión.

Hipótesis 3 (Aprendizaje incremental): "El sistema aprende a lo largo del experimento". El sistema aprende a lo largo de las sesiones de los experimentos, y por lo tanto, sugerirá más entidades de conocimiento a través del entrenamiento en el dominio.

5.2.3.2 Cáncer de mama.

El cáncer de mama es el tipo de cáncer más frecuente en las mujeres. El diagnóstico precoz y la importancia de conocer los métodos de detección y prevención es la clave para la disminución de la incidencia de este tipo de cáncer. La ontología obtenida a partir de este experimento contiene información sobre como una mujer puede prevenir y explorar su pecho.

El experimento ha sido llevado a cabo por cuatro estudiantes de doctorado de informática que han utilizado el sistema con un corpus de 4648 palabras distribuido en un total de 310 frases. Este corpus está compuesto por un único texto que trata sobre la detección y la prevención de cánceres mamarios.

Para la consecución del experimento, el corpus fue dividido en 6 fragmentos, (cada uno teniendo aproximadamente el mismo número de frases), para un total de 310 frases.

Los cuatro individuos utilizaron sistemas independientes entre sí, esto es, no compartieron las bases de conocimiento en las que se basa el sistema. Los cuatro sistemas utilizados partieron con estas bases de conocimiento vacías, para poder ver cómo el sistema va aprendiendo a lo largo de las sesiones en un mismo dominio.

Resultados

En la tabla 5-1 y la figura 5-1, podemos ver los resultados parciales por cada fragmento, que representan la *exactitud* en cada uno de los 6 fragmentos de texto. La tabla 5-2 y figura 5-2 incluyen los resultados acumulados del proceso de aprendizaje, que representan la *exactitud acumulada* del sistema a lo largo del texto procesado, es decir, el primer valor será el correspondiente al fragmento primero, el segundo el correspondiente al fragmento primero y el segundo, y el último será el correspondiente a todo el texto completo.

Fragmento	Individuo 1	Individuo 2	Individuo 3	Individuo 4
1	44'69	51'07	44'44	49'15
2	67'2	72'79	68'34	67'34
3	74'63	90'32	63'06	76'69
4	83'44	83'49	64'05	79'66
5	83'41	87'03	77'89	88'95
6	82'30	88'0	82'47	92'55

Tabla 5-1. Evolución de la exactitud del sistema en el experimento

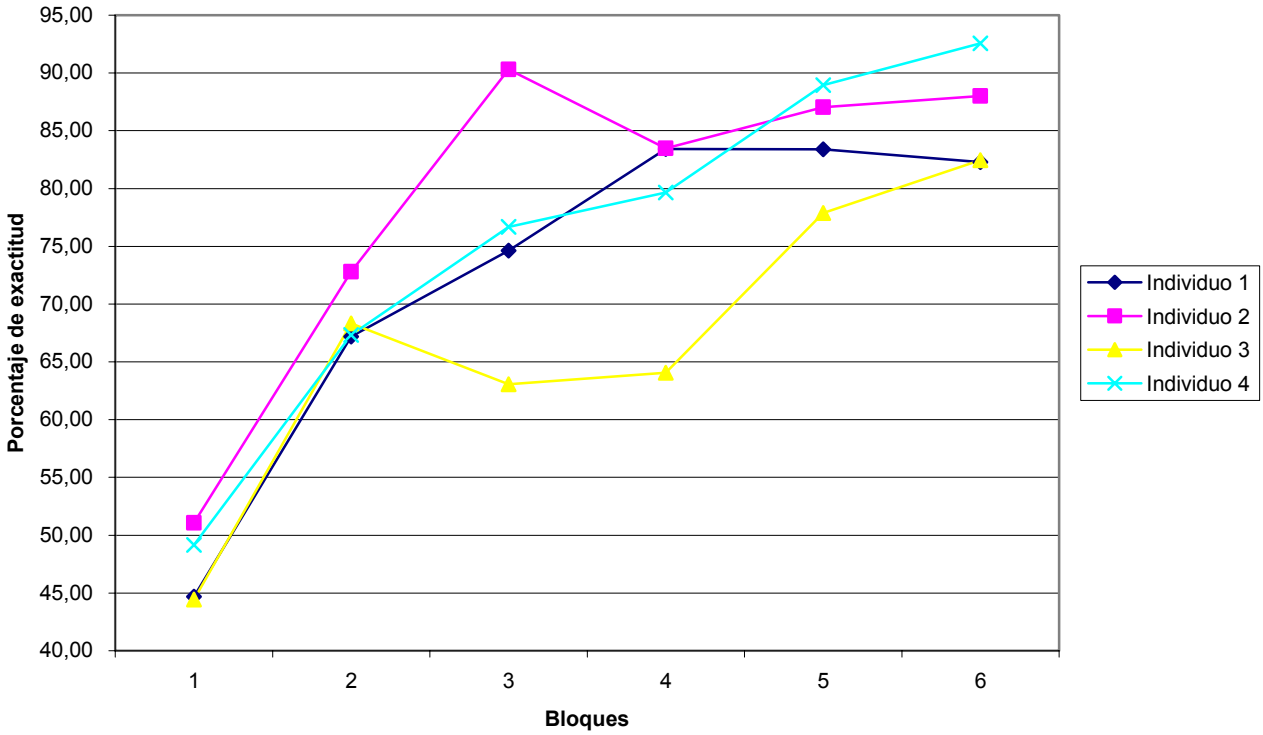

Figura 5-1. Evolución de la exactitud del sistema en el experimento

Fragmento	Individuo 1	Individuo 2	Individuo 3	Individuo 4
1	44'69	51'07	44'44	49'15
2	55'61	61'81	56'98	57'40
3	61'03	69'02	58'77	62'06
4	66'35	72'18	60'30	65'87
5	70'60	75'37	64'95	71'18
6	72'15	77'18	67'03	73'49

Tabla 5-2. Exactitud acumulada del sistema en el experimento

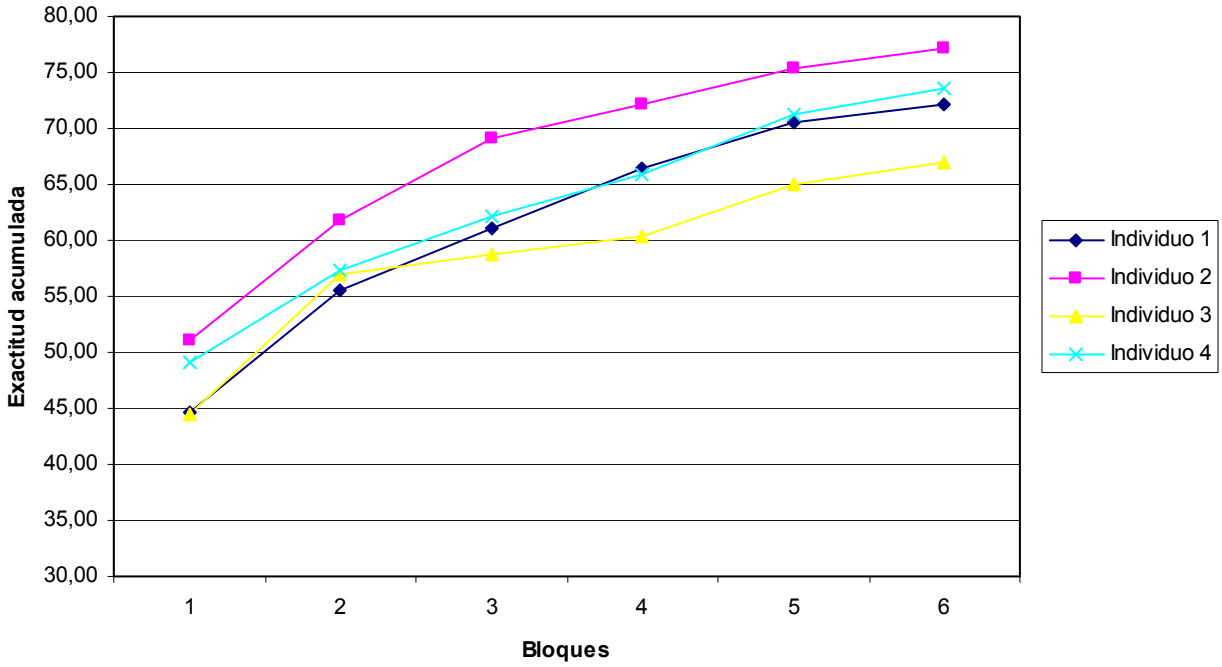

Figura 5-2. Exactitud acumulada del sistema en el experimento

La primera hipótesis (similaridad) se confirma, ya que los resultados obtenidos, así como las curvas de exactitud, son similares para cada estudiante (figuras 5-1 y 5-2). Además, se puede ver que el sistema permite a la gente con conocimiento similar obtener resultados similares. Esta es una propiedad deseable porque implica que los individuos que participaron en el experimento identificaron las mismas entidades de conocimiento aunque obtengan el conocimiento explícito

de diferentes maneras. Como se puede ver en la última entrada de la tabla 5-1 los individuos obtienen una *exactitud* similar para el último fragmento (82.30, 88.0, 82.47, 92.55) con una media del 86.33%. La última entrada de la tabla 5-2 indica la exactitud acumulada que ha conseguido el sistema en todo el texto. Como se puede ver también, los resultados son similares (72.5, 77.18, 67.03, 73.49) con una media del 72,55%.

La segunda hipótesis (utilidad) queda confirmada a partir de los resultados presentados en la Tabla 5-2, ya que el sistema ha inferido un buen porcentaje de entidades de conocimiento útiles para la creación de la ontología. En particular, el sistema ha obtenido el mejor resultado para el individuo 2, con un 77.18%. Además, el resultado medio es 72.55%. De acuerdo con las condiciones iniciales del experimento, este resultado puede considerarse como muy bueno, teniendo en cuenta que la base de conocimiento inicial estaba vacía.

La hipótesis tercera (aprendizaje incremental) se confirma debido al aumento de la exactitud del sistema a través de los fragmentos de texto. Como las bases de conocimiento iniciales estaban vacías, el sistema no fue capaz de sugerir conocimiento en los primeros fragmentos de texto. Las tablas y figuras confirman esta hipótesis, ya que se puede observar cómo el sistema sugiere más conocimiento acertado a través de los fragmentos de texto. Más concretamente, en la *exactitud acumulada* (tabla 5-2), el sistema obtiene un resultado menor del 50% en el primer bloque (44.69, 51.07, 44.44, 49.15), aumentando éste en los bloques siguientes. Debido a que en los fragmentos siguientes el sistema posee más conocimiento, es capaz de hacer mejores sugerencias, como se puede ver para el bloque 6, donde la *exactitud acumulada* del sistema es mayor del 75%.

Las ontologías creadas con este experimento han servido de base para la creación de un sistema basado en conocimiento para la asignación de tratamientos de cáncer de pecho en las unidades de oncología realizado por el grupo de investigación de Tecnologías del Conocimiento y Modelado Cognitivo (TECNOMOD) de la Universidad de Murcia. En la figura 5-3 se muestra una parte de la ontología extraída con este texto y ampliada con el conocimiento de expertos en oncologías para el desarrollo del sistema.

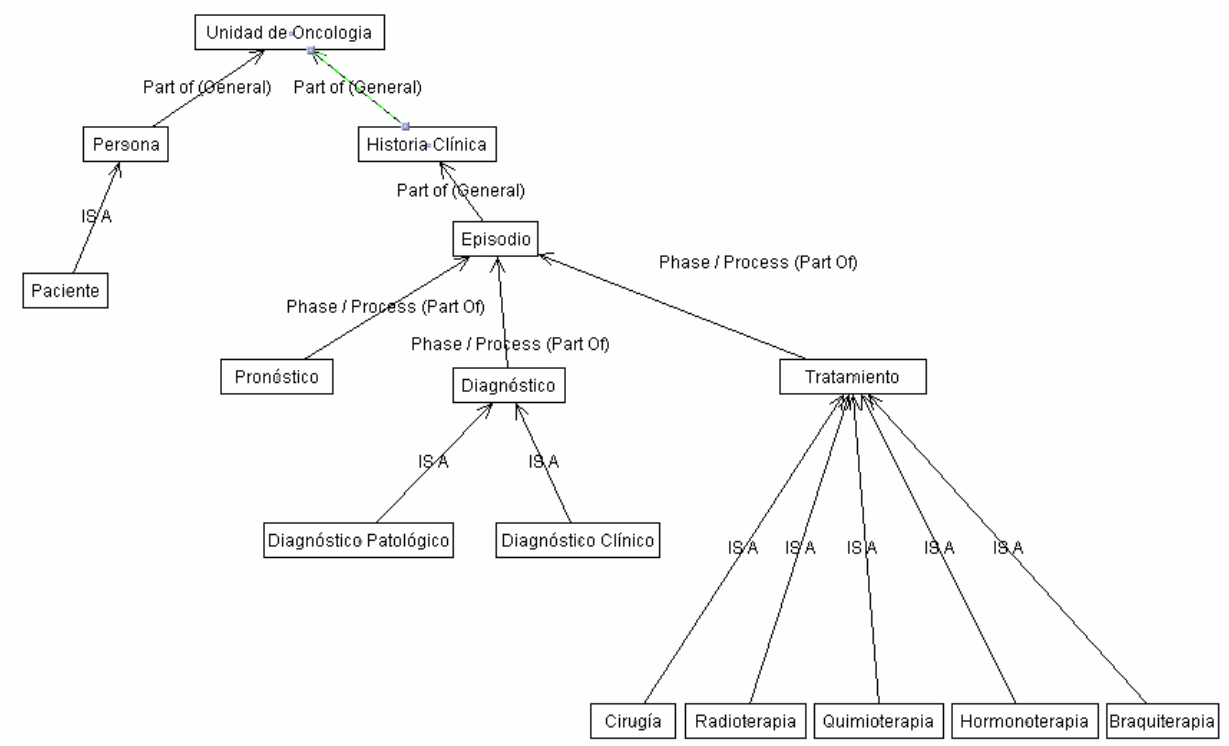

Figura 5-3. Parte de la ontología para la asignación de tratamientos de cáncer de mama

5.2.3.3 Leucemia

Se ha realizado otro experimento en el dominio de la leucemia utilizando cuatro alumnos de doctorado a los cuales se le proporcionó el sistema vacío para que pudiesen crear ontologías a partir de cinco textos (Tabla 5-3) que trataban sobre el dominio de la Leucemia.

Texto	Tema que trata	Frases	Palabras
1	La Leucemia y su Tratamiento	47	954
2	Leucemia, diagnóstico y tratamiento	74	1270
3	Tipos de Leucemia	18	158
4	Evaluación de casos de leucemia	28	675
5	Leucemia, síntomas, diagnóstico y tratamiento	171	2626
Total		**338**	**5683**

Tabla 5-3. Textos utilizados en el experimento de leucemia

Como se puede observar el número de palabras totales del corpus es 5683, que están distribuidas en un total de 5 textos distintos.

Los cuatro individuos participantes en el experimento utilizaron sistemas independientes entre sí, como en el caso del experimento anterior. La diferencia principal con el experimento anterior es que el corpus de este experimento está formado por 5 textos distintos, y no por distintos fragmentos dentro de un mismo texto.

Con este experimento queremos constatar que el sistema aprende a través de los textos de un mismo dominio, teniendo un mejor ratio de acierto en las sesiones futuras. El ejemplo anterior ha demostrado esta hipótesis dentro de un mismo texto y este experimento intenta demostrarlo entre textos distintos de un mismo dominio.

Resultados

En la tabla 5-4 y la figura 5-4, podemos ver los resultados parciales por cada texto. La tabla 5-5 y figura 5-5 incluyen los resultados acumulados del proceso de aprendizaje.

Texto	Individuo 1	Individuo 2	Individuo 3	Individuo 4
1	22,05	25,48	62,86	34,54
2	30,60	32,62	58,23	48,29
3	22,50	19,05	59,26	37,14
4	15,67	16,00	59,94	36,64
5	31,28	26,80	66,04	55,00

Tabla 5-4. Evolución de la exactitud del sistema en el experimento de leucemia

Texto	Individuo 1	Individuo 2	Individuo 3	Individuo 4
1	22,05	25,48	62,86	34,54
2	27,03	29,59	58,23	42,45
3	26,76	28,39	59,26	42,07
4	24,91	26,49	59,94	40,93
5	28,08	26,64	66,04	45,50

Tabla 5-5. Exactitud acumulada del sistema en el experimento de leucemia

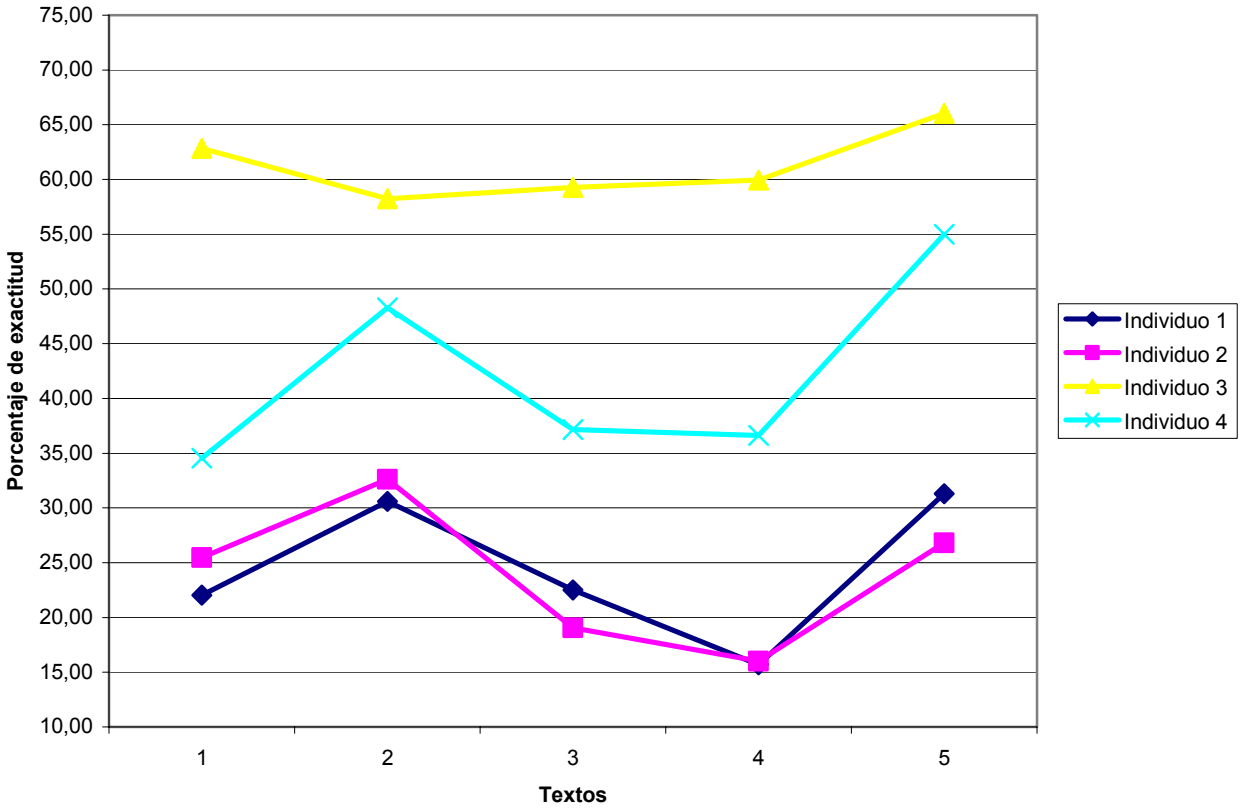

Figura 5-4. Evolución de la exactitud del sistema en el experimento de leucemia

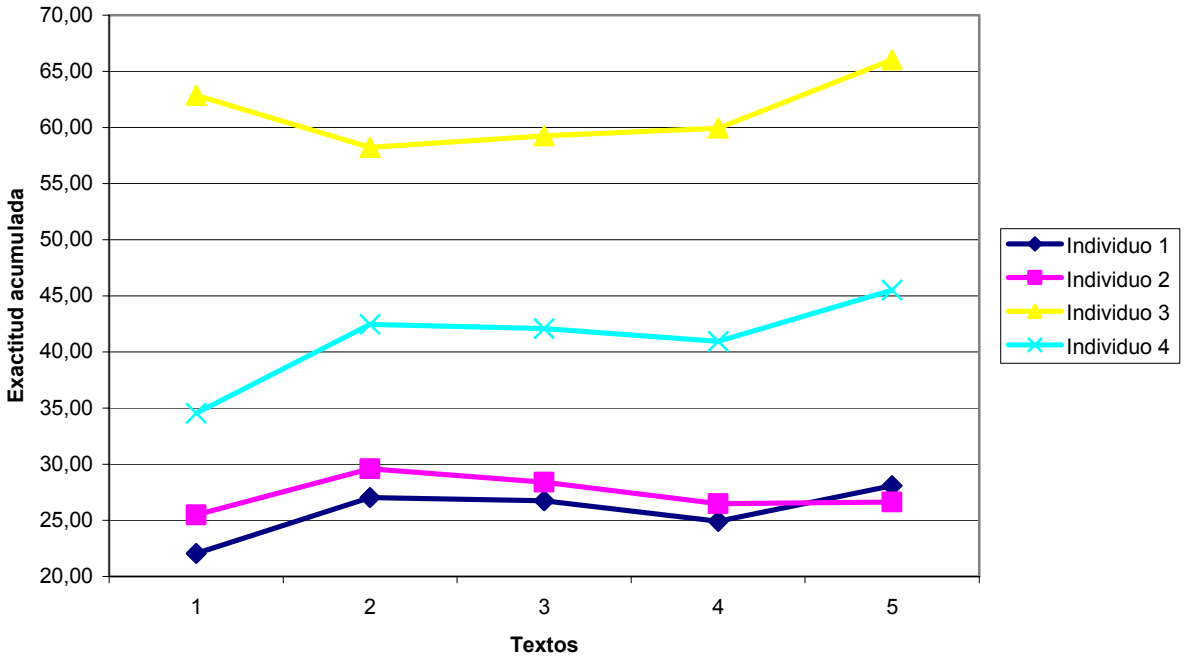

Figura 5-5. Evolución de la exactitud acumulada del sistema en el experimento de leucemia

La primera hipótesis (similaridad) no se confirma en los resultados obtenidos debido a que las curvas de exactitud son diferentes para cada estudiante, aunque hay dos estudiantes donde la curva de exactitud es similar y muy próxima (figuras 5-1 y 5-2).

Podemos observar las diferencias que surgen en este experimento entre los participantes en el experimento. Mientras que hay resultados bastante buenos que confirman la utilidad del sistema (hipótesis 2), hay otros menos buenos que aunque demuestran la utilidad de la herramienta debido a que tienen una tasa de acierto, esta no sería demasiado alta. Esto puede ser debido a la forma de extracción de conocimiento por parte de estos alumnos y a que todavía el sistema no contiene mucha información debido a que sólo ha sido entrenado con un corpus relativamente pequeño.

La hipótesis tercera (aprendizaje incremental) se confirma debido al aumento de la exactitud parcial del sistema a través de los distintos textos. Observando la figura 5-4, podemos

ver cómo tres de los cuatro participantes obtienen un descenso de la exactitud del sistema en los textos 3 y 4 que puede ser debido a que la sintaxis en esos textos es más complicada, además de que los términos que aparecen en ellos no tienen mucha relación con los textos ya analizados. Aún así, se puede ver que el sistema va aprendiendo a través de las sesiones debido a que los expertos obtienen un rendimiento en el último texto superior a los otros cuatro.

También esta hipótesis se demuestra observando la figura 5-5, en la cual se puede observar que la exactitud acumulada del sistema va creciendo a través de los textos con el pequeño descenso en los textos 3 y 4 que puede ser debido a las razones expuestas en el párrafo anterior.

En la figura 5-6 podemos observar una parte de la ontología adquirida por uno de los participantes en el experimento realizado.

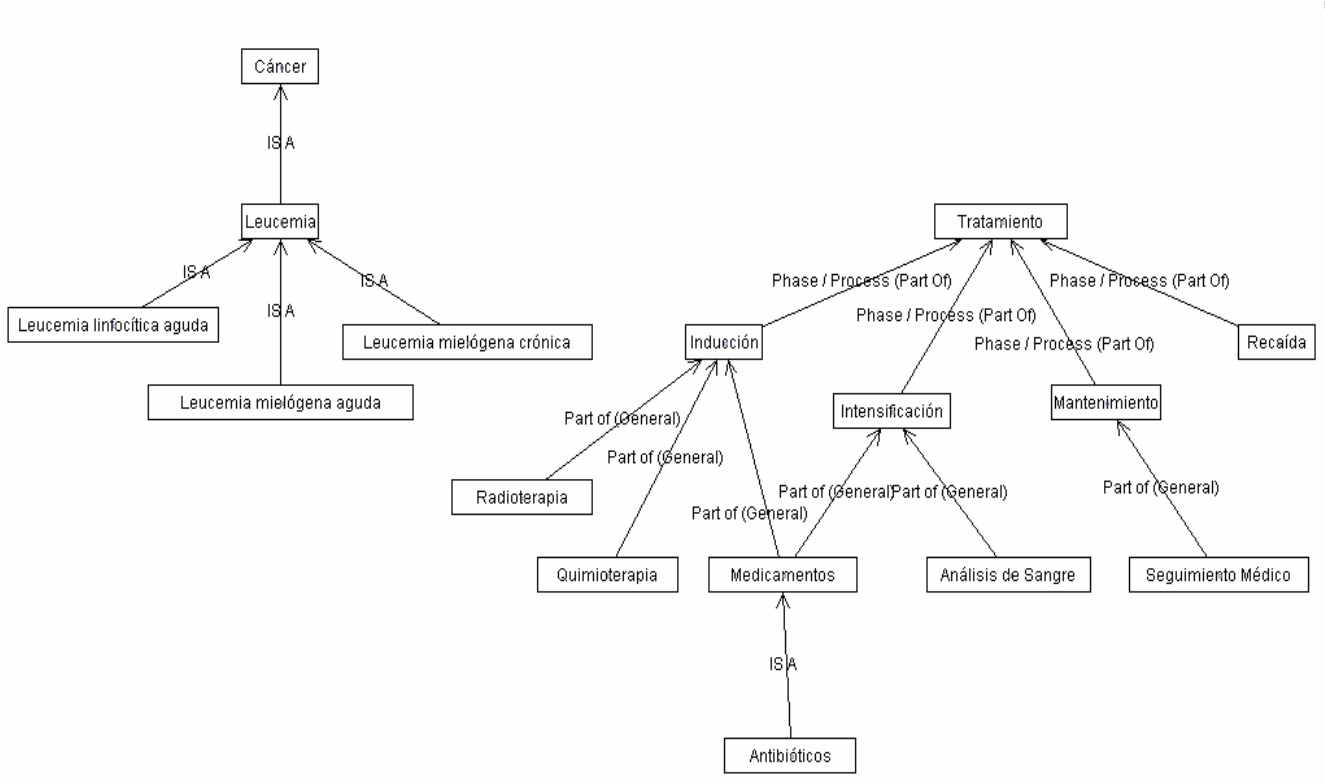

Figura 5-6. Parte de la ontología obtenida en el estudio del dominio de la leucemia

5.2.3.3 Cifrado de datos

Por último, se realizó un experimento con alumnos de quinto curso de la Facultad de Informática para que utilizasen el sistema para crear ontologías en el dominio del cifrado de datos a través de cinco textos distintos del dominio del cifrado de datos (tabla 5-6).

Texto	Tema que trata	Frases	Palabras
1	Introducción a la criptografía	99	1612
2	Programa PGP	43	720
3	Sistemas de clave pública	35	827
4	Firmas digitales	75	1450
5	Sistemas y tipos de cifrado	45	893
Total		**297**	**5502**

Tabla 5-6. Textos utilizados en el experimento de cifrado de datos

Como se puede observar, el número de palabras totales del corpus es 5502 que están distribuidas en un total de 5 textos distintos. Este experimento fue realizado por ocho alumnos de quinto curso de la Facultad de informática, a los cuales se les proporcionaron sistemas vacíos e independientes entre sí.

Utilizando como medida la **exactitud** definida anteriormente queremos afianzar la demostración de las hipótesis definidas, donde se quiere constatar que el sistema aprende a través de los textos de un mismo dominio, teniendo un mejor ratio de acierto en las sesiones futuras.

En la tabla 5-7 y la figura 5-7, podemos ver los resultados parciales por cada texto. La tabla 5-8 y figura 5-8 incluyen los resultados acumulados del proceso de aprendizaje.

Texto	Individuo 1	Individuo 2	Individuo 3	Individuo 4	Individuo 5	Individuo 6	Individuo 7	Individuo 8
1	20,21	20,20	55,03	49,52	52,38	35,02	37,50	30,71
2	22,73	13,74	69,39	45,33	50,67	45,79	50,44	33,98
3	30,15	34,15	83,72	66,67	50,00	63,11	60,00	36,28
4	32,98	38,95	87,05	80,67	65,41	61,16	60,86	45,16
5	18,72	33,33	84,75	83,33	66,67	68,05	58,96	39,02

Tabla 5-7. Evolución de la exactitud del sistema en el experimento de cifrado

Texto	Individuo 1	Individuo 2	Individuo 3	Individuo 4	Individuo 5	Individuo 6	Individuo 7	Individuo 8
1	20,21	20,20	55,03	49,52	52,38	35,02	37,50	30,71
2	20,93	18,58	57,98	48,42	51,93	38,19	41,64	31,65
3	22,78	22,30	61,92	51,33	51,47	43,68	45,85	32,77
4	26,47	28,05	70,24	60,33	56,07	50,88	52,35	37,38
5	25,30	28,79	72,03	62,84	57,32	55,74	54,17	37,61

Tabla 5-8. Exactitud acumulada del sistema en el experimento de cifrado

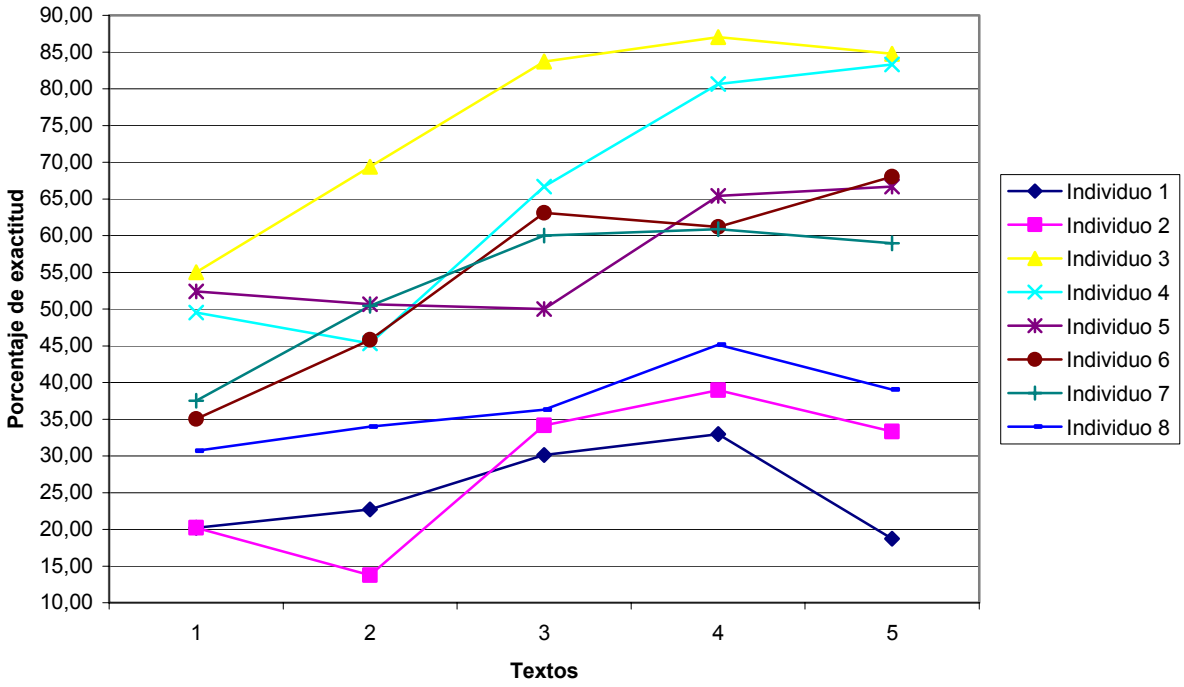

Figura 5-7. Evolución de la exactitud del sistema en el experimento de cifrado

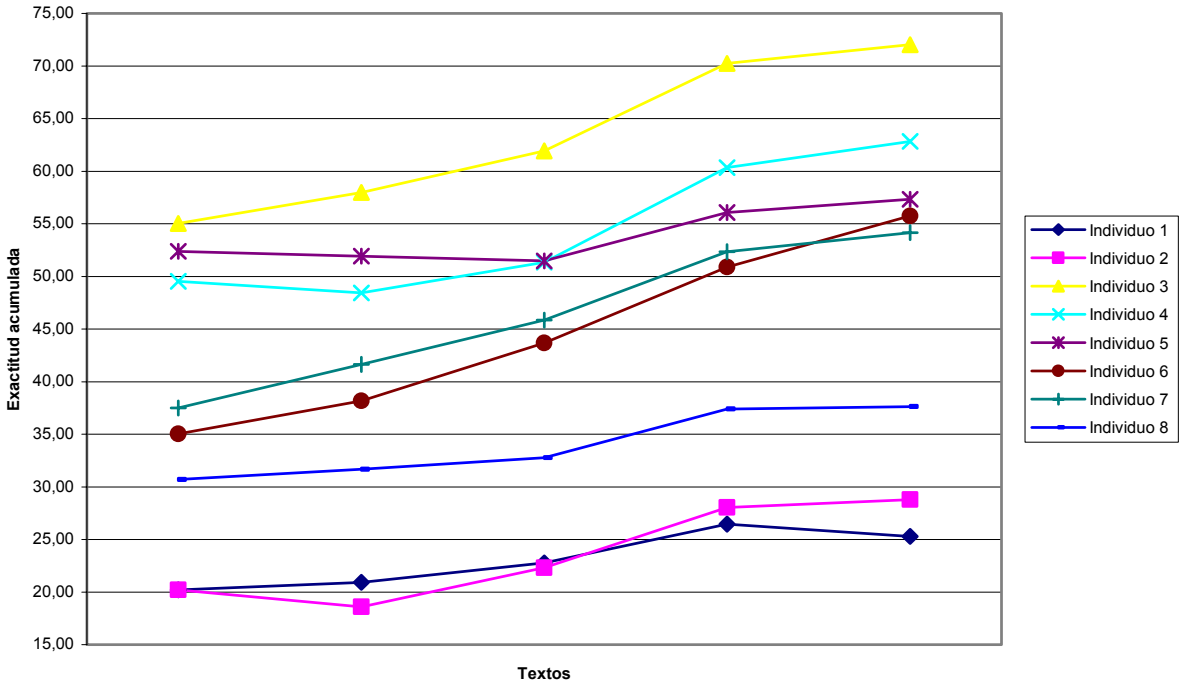

Figura 5-8. Evolución de la exactitud acumulada del sistema en el experimento de cifrado

La segunda hipótesis (utilidad) queda confirmada a partir de los resultados presentados en la Tabla 5-2, ya que el sistema ha inferido un buen porcentaje de entidades de conocimiento útiles para la creación de la ontología. En particular, el sistema ha obtenido el mejor resultado para el estudiante 3, con un 72.03%. El resultado medio es 49.23% debido a que tres participantes han obtenido resultados menores del 40%. Cinco de los ocho expertos han obtenido resultados superiores al 54%, con lo que el sistema adquiere entidades de conocimiento útiles para la construcción de las ontologías.

La hipótesis tercera (aprendizaje incremental) se confirma debido al aumento de la exactitud parcial del sistema a través de los distintos textos. Observando la figura 5-7 podemos ver cómo la mayoría de los estudiantes obtienen un aumento de la exactitud del sistema a través de los textos, debido a que tratan del mismo dominio y los términos y relaciones utilizadas deben ser similares en estos textos. También esta hipótesis se demuestra observando la figura 5-8, en la cual se puede observar que la exactitud acumulada del sistema va creciendo a través de

los textos en prácticamente todos los expertos, obteniéndose una curva de aprendizaje similar (Figura 5-8).

5.2.3.4 Conclusiones

Como hemos podido ver a través de los experimentos, alguna de las hipótesis queda confirmadas, si bien de manera no estadísticamente significativa, a través de los experimentos. Más concretamente, la hipótesis 1 se confirma en el primer y último experimento debido a que la mayoría de expertos obtienen resultados similares.

La hipótesis de utilidad (hipótesis 2) queda demostrada en los tres experimentos, en los que se obtienen distintos ratios de exactitud, obteniendo el primer experimento el mejor ratio y el segundo el peor.

Los malos resultados obtenidos por parte de expertos en el segundo y tercer experimento, se pueden deber a que los sistemas no contienen mucha información debido a que el entrenamiento se ha realizado con un corpus pequeño (aproximadamente 5000 palabras) y a que los textos, aunque pertenezcan al mismo dominio, son diferentes entre sí y, por lo tanto, contendrán mucho conocimiento nuevo con respecto a los anteriores.

La hipótesis de aprendizaje incremental (hipótesis 3) queda también demostrada en todos los experimentos, donde se observa que el sistema aprende a través de las sesiones, dentro de un mismo texto (primer experimento) y entre textos distintos de un mismo dominio.

En definitiva, este sistema ha dado muy buenos resultados con un entrenamiento pequeño del corpus, mientras que otros experimentos necesitan el entrenamiento de corpus de hasta 300.000 palabras (Ogata & Nigel Collier, 2004) para obtener buenos resultados.

5.3 RECONOCIMIENTO DE ÓRDENES EN LENGUAJE NATURAL APLICADO A LA ROBÓTICA MÉDICA.

5.3.1 Introducción

Se ha realizado una pequeña ampliación en el modelo para su aplicación dentro del reconocimiento de órdenes en lenguaje natural a través del habla. Para ello, se escogió el dominio quirúrgico, realizando un trabajo que persigue el desarrollo de un sistema para la realización de operaciones quirúrgicas que interactúa con el cirujano a través de comandos de voz (expresados en lenguaje natural).

5.3.2 Descripción del problema y objetivos.

La aplicación de sistemas automáticos en medicina tiene una larga tradición. Los primeros fueron empleados para propósitos diagnósticos. Por ejemplo, TC (Tomografía Computarizada) y RMN (Resonancias magnéticas nucleares) son los ejemplos más conocidos de test clínicos automáticos, que utilizan una computadora, que emplean la computadora para elaborar y transformar señales clínicas en imágenes. Hace aproximadamente diez años aparecieron los primeros robots aplicados a la medicina. El AESOP1000 (Automated Endoscopic System for Optimal Positioning) puede considerarse como el primer robot médico, construido por Computer Motion Inc. Dos años después, una nueva versión salió al Mercado con el nombre de AESOP2000, que podía manejarse con órdenes simples a través de la voz (Koneckny, 1996; Kukleta, 1998).

La cirugía endoscópica es una aplicación importante de la robótica en la medicina. Entre los robots para la endoscopia podemos destacar los sistemas robóticos quirúrgicos DAVINCI y ZEUS, que es una evolución del robot AESOP (D'Attellis *et al*, 2002; Reuthebuch *et al*, 2002; Austad *et al*, 2001). Estos sistemas pueden tanto operar a través de la transmisión de datos clínicos para el diagnóstico y la terapia, como implementar la "tele-cirugía". Sin embargo, la

"tele-cirugía" robótica está todavía en sus comienzos, por estar afectada severamente por el retraso entre la transmisión y la recepción.

La ortopedia es otro campo donde se han utilizado varios sistemas robóticos, para realizar tareas como la inserción de tornillos y clavos para reducir las fracturas. Dos robots de este tipo son el ACROBOT y el PROBOT: ACROBOT es un robot semi activo para la cirugía de rodilla, mientras que PROBOT es un robot activo para operaciones en la próstata (Harris *et al*, 1997). En los últimos años se han desarrollado sistemas para permitir un buen rendimiento para tareas radio-quirúrgicas, como la irradiación de zonas de tumores en el cerebro en otras partes del cuerpo del paciente (Dinsmore *et al*, 1996; Beatty *et al*, 1996).

La cirugía mini-invasiva se ha convertido en un campo muy popular en la práctica médica. Esta clase de cirugía actúa a través de cortes pequeños, obteniendo los mismos resultados que la cirugía tradicional, pero evitando bastantes cortes externos o internos. Las técnicas de cirugía mini-invasiva superan las técnicas de las operaciones tradicionales, por lo que la cirugía mini-invasiva ha supuesto una revolución del dominio quirúrgico. La principal ventaja es la minimización de los traumas en los tejidos sanos, por lo que el tiempo de hospitalización y los riesgos de operaciones post-operatorias disminuyen (Tendick&Cavusoglu, 1997).

En la cirugía laparoscópica y torascópica, las operaciones se llevan a cabo insertando instrumentos quirúrgicos en el cuerpo del paciente a través de uno o mas agujeros y observando la operación a través de cámaras de micro-televisión que se inserta a través de otro agujero. Los robots utilizados en laparoscópica son los necesarios para mover la cámara. Varios programas de investigación están investigando actualmente en este campo. En la Universidad de California en Berkeley, se está desarrollando un proyecto para construir una estación para cirugía laparoscópica. Además, los investigadores del Centre of Nuclear Researches of Karlsruhe y de la Universidad de Tübingen (Alemania) están estudiando varias clases de manipuladores quirúrgicos y brazos para mover las cámaras y los instrumentos quirúrgicos.

Otro campo importante es la cirugía estereotáctica, que emplea sistemas para la reconstrucción tridimensional de imágenes, por lo que puede mostrar al cirujano con información espacial el punto donde él está operando. Estos sistemas están normalmente integrados con dispositivos robóticos para lograr la posición exacta de los instrumentos médicos y la ejecución de la operación. La investigación en el campo de la cirugía estereotáctica se centra en la implementación de un sistema de tele-cirugía que permita al cirujano llevar a cabo operaciones y diagnósticos de pacientes situados en un lugar distinto. Por supuesto, tal sistema necesitaría una red de transmisión de datos para transferir información multimedia en tiempo real. La cirugía mini-invasiva supone con todo, uno de los campos mas propicios para la investigación en el dominio de la tele-cirugía.

Gracias a todas las tecnologías de realidad virtual, sistemas de imagen, computadoras y manipuladores, los cirujanos pueden usar imágenes tridimensionales. Además, la transmisión de sensaciones táctiles resulta ya factible (Fearing *et al*, 1997; Gray & Fearing, 1996). Algunas de estas aplicaciones se han llevado a cabo. En 1991, Jaron Lanier creó un modelo computerizado tridimensional para el nervio óptico. En 1994 la primera operación de neurocirugía en 3D se llevó a cabo en el hospital Brigham de Boston, y el cirujano Belga Jacques Himpens ejecutó la primera operación de tele-cirugía. En septiembre de 1995, el cirujano Enrico Pisani del laboratorio de Tele-Robótica de la Politécnica de Milán, operó un paciente, situado a 10 kilómetros de distancia, a través de un robot conectado por fibras ópticas. La operación consistía en una biopsia, tele-robotizada, llevada a cabo con una anestesia total. La operación del paciente se ejecutó por el robot asistente después de más de 1200 simulaciones en modelos con el mismo procedimiento, y después de tres años de pruebas en muñecos, para conseguir que el sistema fuera seguro, excluyendo todas las posibles deficiencias técnicas de las máquinas y del software.

La cirugía laparoscópica y mini-invasiva son campos donde los desarrollos de la cirugía moderna son más evidentes. Sin embrago, detrás de estas grandes ventajas, la cirugía mini-invasiva ha creado también nuevos problemas: la pérdida de visión tridimensional, de la sensibilidad táctil y de la coordinación mano-ojo. La realidad virtual se centra en la resolución de esta clase de problemas. Muchos sistemas han sido diseñados para este propósito por diferentes investigadores. Algunos de estos sistemas operan con sistemas computerizados, que procesan señales digitales para eliminar el posible temblor de la mano del cirujano.

Otro campo del desarrollo de la cirugía robótica es la cirugía guiada por ordenador, que está basada en un principio similar de los pilotos automáticos en aviones. Después de estudiar, estandarizar y almacenar las posibles situaciones de cierta operación quirúrgica, un algoritmo de ayuda a la decisión se emplea para simular la operación y elegir la acción óptima. El objetivo final de esta técnica es programar y ejecutar la operación quirúrgica completa automáticamente. Por eso el cirujano sólo actúa como un controlador externo, que está preparado para tomar el control si ocurre algo inesperado (Autret, 1996).

El problema a resolver con esta aproximación está relacionado con la técnica explicada arriba, por lo que el objetivo de este trabajo es automatizar operaciones quirúrgicas simples, donde el cirujano puede interactuar con el robot a través del habla.

La finalidad principal del trabajo presentado aquí es simular operaciones quirúrgicas en un entorno de realidad virtual, en la que el usuario debe emitir comandos al manipulador robótico a través de la voz. Además, el usuario puede observar la simulación de las acciones ordenadas al sistema, para una mejor monitorización de la operación.

5.3.3 Descripción del framework para la simulación de operaciones quirúrgicas.

La figura 5-9 representa la arquitectura del sistema (framework) desarrollado. El primer módulo procesa los comandos del cirujano expresados en lenguaje natural a través de la voz. Estos comandos son entonces procesados por el sistema de construcción de ontologías mostrado anteriormente para obtener una ontología que pueda traducirse en una acción posible del robot. Una vez obtenida esta acción, el sistema realizará una planificación (utilizando STRIPS) del movimiento que el brazo Robot debe realizar. Estos movimientos se simulan en un entorno software realizado en Java.

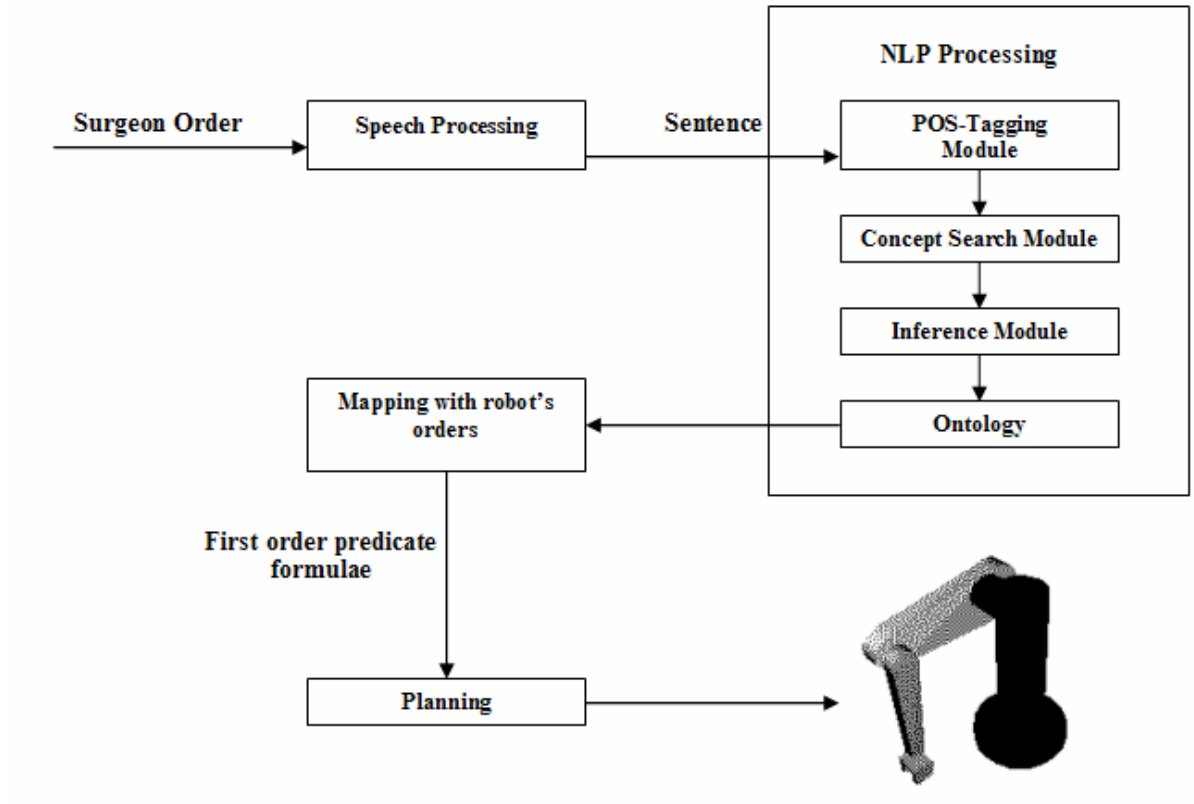

Figura 5-9. Módulos del sistema

5.3.3.1 Módulo de procesamiento de la voz

Existen varios reconocedores de voz comerciales (IBM ViaVioce, Dragon Naturally Speak, etc.), que pueden trabajar en dos modos distintos. El primer modo es el denominado modo de reconocimiento de comandos, que emplea gramáticas predefinidas, que representan un lenguaje simple para reconocer frases simples del usuario. El segundo modo se denomina de dictado y obtiene una lista con las palabras más probables que pueda haber dictado el usuario.

Este experimento se realizó empleando el IBM ViaVoice como reconocedor de voz, utilizando el modo de dictado para obtener todas las palabras pronunciadas por el cirujano. La integración del IBM ViaVoice en el sistema completo se realizó utilizando la API de Java Speech, que especifica una interfaz para soportar comandos y controlar los reconocedores, sistemas de dictado y sintetizadores de voz.

IBM ViaVoice ha sido utilizado para el reconocimiento de vez en otros sistemas dentro del campo de la Inteligencia Artificial como el trabajo presentado en (Dorai G.&Yacoob, 2002), donde se utilizan lo que se denomina *Embedded Grammar Tags* para representar todas las posibles órdenes o comandos que pueden ser pronunciadas por los usuarios para obtener varios contenidos dentro de un portal de Web Semántica.

5.3.3.2 Módulo de procesamiento de lenguaje natural

El módulo de procesamiento del lenguaje natural utilizado para reconocer las órdenes en lenguaje natural está basado en la metodología explicada en el capítulo 3 de este trabajo.

Esta metodología se utiliza en este ejemplo de manera automática y no-supervisada, es decir, se pretende construir la ontología automáticamente sin ayuda del experto. Por eso no se pide confirmación del conocimiento adquirido a través del sistema. Anteriormente, el sistema ha sido entrenado en contextos relacionados con la cirugía, para que cuando se utilice de una manera automática, pueda obtener el conocimiento suficiente como para reconocer la orden.

El idioma elegido para la realización de este prototipo es el inglés, debido a la colaboración con una universidad italiana, Università degli studi di Udine.

Como podemos observar en la figura 5-9, el sistema constaba de tres fases secuenciales:

POS-Tagging

El objetivo principal de este módulo es obtener la categoría gramatical de cada palabra en la frase actual. Para ello, se ha utilizado el tagger descrito en el capítulo 2 de este trabajo, donde se presenta un POS-Tagger basado en árboles C4.5.

Por ejemplo, si se introduce la siguiente frase "Now, it is necessary to cut slowly the skin using the scalpel", el sistema obtendrá las siguiente categorías gramaticales:

Now [Adv], [Punt] it [Pro] is[Verb (aux)] necessary [Adj] to [Part] cut [Verb (lexical)] slowly [Adv] the [Det] skin [Noun] using [Verb(Lexical)] the [Det] scalpel[Noun]

Fase de búsqueda de conceptos (Concept search module)

A través de este módulo se identifican las expresiones lingüísticas que representen conceptos del dominio de la cirugía. La asociación entre las expresiones lingüísticas y los conceptos asociados se almacenan en la base de conocimiento de conceptos en el entrenamiento del sistema previo.

Por ejemplo, si la base de conocimiento de conceptos contiene las expresiones mostradas en la tabla 5-9, esta fase obtendrá de la frase "Now, it is necessary to cut slowly the skin using the scalpel", los conceptos *Cut*, *Scalpel* y *Skin*.

Linguistic expressions	Concept Associated
Cut	Cut
…	…
Scalpel	Scalpel
Skin	Skin
Skull	Skull
…	….

Tabla 5-9. Base de conocimiento de conceptos.

Fase de inferencia (Inference Module)

La idea principal sobre el modus operandi de esta fase es que en lenguaje natural, las relaciones entre conceptos suelen estar asociadas a los verbos (Ruíz-Sánchez *et al*, 2003; Valencia-García *et al*, 2004a). Aunque este subproceso se encarga principalmente de obtener las relaciones entre conceptos, puede utilizarse también para adquirir otras entidades de conocimiento como conceptos, atributos y valores. En esta fase se utiliza una base de conocimiento que contiene las expresiones lingüísticas que representan las relaciones entre conceptos genéricas (descritas en este trabajo), y por un subsistema de inferencia basado en reglas que obtenga los participantes en esas relaciones. Este módulo se ha descrito con más detalle en el capítulo 3 de este trabajo.

5.3.3.3 Mapeo de las ontologías adquiridas por el modulo NLP a órdenes del sistema

Se ha creado una ontología del dominio quirúrgico utilizando el sistema descrito en los capítulos anteriores mediante la inserción de un corpus relacionado con el dominio de la cirugía. La finalidad de esta ontología es representar el sistema y las acciones posibles a realizar en el sistema. En la Figura 5-10 podemos ver cómo se representan tres de las posibles acciones a realizar por el robot y su relación con los respectivos instrumentos necesarios para ejecutar dichas acciones.

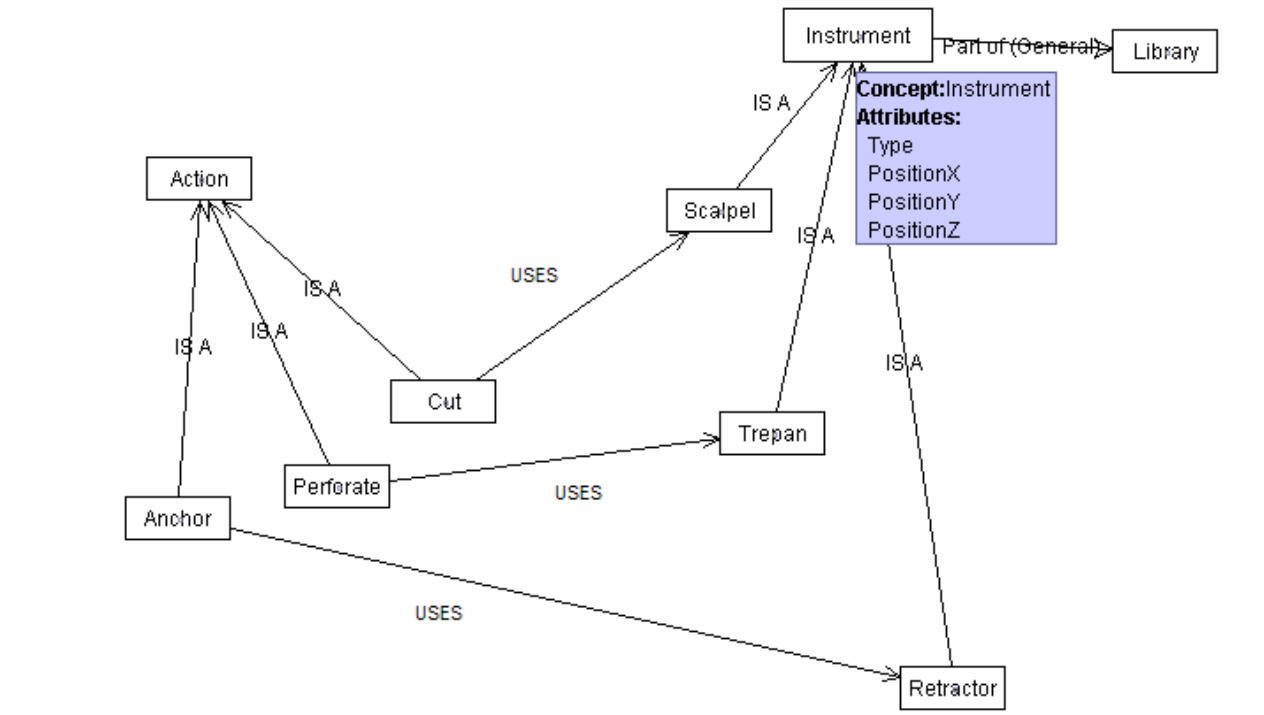

Figura 5-10. Parte de la ontología que representa el sistema.

La ontología anterior contiene también un conjunto de axiomas para chequear la integridad y completitud de la ontología adquirida. En la Tabla 5-10 se puede observar parte del conjunto de axiomas desarrollado.

Axiom	Description
if *IS_A_children_of*("Action").size()!=1 **then** NO_ACTION	Si en la ontología parcial adquirida no hay ningún concepto de la jerarquía de **Action**, o existe más de un concepto de dicha jerarquía, la acción no podrá llevarse a cabo.
if *IS_A_children_of*("Instrument").size()>1 **then** NO_ACTION	Si la ontología parcial contiene más de un concepto de la jerarquía **Instrument**, la acción no podrá llevarse a cabo.
if *Concept*("Cut") **&&** (*Instance*("Instrument")!="Scalpel") **then** NO_ACTION	Si el concepto **Cut** (Cortar) está presente en la ontología y hay alguna instancia del concepto **Instrument** diferente a **Scalpel**, la acción no podrá llevarse a cabo.
if *Concept*("Cut") && *IS_A_children_of*("Instrument").size()==0 **then {** new Concept("Scalpel"); new Relation("USES", "Cut", "Scalpel"); }	Si la ontología parcial contiene el concepto **Cut** y no existe ningún concepto de la jerarquía de **Instrument**, el sistema introduce el concepto **Scalpel** en la ontología parcial y la relación **USES** entre **Cut** y **Scalpel**.

Tabla 5-10. Axiomas utilizados para chequear las integridad, consistencia y completitud de la ontología

Una vez la ontología es consistente y completa, tiene que ser transformada en el estado final del planificador lineal implementado. Otro conjunto de axiomas es el que va a permitir realizar esta traducción. En la tabla 5-11 podemos ver algunos ejemplos de estos axiomas. La explicación de la fórmula del predicado lógico de primer orden se puede ver en la siguiente sección.

Axiom	Description

if *Concept*("Cut") and *Concept*("Scalpel") **then** FinalState= {
patient_status(skin,cut)
patient_status(skull,intact)
patient_status(tumour,present)
}
	Si la ontología representa la orden Cut (Cortar), el sistema debe de traducir la ontología parcial en el estado final que represente que el tumor está todavía presente, el cráneo está intacto y se ha cortado la piel del paciente.

if *Concept*("Anchor") and *Concept*("Retractor") **then** FinalState= {
patient_status(skin,open)
patient_status(skull,intact)
patient_status(tumour,present)
}
	Si la ontología representa la orden Anchor (Anclar), el sistema debe traducir la ontología parcial a un estado final que represente que el tumor está todavía presente, el cráneo (Skull) está intacto y la piel está cortada.

Tabla 5-11. Axiomas usados para la traducción de la ontología en un estado final

Una vez la ontología parcial ha sido traducida a un estado final, el sistema debe obtener un plan de los pasos necesarios para obtener el estado final usando un planificador basado en STRIPS.

5.3.3.4 Módulo de planificación STRIPS

STRIPS (Fikes & Nilsson, 1971) es un planificador lineal que intenta encontrar una secuencia de operadores en un espacio de modelos del mundo (world models) para transformar un modelo inicial dado a un modelo final donde se pueda probar que las fórmulas de que está formado son verdaderas. Un modelo del mundo se representa como un conjunto de fórmulas de predicados de primer orden y trabajo con modelos que contienen un gran número de fórmulas.

Para cada modelo del mundo, definimos un conjunto de operadores aplicables que transforman un modelo dado en otro. Un demostrador de teoremas puede encontrar una composición de operadores para transformar un modelo del mundo inicial a otro que satisfaga una condición de fin.

En STRIPS un modelo del mundo se representa por un conjunto de fórmulas bien formadas de predicado de lógica de primer orden. Cada operador en una solución corresponde a una acción cuya ejecución causa que el robot realice ciertos movimientos.

Cada operador en STRIPS se compone por:

- Un conjunto de **precondiciones**. Para ejecutar la acción relacionada con un operador, es necesario que las precondiciones se cumplen antes de que el operador pueda ser aplicado.

- Una lista de **borrados**, que es un conjunto de fórmulas que no serán verdad después de que se aplique, luego el planificador tiene que borrarla del modelo del mundo actual.

- Una lista de **añadidos**, que es un conjunto de formulas que serán verdad después de que el operador haya sido aplicado, luego el planificador debe añadirlas al modelo del mundo actual.

Por ejemplo, dos operadores para el problema del mundo de los bloques se muestran en la Tabla 4:

	STACK(x, y)	UNSTACK(x,y)
Preconditions:	CLEAR(y) HOLDING(x)	ON(x, y) CLEAR(x) ARMEMPTY
Add list:	ARMEMPTY ON(x,y)	HOLDING(x) CLEAR(y)
Delete list:	CLEAR(y) HOLDING(x)	ON(x, y) ARMEMPTY

Tabla 5-12. Dos operadores del problema del mundo de los bloques

En la Tabla 5-12, *CLEAR(block)*, *ON(block1,block2)*, *ARMEMPTY* y *HOLDING(block)* son fórmulas bien formadas.

STRIPS, como la mayoría de los planificadores modernos, ha sido aplicado al problema del mundo de los bloques como un benchmark efectivo (Slaney & Thiébaux , 2001). El problema del mundo de los bloques consiste en un número finito de bloques apilados en torres sobre una mesa suficientemente larga para contenerlos todos. La posición de las torres en la mesa es irrelevante. El problema de planificación del mundo de los bloques es cambiar un estado inicial a un estado final, moviendo un bloque cada vez desde la cima de una torre a otra torre o a la mesa. La solución óptima del problema de los bloques es realizarlo en el mínimo número de movimientos.

La mayoría de los planificadores han adoptado la representación de STRIPS, incluido el planificador GRT (Refanidis & Valvas, 2001) y el sistema de planificador Fast-Foward (Hoffmann, 2001).

Planificando operaciones de neurocirujía

Debido al hecho que el dominio de las operaciones quirúrgicas elegido para este trabajo es un dominio estructurado y conocido, hemos utilizado STRIPS para la planificación de operaciones quirúrgicas (simples). El problema es muy similar al del mundo de los bloques, donde los modelos del mundo se representan por un conjunto de formulas de lógica de primer orden, como se puede ver en la Tabla 5-13.

Fórmula	Descripción
Allocated(instrument, library)	El instrumento está colocado en la librería
Position(pos)	El tumor está situado en la posición representada por *pos*
grasped(instrument)	El robot tiene cogido el instrumento
Patient_status(part, status)	La parte del cuerpo del tumor tiene un estado, por ejemplo: Patient_status(skin,closed); Patient_status(tumour,present);

Tabla 5-13. El conjunto de fórmulas.

El estado inicial está formado por una formula que representa que todos los instrumentos están colocados en la librería, la posición del robot es X0, el robot no tiene cogido ningún instrumento, y el tumor está todavía presente. Las fórmulas del estado final se obtienen en la fase anterior y son diferentes para cada comando del cirujano. El estado final de la operación quirúrgica tiene que representar que los instrumentos están en su posición correcta en la librería, el tumor ha sido extirpado y la piel del paciente ha sido cerrada (ver Tabla 5-14).

Estado inicial	Estado final
Allocated(scalpel,library_scalpel)	Allocated(scalpel,library_scalpel)
Allocated(retractor,library_retractor)	Allocated(retractor,library_retractor)
Allocated(trepan,library_trepan)	Allocated(trepan,library_trepan)
Allocated(scissors,library_scissors)	Allocated(scissors,library_scissors)
Allocated(forceps,library_forceps)	Allocated(forceps,library_forceps)
Allocated(stapler,library_stapler)	Allocated(stapler,library_stapler)
Position(x0)	Grasped(nothing)
Grasped(nothing)	Patient_status(skin,closed)
Patient_status(skin,closed)	Patient_status(skull,intact)
Patient_status(skull,intact)	Patient_status(tumour,absent)
Patient_status(tumour,present)	

Tabla 5-14. Estado inicial y final

Los operadores diseñados para el dominio de la cirugía se pueden ver en las Tablas 5-15, 5-16, 5-17 y 5-18.

Operador		
move_to(x,y)	Precondiciones:	Position_robot(x)
	Lista de borrados:	Position_robot(x)
	Lista de añadidos:	Position_robot(y)
take(instrument, library_instrument)	Precondiciones:	Position(library_instrument) Allocated(instrument, library_instrument) Grasped(nothing)
	Lista de borrados:	Grasped(nothing) Allocated(instrument, library_instrument)
	Lista de añadidos:	Grasped(instrument)

Tabla 5-15. Operadores quirúrgicos.

Operador		
go_up(x,y)	Precondiciones:	Position(x)
	Lista de borrados:	Position(x)
	Lista de añadidos:	Position(y)
go_down(x,y)	Precondiciones:	Position(x)
	Lista de borrados:	Position(x)
	Lista de añadidos:	Position(y)
put_down(instrument, library_instrument)	Precondiciones:	Position(library_instrument) Allocated(nothing,library_instrument) Grasped(instrument)
	Lista de borrados:	Allocated(nothing, library_instrument) Grasped(instrument)
	Lista de añadidos:	Allocated(instrument, library_instrument) Grasped(nothing)
Cut(x,y)	Precondiciones:	Position(x) Grasped(scalpel) Patient_status(skin,closed)
	Lista de borrados:	Position(x) Patient_status(skin, closed)
	Lista de añadidos:	Position(y) Patient_status(skin,cut) Patient_status(skull,intact) Patient_status(tumour,present)

Tabla 5-16. Operadores quirúrgicos. (cont.)

Operador		
anchor(x,y)	Precondiciones:	Position(x) Grasped(retractor) Patient_status(skin,cut)
	Lista de borrados:	Position(x) Patient_status(skin, cut)
	Lista de añadidos:	Position(y) Patient_status(skin,open) Patient_status(skull,intact) Patient_status(tumour,present)
disanchor_from_to(x,y)	Precondiciones:	Position(x) Grasped(retractor) Patient_status(skin,open)
	Lista de borrados:	Position(x) Patient_status(skin, open)
	Lista de añadidos:	Position(y) Patient_status(skin, cut)
close(x,y)	Precondiciones:	Position(x) Grasped(stapler) Patient_status(skin,cut)
	Lista de borrados:	Position(x) Patient_status(skin,cut)
	Lista de añadidos:	Position(y) Patient_status(skin,closed)

Tabla 5-17. Operadores quirúrgicos (cont.)

Operator		
perforate(x,y)	Precondiciones:	Position(x) Patient_status(skull,intact) Grasped(trepan) Patient_status(skin,open) Patient_status(tumour,present)
	Lista de borrados:	Position(x) Patient_status(skull, intact)
	Lista de añadidos:	Position(y) Patient_status(skull,hole)
separate(x)	Precondiciones:	Position(x) Grasped(scissors) Patient_status(tumour,present) Patient_status(skull, hole) Patient_status(skin, open)
	Lista de borrados:	Patient_status(tumour, present)
	Lista de añadidos:	Patient_status(tumour, detached)
grasping_tumour(tumour)	Precondiciones:	Patient_status(tumour,detached) Grasped(forceps)
	Lista de borrados:	Patient_status(tumour, detached)
	Lista de añadidos:	Patient_status(tumour, absent)

Tabla 5-18. Operadores quirúrgicos (cont.)

Se ha implementado un simulador robótico, usando el lenguaje de programación Java, para que el cirujano pueda chequear el movimiento del robot en un entorno virtual antes de la ejecución de la operación en el mundo real. La figura 5-11 muestra la interfaz gráfica del simulador.

Antes de realizar la operación quirúrgica simulada, el cirujano debe introducir algunos datos preliminares (Figura 5-11) como:

- El punto de inicio del robot;
- La posición de los instrumentos quirúrgicos;
- La posición del tumor, de acuerdo al diagnóstico basado en algunos test médicos realizados con anterioridad (como Tomografía Computerizada o Resonancias Magnéticas).

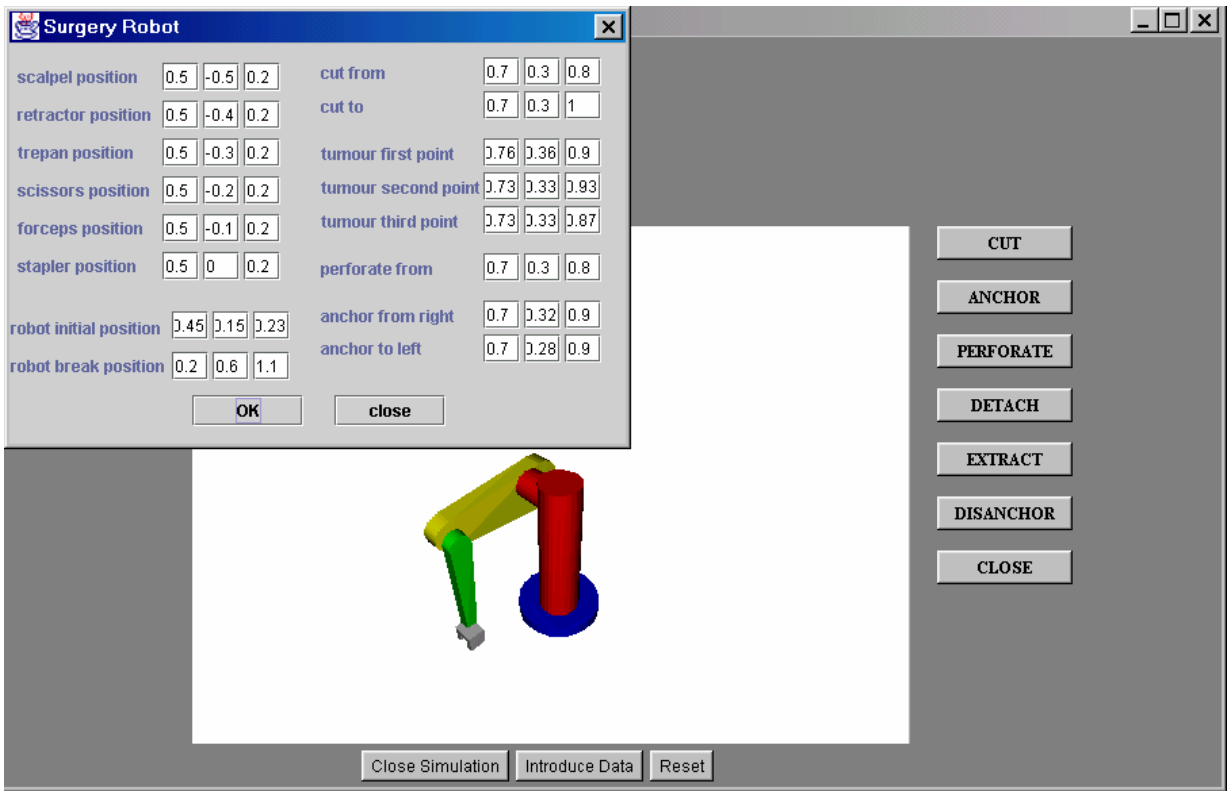

Figura 5-11. Inserción de parámetros antes de la operación

El manipulador robótico elegido para realizar la operación es un Puma 560 comercial, en que las características son bien conocidas. Por supuesto, las operaciones reales tendrían que ser ejecutadas en dispositivos estándar de robótica médica, para que tenga un nivel adecuado de seguridad.

5.4 RESUMEN

En este capítulo se ha evaluado la utilidad y aplicabilidad de la metodología para ontology learning presentada en el capítulo 3 en dos áreas principalmente. El primer área en el que se ha validado ha sido la construcción de ontologías del dominio para la Web Semántica. Dentro de esta área, se ha realizado un estudio en tres diferentes dominios, cáncer de mama, leucemia y cifrado de datos, con la finalidad de analizar cómo el sistema es capaz de aprender y sugerir conocimiento a través de las diferentes sesiones. Para demostrar esto, aunque sin pretenderlo hacer con rigor matemático (validez estadística), se ha definido una medida que hemos denominado *exactitud*, que es el resultado de la división entre la cantidad de entidades de conocimiento sugeridas por el sistema y aceptadas por el usuario experto y la cantidad total de entidades de conocimiento existentes en el texto. En base a esta medida, se han formulado tres hipótesis que se pretenden demostrar a través de los experimentos:

- Hipótesis 1 (Similaridad): “El sistema obtiene una exactitud similar para cada experto”.

- Hipótesis 2 (Utilidad): "El sistema es útil para la construcción de la ontología"

- Hipótesis 3 (Aprendizaje incremental): “El sistema aprende a lo largo del experimento”.

Para los experimentos, se proporcionó el sistema vacío a distintos individuos y los mismos textos a analizar, para comprobar cada una de las hipótesis nombradas anteriormente.

El otro área de aplicación está relacionada con el reconocimiento de órdenes en lenguaje natural, que ha sido aplicado a la robótica médica. Con esta finalidad, se ha realizado una pequeña ampliación en el modelo para su aplicación dentro del reconocimiento de órdenes en lenguaje natural a través del habla. Así, se ha realizado un trabajo que persigue el desarrollo de

un sistema para la realización de operaciones quirúrgicas que interactúa con el cirujano a través de comandos de voz (expresados en lenguaje natural).

Este sistema pretende traducir las órdenes del cirujano expresadas en lenguaje natural a una secuencia de pasos que debe ejecutar el robot para llevar a cabo esas órdenes. Para la planificación del brazo del robot se ha utilizado STRIPS.

CAPÍTULO 6

UNA APLICACIÓN SOFTWARE PARA ONTOLOGY LEARNING

6.1 INTRODUCCIÓN

El objetivo general de la aplicación software que se presenta en este capítulo es desarrollar un sistema que pueda servir como entorno para la creación de ontologías a partir de texto utilizando la metodología de ontology learning diseñada. Esta aplicación toma un texto como entrada, devolviendo al final un fichero XML con la ontología obtenida tras el procesamiento total de ese texto.

La herramienta software puede ser usada por aquellas personas de la comunidad de Adquisición de Conocimiento interesadas en representar el conocimiento contenido en textos por medio de ontologías y por aquellos que deseen compartir su conocimiento ontológico con el resto de usuarios.

6.1.1 Tipos de usuario

En el sistema pueden distinguirse dos tipos de usuarios que dan lugar a dos modos posibles de utilización de la herramienta:

- Modo experto

Estos usuarios tienen permitida la utilización de todas las funcionalidades disponibles del sistema relativas a la adquisición del conocimiento: modificar expresiones y relaciones ontológicas, añadir conceptos, atributos o valores a los atributos, y crear relaciones entre conceptos.

El modus operandi es como sigue. Los expertos introducen en el sistema el conocimiento que adquieren a través de las sesiones, como se ha explicado anteriormente en los capítulos 3 y 4. Este conocimiento es entonces utilizado por el sistema para inferir, si es el caso, nuevas entidades de conocimiento, principalmente conceptos y relaciones entre los conceptos. Los

resultados de la inferencia del conocimiento son mostrados al usuario para que éste pueda incluirlos dentro de la ontología que pretende obtener a partir del texto.

- Modo usuario normal

Este modo de trabajo ha sido concebido para los usuarios que pretenden construir una ontología automáticamente a partir de un texto, así como del conocimiento, proveniente de otros expertos, que contiene la herramienta. Así, estos usuarios no pueden entrenar el sistema ni interactuar de manera supervisada para modificar la ontología mientras se procesa. Solamente tienen la posibilidad de obtener una ontología inferida a partir del texto que introducen en el sistema.

6.1.2 Idiomas de la aplicación

Como se ha comentado a lo largo de este trabajo, el sistema que se ha propuesto es independiente del idioma, así como los módulos que utiliza. El POS-Tagger eTiKeT@ utilizado permite categorizar gramaticalmente un texto en cualquier idioma que contenga el alfabeto occidental, pudiendo, asimismo, configurar las categorías gramaticales más relevantes en cada idioma.

Además, el sistema de inferencia está basado en un sistema de reglas que puede funcionar con cualquier idioma, debido a que las reglas se forman en base al texto que se analice, y, por tanto, es sensible al idioma del texto.

Más concretamente, se ha personalizado la aplicación para dos idiomas principalmente castellano e inglés.

6.2 UNA SESIÓN CON KATEXT

En este apartado se muestran las facilidades para la adquisición de ontologías a partir de texto a través de una aplicación software que ha sido implementada y que se basa en la metodología introducida en los capítulos anteriores. En primer lugar, se describen las distintas opciones relacionadas con usuarios.

6.2.1 Gestión de Usuarios

En este apartado se describe cómo los usuarios se pueden dar de alta en el sistema, o bien acceder al mismo. La Figura 6-1 es la ventana principal de la aplicación. Al hacer clic en el menú Usuario, los usuarios pueden realizar tres operaciones:

- *Nuevo usuario*: Un nuevo usuario quiere darse de alta

- *Editar usuario*: Para editar y modificar los datos del usuario actual

Figura 6-1 Ventana principal de la aplicación

6.2.1.1 Alta de nuevos usuarios

Supongamos que un nuevo usuario quiere registrarse. En ese caso aparecerá un cuadro de diálogo (ver Figura 6-2). El nuevo usuario debe introducir algunos datos para ser un usuario de este sistema. Tal información incluye detalles personales y profesionales, como (1) el nombre de usuario; (2) los apellidos; (3) la profesión; (4) el correo electrónico y (5) la contraseña. Se ha dicho previamente que existen tres tipos de usuarios, a saber, los usuarios normales, usuarios expertos, y usuarios técnicos para administrar la herramienta. Por defecto, cada nuevo usuario es un usuario normal.

Figura 6-2 Diálogo de registro

Una vez que se ha registrado el nuevo usuario en el sistema vuelve a aparecer la ventana principal (Figura 6-1), por lo que el usuario puede acceder al sistema. Para ello, el usuario necesita introducir su nombre de usuario y contraseña en el correspondiente cuadro de diálogo (Figura 6-3).

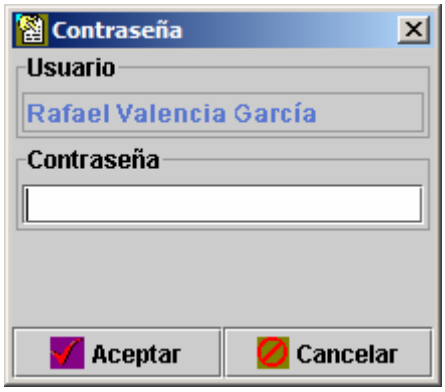

Figura 6-3 Diálogo de acceso al sistema

6.2.1.2 Modificar o eliminar un usuario existente

Supongamos que un usuario quiere modificar sus datos o bien eliminarse como usuario del sistema, seleccionando la opción "Editar Usuario" del menú "Usuario". En este caso nos aparecerá un diálogo donde se podrán editar los datos introducidos anteriormente, así como cambiar el tipo de usuario (Figura 6-4).

Figura 6-4 Diálogo de edición de usuarios

Como se puede observar, se ofrecen dos opciones en este diálogo: “Modificar el usuario” o “Eliminar el usuario”. Si queremos modificar el usuario, deberemos cambiar los datos incorrectos a los correctos, introduciéndolos de nuevo en los campos correspondientes, y una vez corregidos los datos, pulsamos el botón *Usuario Modificado*, y el usuario quedará modificado en el sistema.

Si, por el contrario, queremos eliminar un usuario, utilizaremos el botón *Eliminar usuario* y el usuario será eliminado del sistema.

6.2.2 Gestión de Dominios

Una vez se haya dado de alta al usuario experto, deberemos dar de alta el dominio y la tarea de la que trata el texto si es que ya no se ha dado de alta anteriormente. Dentro del menú “Dominio” del menú principal tenemos la opción “Editar Dominio”. En este caso aparecerá un diálogo en el cual podremos insertar dominios y tareas dentro de cada uno de los dominios (ver figura 6-5).

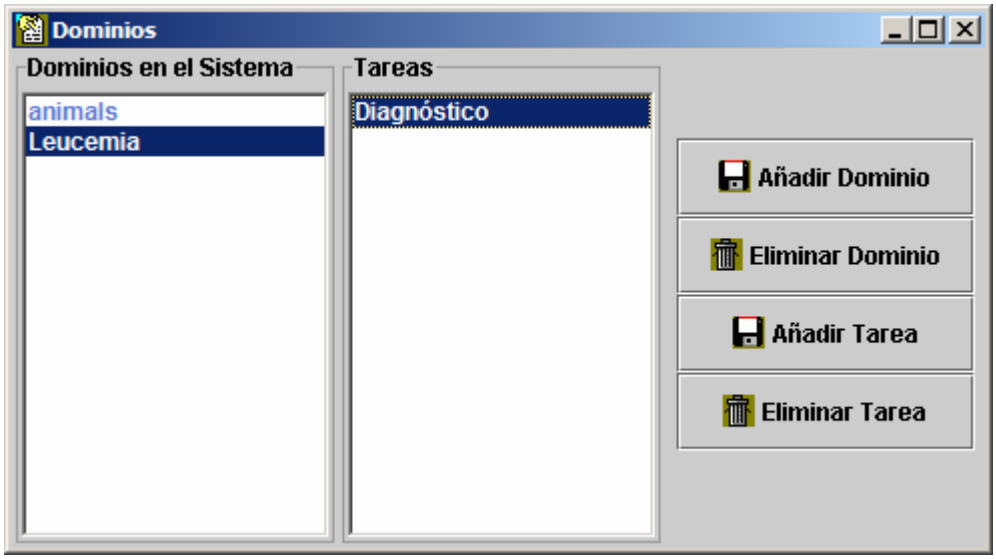

Figura 6-5 Diálogo de edición de dominios

Como podemos observar en la figura 6-5, aparece un listado de los dominios que existen en el sistema, y a su lado, las tareas relacionadas con el dominio que esté seleccionado, de entre los que se han introducido en el sistema. También podemos observar que, mediante este diálogo, se pueden añadir y eliminar dominios y tareas, como se explica a continuación.

6.2.2.1 Altas de nuevos dominios y tareas

Para añadir nuevos dominios de aplicación, deberemos pulsar el botón *Añadir Dominio*, donde aparecerá un diálogo que nos solicitará el nombre del nuevo dominio. Una vez se inserte el dominio, aparecerá en la lista de dominios que contiene el sistema.

Para añadir tareas dentro de un dominio, la forma de actuar será similar, pero previamente se debe seleccionar el dominio al que va a pertenecer esa tarea, como se muestra en la figura 6-5. Luego pulsaremos el botón *Añadir Tarea*, y aparecerá un diálogo donde deberemos insertar el nombre de la tarea.

6.2.2.2 Eliminación de dominios y tareas

Para eliminar un dominio del sistema, debemos seleccionar el dominio en el recuadro *Dominios en el sistema*, donde aparecen los dominios, y pulsamos a continuación el botón *Eliminar Dominio*. Se pedirá una confirmación de la eliminación del dominio, y una vez confirmado, se eliminará el dominio del sistema.

Para eliminar una tarea concreta de un dominio del sistema, seleccionaremos el dominio al que le vamos a eliminar la tarea (en el recuadro *Dominios en el Sistema*), y seguidamente, seleccionaremos la tarea a eliminar en el recuadro *Tareas*. Una vez esté seleccionada la tarea que queremos eliminar, pulsaremos el botón *Eliminar Tarea*. Aquí también se pedirá una confirmación antes de eliminar la tarea.

6.2.3. Gestión de tipos de relaciones

El sistema KAText permite el uso de diversos tipos de relaciones predeterminadas, además de la inserción de nuevas relaciones que desee utilizar el usuario. Las relaciones predeterminadas, junto con sus propiedades, se pueden observar en la tabla 6-1 tal como aparece en Ruiz-Sánchez *et al* (2003).

RELACIÓN	REFLEXIVA	SIMÉTRICA	TRANSITIVA
Equivalencia	X	X	X
Taxonómica			X
Dependencia	X		X
Topológica	X	X	X
Causal			
Funcional			X
Temporal			X
Condicional			X
Propósito			X
Conostituyente/Objeto			
Componente/Objeto			X
Miembro/Colección			
Porción/Masa			
Característica/Actividad			X
Lugar/Área			X
Fase/Proceso			X

Tabla 6-1. Tipo de relaciones predeterminadas en el sistema. Extraída de (Ruiz-Sánchez *et al*, 2003)

Para gestionar los tipos de relaciones con los que trabajar, se debe acceder a la opción "Editar tipo de relación" dentro del menú "Tipo de relación", tras lo cual aparecerá un diálogo con todos los tipos de relaciones que contiene el sistema (Figura 6-6).

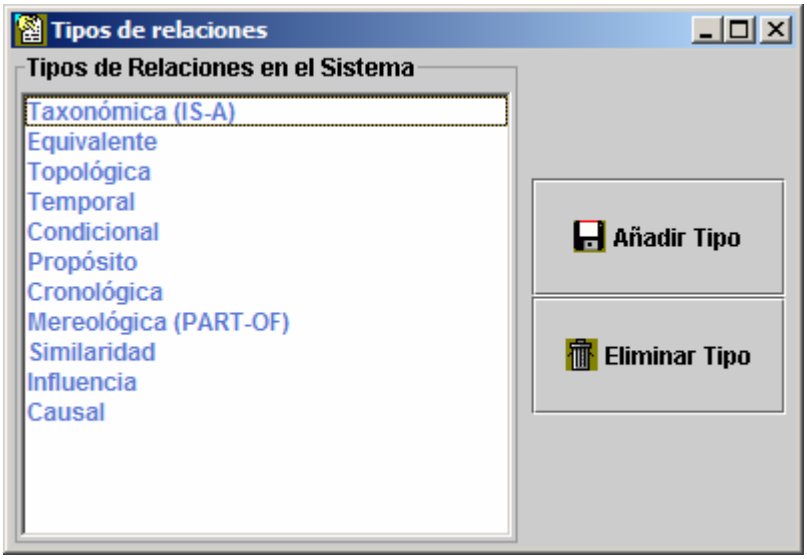

Figura 6-6. Diálogo de gestión los tipos de relaciones.

Como podemos observar, en el recuadro llamado *Tipos de Relaciones en el Sistema* aparecen los tipos de relaciones que tiene registrados el sistema. Mediante ese diálogo podemos añadir y eliminar tipos de relaciones, como se explica a continuación.

6.2.3.1 Crear un nuevo tipo de relación

Para añadir nuevos tipos de relaciones en el sistema, deberemos pulsar el botón *Añadir Tipo*, donde aparecerá un diálogo que nos solicitará el nombre del tipo de relación. Una vez se inserte el tipo de relación, aparecerá en la lista de tipos de relaciones posibles que contiene el sistema.

6.2.3.2 Eliminar un tipo de relación existente

Para eliminar un tipo de relación del sistema, debemos seleccionar el tipo de relación en el recuadro *Tipos de relaciones del sistema*, y pulsamos el botón *Eliminar Tipo*. Se pedirá una

confirmación de la eliminación del tipo de relación, y una vez confirmado, se eliminará el tipo de relación del sistema.

6.2.4. CREAR SESIONES

El proceso de ontology learning del sistema se realiza en base a sesiones. Una sesión se compone de un único texto procesado por un experto determinado. Además, este texto se incluye dentro de una tarea y un dominio determinados. El experto entonces irá procesando el texto, indicándole al sistema el conocimiento existente en ese texto, con lo que el sistema se irá realimentando con nueva información tal y como hemos visto en los ejemplos del capítulo 4.

6.2.4.1. Crear una nueva sesión

Para crear una nueva sesión utilizamos la opción "Nueva Sesión" dentro del menú "Sesión". Nos aparecerá entonces el diálogo que muestra la figura 6-7.

Figura 6-7. Diálogo para la creación de nuevas sesiones.

Como podemos observar en la figura 6-7 en este diálogo se selecciona el usuario que va a realizar la extracción de conocimiento (en este caso Rafael Valencia García), el dominio y la tarea asociadas al texto del que se va a extraer conocimiento (Leucemia y Diagnóstico en nuestro caso). Además, la ventana contiene un campo para seleccionar la ruta del fichero de texto que se quiere someter al proceso de extracción de conocimiento desde texto.

Una vez se hayan seleccionado cada campo y el fichero donde se pretende realizar el proceso de extracción de conocimiento, pulsaremos el botón *OK* y la sesión quedará creada y seleccionada, apareciendo en la ventana principal el texto con el que se va a trabajar (Figura 6-8).

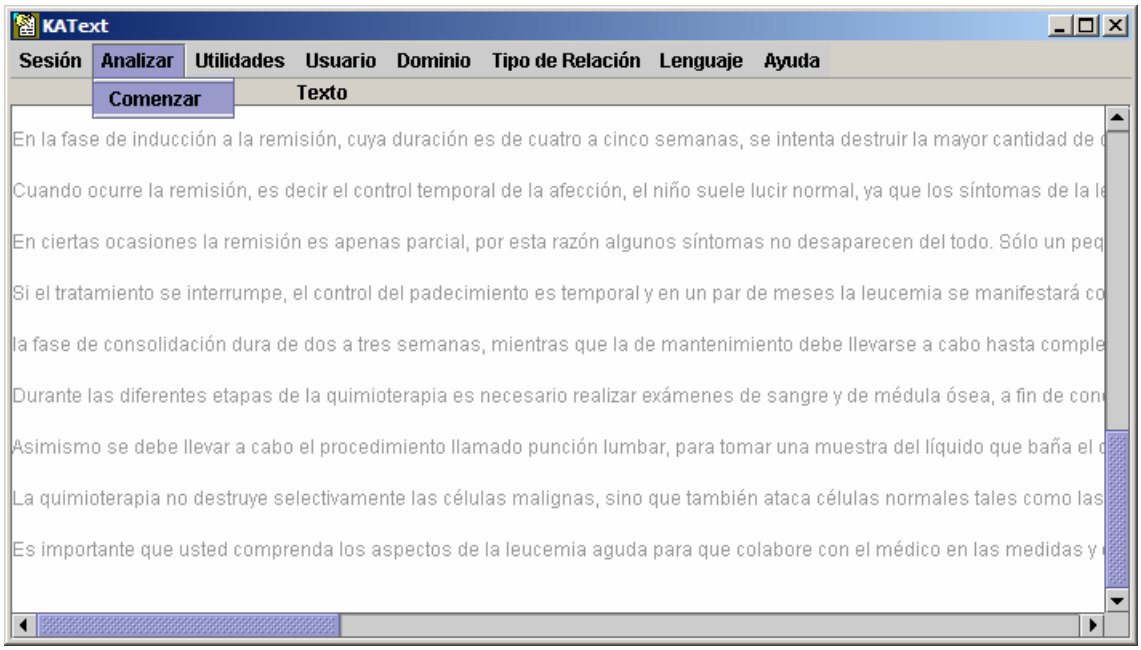

Figura 6-8. Ventana principal después de seleccionar una nueva sesión.

A partir de aquí, el sistema está preparado para empezar a procesar el texto en busca de entidades de conocimiento que den lugar a una ontología del dominio con el conocimiento que contiene el texto. En el siguiente apartado se explica cómo se interactúa con la herramienta introduciendo conocimiento y validando el conocimiento inferido por el sistema.

6.2.5. Análisis, inferencia y aprendizaje

Una vez que tenemos una sesión cargada en la herramienta, ya sea nueva o recuperada, para comenzar con el análisis (o continuarlo por la frase por la que se quedó analizando la última vez), seleccionamos la opción "Comenzar" del menú "Analizar", como se muestra en la figura 6-8.

Una vez se elija esa opción se empezará el análisis del texto frase a frase, donde el experto tendrá que identificar el conocimiento que se pueda extraer de cada frase utilizando los diálogos de inferencia que se muestran en la figura 6-9.

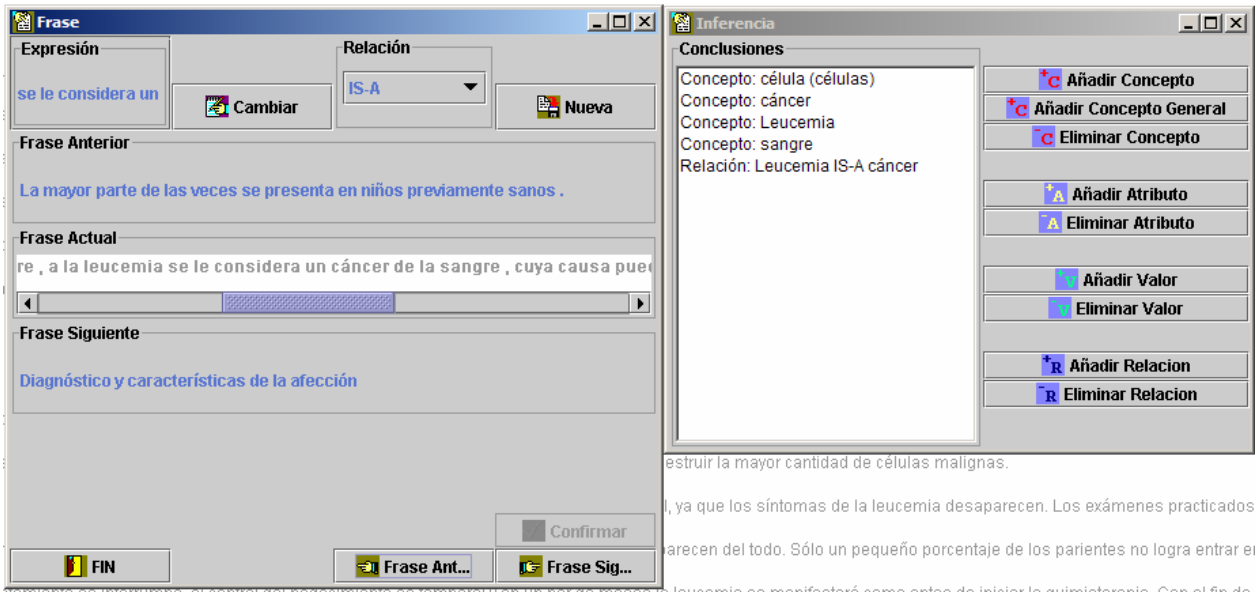

Figura 6-9. Diálogos de inferencia

Para cada una de las frases que se van analizando, el sistema muestra el conocimiento que va infiriendo mediante estos dos diálogos (Figura 6-9). En el diálogo llamado *Frase* se muestran los datos de la frase actual, la frase anterior, la frase posterior, la expresión reconocida como expresión verbal (si se ha encontrado alguna en la frase) y la relación ontológica asociada a esa expresión (si la hay). El diálogo llamado *Inferencia* muestra las entidades de conocimiento (conceptos, atributos, valores y relaciones) que contiene la frase actual. Es en este diálogo donde aparecerán las entidades de conocimiento que va infiriendo el sistema y el usuario tendrá que modificar esas entidades de conocimiento o bien insertar alguna adicional.

El diálogo *Frase* de la figura 6-9 tiene seleccionada que la expresión verbal principal de la frase actual es "se le considera un" y representa a un tipo de relación IS-A entre leucemia y cáncer. Aparecen también la frase anterior, que ya ha sido procesada, y la siguiente a ser procesada para que se pueda mostrar el contexto en el que se encuentra la frase actual.

El diálogo *Inferencia* de la figura 6-9 contiene cinco entidades de conocimiento, cuatro conceptos y una relación. Estas entidades de conocimiento han sido inferidas, o bien han sido identificadas por el experto, como entidades de conocimiento relevantes en ese dominio. Las entidades de conocimiento que se encuentran en este diálogo se muestran a continuación:

Concepto: célula (células)

Concepto: cáncer

Concepto: Leucemia

Concepto: sangre

Relación: Leucemia IS-A cáncer

Podemos observar que el identificador (nombre) del primer concepto, *célula*, va seguido entre paréntesis por la expresión lingüística de cómo aparece en el texto actual "células". Los demás conceptos aparecen tal cual están en el texto y representan el concepto con el mismo nombre.

A continuación explicamos cómo se cambia en el sistema la expresión verbal y el tipo de relación en el diálogo *Frase*, y cómo se introducen y eliminan las entidades de conocimiento del diálogo *Inferencia*.

6.2.5.1 Cambio de la expresión verbal

El sistema, tal y como se ha explicado en el apartado 3 y 4 de este trabajo, busca la expresión verbal principal de cada frase mediante los criterios similar y aceptable definidos en apartados anteriores. Si no encuentra ninguna expresión verbal en la base de conocimiento de relaciones, entonces seleccionará como expresión candidata el verbo que considere principal, al que no se le asociará ninguna relación ontológica.

Si queremos corregir la expresión verbal inferida, pulsaremos el botón *Cambiar* que aparece en el diálogo *Frase* de la figura 6-9. Una vez que pulsemos este botón, nos aparecerá un diálogo solicitando la expresión verbal correcta dentro de la frase.

El sistema comprueba si la expresión introducida realmente existe en la frase. Si no existe esta expresión, no dejará cambiar la expresión verbal actual. En el caso en el que exista, el sistema modificará la expresión verbal actual y mostrará un diálogo como el de la figura 6-10.

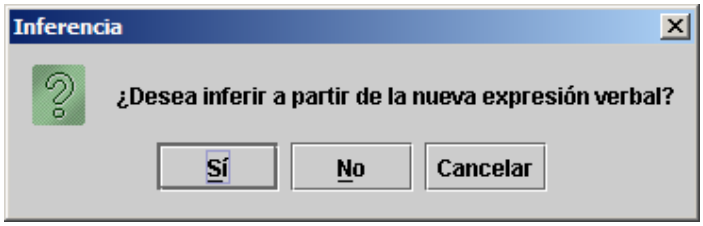

Figura 6-10. Diálogo para mantener las conclusiones actuales

El sistema pregunta si queremos que intente inferir alguna entidad de conocimiento a partir de este cambio, como podía suceder en los ejemplos mostrados en el capítulo 4. En el caso de pulsar el botón *Sí*, el sistema tratará de inferir entidades de conocimiento a partir de la nueva expresión verbal, manteniendo las entidades de conocimiento introducidas por el experto hasta ese momento. Si pulsamos el botón *No*, el sistema no tratará de realizar ninguna inferencia con los nuevos datos.

6.2.5.2 Cambio del tipo de relación

Si el sistema ha inferido también mal la relación ontológica que está asociada a la expresión verbal deberemos modificarla. Para ello, sólo deberemos seleccionar en el cuadro *Relación* del diálogo *Frase* la relación correcta de entre las posibles del sistema. Una vez modificado el tipo de relación que representa la expresión verbal, nos aparecerá el diálogo de la figura 6-10 para ver si queremos que el sistema intente inferir nuevo conocimiento a partir del nuevo cambio.

A continuación mostramos las operaciones que se pueden realizar en el diálogo de *Inferencia*, esto es, añadir y eliminar entidades de conocimiento.

6.2.5.3 Añadir un nuevo concepto

Para añadir un nuevo concepto, se utiliza el botón *Añadir Concepto* del diálogo *Inferencia*. Nos aparece un diálogo donde introduciremos la expresión lingüística que representa este nuevo concepto y el nombre del concepto en cuestión.

En el caso de que el concepto que represente sea distinto de la expresión lingüística, aparecerá en el diálogo de *Inferencia* el identificador del concepto en cuestión y seguido entre paréntesis, la expresión lingüística en la frase actual que representa ese concepto. Como ejemplo, podemos ver el concepto *célula* en la figura 6-9.

6.2.5.4 Añadir una nueva relación

Cuando pulsamos el botón *Añadir Relación* del diálogo *Inferencia*, aparecerá un diálogo, en el que debemos seleccionar los conceptos que participan en la nueva relación. Los conceptos participantes en la relación deben introducirse previamente. En la figura 6-11 podemos ver el diálogo para añadir la relación "Leucemia IS-A cáncer".

Figura 6-11. Diálogo para añadir una nueva relación ontológica.

Una vez se añada la relación pulsando botón *Aceptar*, la relación aparecerá en el cuadro *Conclusiones* del diálogo de *Inferencia*, tal y como se muestra en la figura 6-9.

6.2.5.5 Añadir un atributo a un concepto

Para añadir un atributo a un concepto, hay que seleccionar dicho concepto en el recuadro *Conclusiones* de la ventana *Inferencia*, y a continuación pulsar el botón *Añadir atributo*.

Nos aparece un diálogo en el que se solicita la expresión lingüística que representa ese atributo. También se puede introducir conocimiento de atributos que no aparecen explícitamente, ya que puede suceder que en una frase aparezca un concepto y el valor de un atributo relacionado con el concepto, donde el atributo no se encuentra explícito en el texto (Valencia-García *et al*, 2004b). Por ejemplo en la frase siguiente:

"A tight mutant is a mutant which displays its non-wild phenotype distinctly"

Aquí *phenotype* representa un concepto y *non-wild* es un valor del atributo *purity* del concepto *phenotype*.

Una vez se introduzca el atributo, aparecerá en el diálogo de *Inferencia*, representado por expresión *concepto.atributo*. Por ejemplo, el atributo *purity* del concepto *phenotype* se mostrará como "*phenotype.type*".

6.2.5.6 Añadir un valor a un atributo

Para añadir un valor a un atributo debemos seleccionar dicho atributo en el recuadro *Conclusiones* de la ventana *Inferencia*, y a continuación pulsar el botón *Añadir valor*. Nos aparece, entonces, un diálogo en el que se solicita la expresión lingüística que representa ese valor.

Una vez se introduce el valor, éste aparece en el diálogo de *Inferencia*, representado por la expresión *concepto.atributo=valor*. Así, por ejemplo, el valor *non-wild* del atributo *purity* se mostrará como "*phenotype.purity=non-wild*"

6.2.5.7 Eliminar conceptos, atributos, valores o relaciones

Podemos eliminar cualquier entidad de conocimiento que haya sido inferida erróneamente, así como si el usuario se ha equivocado al introducirla.

Para eliminar un concepto, primero debemos seleccionarlo dentro del recuadro *Conclusiones* y después utilizamos el botón *Borrar Concepto* situado en el diálogo de *Inferencia*. Aparecerá entonces, un diálogo de confirmación de eliminación del concepto. Si se confirma la eliminación, se eliminarán el concepto, los atributos asociados a ese concepto y los valores asociados a los atributos del concepto.

Si lo que queremos es eliminar un atributo de un concepto, lo debemos seleccionar previamente en el recuadro *Conclusiones* y, después, pulsamos el botón *Borrar atributo*. Automáticamente, se borrarán los valores que ese atributo tuviese asociados, previa solicitud de confirmación de la eliminación del atributo.

También es posible eliminar un valor de un atributo, seleccionando ese valor en el recuadro *Conclusiones* y pulsando el botón *Borrar Valor*.

Por último, también podemos eliminar relaciones de la misma manera que las otras entidades de conocimiento, es decir, seleccionándola en el recuadro *Conclusiones* y pulsando el botón *Borrar Relación*.

6.2.5.8 Confirmar la frase actual y pasar a la siguiente

Una vez que hemos terminado de extraer las entidades de conocimiento contenidas en la frase actual, debemos confirmarla mediante el botón *Confirmar* situado en la ventana *Relación*. Una vez confirmada ésta, quedará guardado el estado de la sesión en una base de datos para poder continuar, en otro momento, la extracción de conocimiento en el texto.

Una vez confirmada la frase actual, ya podemos avanzar a la frase siguiente mediante el botón *Frase Siguiente (*Figura 6-12).

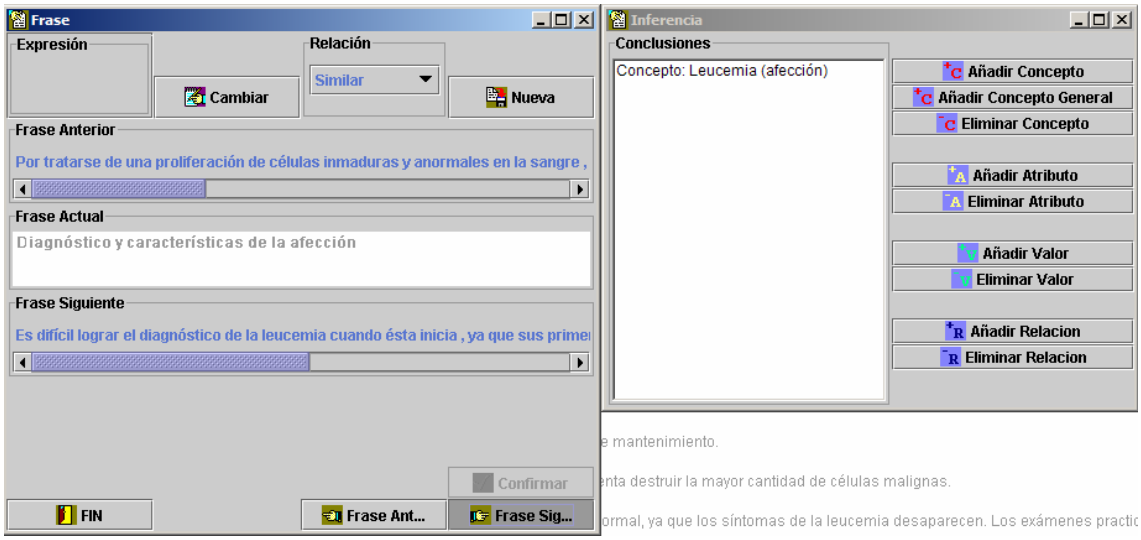

Figura 6-12. Avanzando a la frase siguiente en el análisis

También podemos retroceder e ir a la frase anterior analizada, en caso de que la hubiese, mediante el botón *Frase Anterior,* y de este modo, podríamos ver todo el conocimiento que contienen las frases anteriores.

6.2.6 Almacenar una sesión

Para poder continuar trabajando en otro momento con una sesión determinada, debemos almacenarla en el sistema. Para ello se utiliza la opción *Guardar Sesión* o *Guardar Sesión Como* del menú *Sesión*, dependiendo de si queremos sobrescribir la sesión actual o guardarla con un nuevo identificador (de modo que se mantengan la antigua y la actual con las modificaciones).

Al guardar una sesión, se crea un fichero XML de un lenguaje de formalización de ontologías utilizado por el grupo de investigación Tecnologías del Conocimiento y Modelado Cognitivo (TECNOMOD) de la Universidad de Murcia.

A continuación se presenta el DTD del lenguaje XML. Dicha definición de DTD incluye todos los aspectos cubiertos por el modelo ontológico y por el lenguaje anterior, esto es, conceptos, relaciones y atributos.

```
<!ELEMENT ontology (concept+, relation*, axioms?)>
<!ATTLIST ontology
        domain CDATA #REQUIRED
        author CDATA #IMPLIED
        comment CDATA #IMPLIED
>
<!ELEMENT concept (alternative-names*, specific-attributes*, inheritattributes*)>
<!ATTLIST concept name CDATA #REQUIRED>
<!ELEMENT alternative-names (name+)>
<!ELEMENT specific-attributes (attribute+)>
<!ELEMENT inherit-attributes (attribute+)>
<!ELEMENT atrribute (value*)>
<!ATTLIST attribute
        name CDATA #REQUIRED
        comment CDATA #IMPLIED
        type CDATA #REQUIRED
>
<!ELEMENT value ( intervalo | #PCDATA)>
<!ELEMENT intervalo (from, to) >
<!ELEMENT from (#PCDATA) >
<!ELEMENT to (#PCDATA) >
```

```
<!ELEMENT relation (relation_type, concept_name, specialization_list?)>

<!ELEMENT relation_type (property*)>

<!ATTLIST relation_type name CDATA #REQUIRED>

<!ELEMENT specialization_list (attribute+)>

<!ELEMENT axioms axioma*>

<!ELEMENT axioma (#PCDATA)>

<!ATTLIST axioma type CDATA #REQUIRED>

<!ELEMENT concept_name (#PCDATA)>

<!ELEMENT property (#PCDATA)>

<!ELEMENT name (#PCDATA)>
```

A continuación mostramos como queda representada la ontología de la figura 6-15, representada con este lenguaje:

```
<?xml version='1.0' encoding='ISO-8859-1'?>
<?xml-stylesheet type='text/xsl' href='ontology.xsl'>
<!DOCTYPE ontology SYSTEM 'ontology.dtd'?>

<ontology domain="Leucemia" author="Rafael Valencia García" comment="">

<concept name="Cáncer" comment="" category="Object">
<specific-attributes>
<attribute name="tipo" comment="" type="STRING"></attribute>
</specific-attributes>
</concept>

<concept name="Leucemia" comment="" category="Object">
</concept>

<concept name="Leucemia linfocítifa aguda" comment="" category="Object">
</concept>

<concept name="Leucemia mielógena crónica" comment="" category="Object">
</concept>

<concept name="Leucemia mielógena aguda" comment="" category="Object">
</concept>
```

<concept name="Leucemia mielógena" comment="" category="Object">
<specific-attributes>
<attribute name="Intensidad" comment="" type="STRING"></attribute>
</specific-attributes>
</concept>

<relation name="TAXONOMIC">
<concept_name>Leucemia linfocítifa aguda</concept_name>
<concept_name>Leucemia</concept_name>

<relation_type>
<property>NonSymmetry</property>
</relation_type>

<specialization_list>
<attribute name="tipo" comment="" type="STRING"></attribute>
</specialization_list>
</relation>

<relation name="TAXONOMIC">
<concept_name>Leucemia</concept_name>

<concept_name>Cáncer</concept_name>

<relation_type>
<property>NonSymmetry</property>
</relation_type>

<specialization_list>
<attribute name="tipo" comment="" type="STRING"></attribute>
</specialization_list>
</relation>

<relation name="TAXONOMIC">
<concept_name>Leucemia mielógena crónica</concept_name>

<concept_name>Leucemia mielógena</concept_name>

<relation_type>
<property>NonSymmetry</property>
</relation_type>

<specialization_list>
<attribute name="Intensidad" comment="" type="STRING"></attribute>
</specialization_list>
</relation>

```
<relation name="TAXONOMIC">
<concept_name>Leucemia mielógena</concept_name>

<concept_name>Leucemia</concept_name>

<relation_type>
<property>NonSymmetry</property>
</relation_type>

<specialization_list>
<attribute name="tipo" comment="" type="STRING"></attribute>
</specialization_list>
</relation>

<relation name="TAXONOMIC">
<concept_name>Leucemia mielógena aguda</concept_name>

<concept_name>Leucemia mielógena</concept_name>

<relation_type>
<property>NonSymmetry</property>
</relation_type>

<specialization_list>
<attribute name="Intensidad" comment="" type="STRING"></attribute>
</specialization_list>
</relation>
</ontology>
```

Actualmente el grupo está desarrollando aplicaciones para la importación/exportación de/a ontologías representadas mediante OWL.

6.2.7 Recuperar una sesión

Para recuperar una sesión almacenada en el sistema, se debe usar el menú *Sesión*, opción *Abrir Sesión*. Nos aparece, entonces la ventana que mostramos en la figura 6-13, en la que debemos seleccionar la sesión a recuperar y pulsar, a continuación, el botón *Abrir Sesión*.

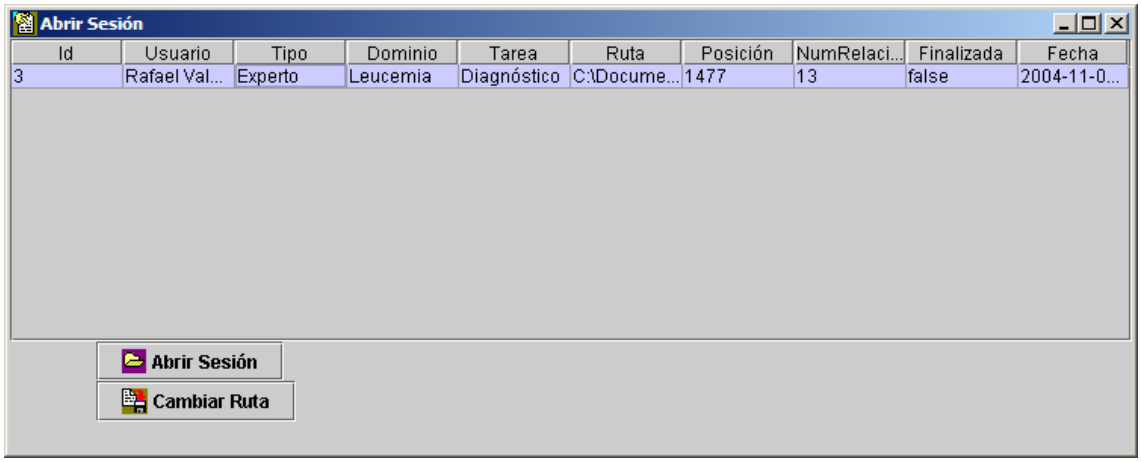

Figura 6-13. Recuperando sesiones.

Como se puede ver en la figura 6-13, la información que se muestra para cada sesión, en orden desde las columnas de izquierda a derecha es la siguiente:

- Identificador de sesión
- Nombre de usuario
- Tipo de usuario
- Dominio
- Tarea
- Ruta del fichero de texto
- Posición en el fichero de texto por la que se va analizando
- Número de frases analizadas
- Indicador de si se han confirmado todas las frases de la sesión
- Fecha en la que se grabó la sesión por última vez

Una vez que pulsemos el botón *Abrir Sesión*, se solicita que se introduzca la clave del experto en el sistema y, una vez verificada, se carga la sesión seleccionada y mostrando el texto

de la sesión en la ventana principal de la aplicación. Para continuar el análisis por donde lo dejamos la última vez, basta con pulsar la opción *"Comenzar"* del menú "Analizar".

Además, en este diálogo podemos cambiar la ruta de acceso hacia el fichero de texto con el que se trabaja en una sesión. Para ello, basta con seleccionar la sesión en el diálogo y usar el botón *Cambiar Ruta*. Nos aparece entonces, una ventana para seleccionar el fichero de texto en su nueva localización.

6.2.8 Eliminar una sesión

Para eliminar una sesión almacenada en el sistema, se ha de emplear la opción "Eliminar Sesión" del menú "Sesión".

Nos aparece un diálogo prácticamente igual al utilizado para abrir una sesión, con la diferencia de que ahora hay un botón, *Eliminar Sesión*, que sirve para eliminar la sesión que hayamos seleccionado (figura 6-14).

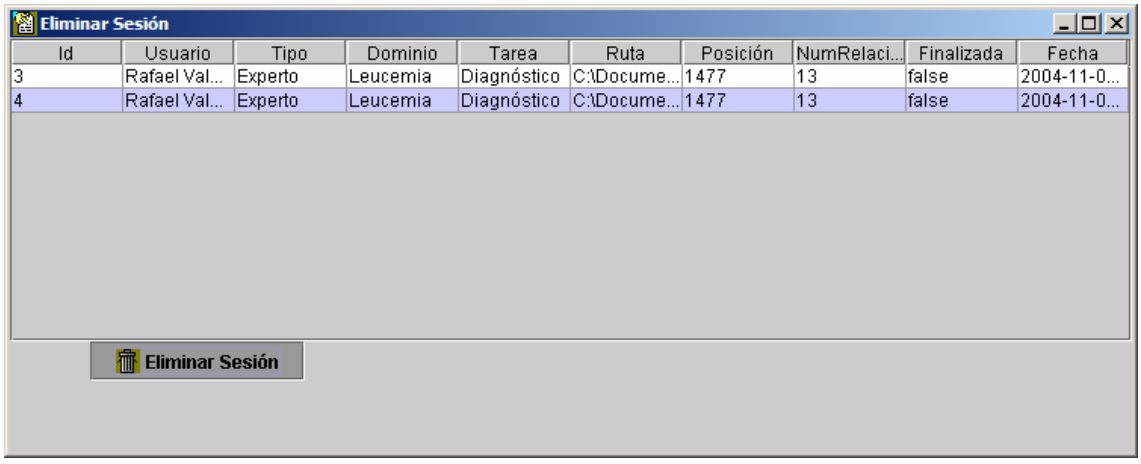

Figura 6-14. Diálogo para eliminar una sesión

Cuando estemos seguros de que hemos seleccionado la sesión que realmente queremos eliminar, pulsamos el botón *Eliminar Sesión* y automáticamente la sesión se eliminará del sistema, previa confirmación e identificación del usuario.

6.2.9 Visualización y edición de la ontología.

El sistema dispone de una opción dentro del menú "Utilidades", llamada "Editar Ontología", para la edición gráfica de la ontología extraída hasta ese momento, de modo que ésta puede completarse con nuevas entidades de conocimiento. Para ello, utilizamos un editor de ontologías desarrollado dentro del grupo de investigación (Vivancos-Vicente *et al*, 2004) (Figura 6-15).

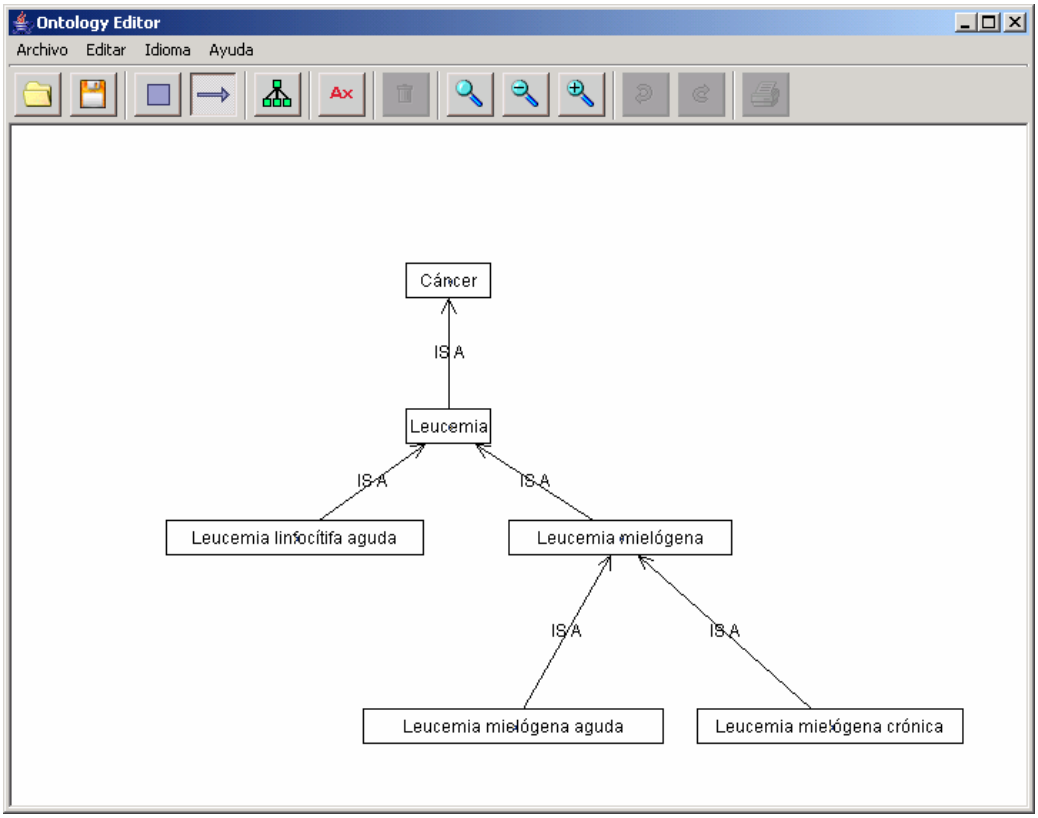

Figura 6-15. Editor de ontologías.

6.2.10 Uso de la herramienta de manera automática.

Hasta ahora hemos visto como se puede utilizar la herramienta con un usuario experto que va construyendo la ontología de manera semi-supervisada con el sistema, el cual va a ir sugiriendo conocimiento.

Este sistema se puede utilizar de manera automática para que obtenga una ontología a partir del procesamiento del texto. Esta ontología estará compuesta por todas las sugerencias de conocimiento obtenidas a lo largo del texto. Así, el sistema obtendrá un fichero XML que podrá ser editado (mediante el editor de ontologías mencionado en el apartado anterior, figura 6-15), para completar y refinar la ontología, así como para eliminar el conocimiento que se considere no válido.

6.3 COMPARACIÓN CON OTRAS HERRAMIENTAS DE ONTOLOGY LEARNING

En este apartado, se compara la herramienta presentada en este capítulo con algunos de los sistemas actuales para ontology learning basándonos en dos criterios: las relaciones ontológicas soportadas y si el proceso es automático o semi-supervisado.

6.3.1 Relaciones ontológicas soportadas.

KAText tiene un conjunto de relaciones predefinidas (tabla 6-1), al que se le puede añadir cualquier tipo de relación. Una vez añadidos los nuevos tipos de relaciones que se consideren necesarios, a lo largo de las sesiones el sistema aprenderá a identificar esas nuevas relaciones en base al conocimiento capturado por el sistema en estas sesiones.

Sin embargo, la mayoría de los sistemas de ontology learning (Omelayenko, 2001; Maedche & Staab, 2001) se centran principalmente en la extracción de jerarquías taxonómicas (Hahn & Schnattinger, 1998b; Suryanto & Compton, 2000; Heyer *et al*, 2001; Sundblad, 2002) o un conjunto reducido de relaciones no exclusivamente taxonómicas. Por ejemplo, en (Navigli *et al,* 2003; Navigli & Velardi, 2004) se presenta el sistema OntoLearn que identifica principalmente relaciones taxonómicas, de similaridad, y partonómica.

6.3.2 Procesos automáticos y semi-supervisados.

En este trabajo se ha presentado una metodología para la construcción de ontologías de manera semiautomática, de forma que una vez el sistema esté entrenado, dicha construcción se puede realizar de manera automática.

Algunos trabajos de ontology learning como Mikrokosmos (Nierenburg *et al*, 1996) y Cyc (Lenat & Guha, 1990) adquieren conocimiento ontológico manualmente, y después construyen una ontología completa, permitiendo adquirir más conocimiento de una forma semiautomática a partir de dicha ontología.

Según (Omelayenko, 2001; Maedche & Staab, 2001), la mayoría de las técnicas de Ontology Learning son semi-automáticas y necesitan de un ingeniero de conocimiento que guíe el proceso de adquisición de conocimiento, aunque también existen técnicas totalmente automáticas, como la presentada en (Shamsfard & Barforoush, 2004).

6.4 RESUMEN

En este capítulo se ha presentado una herramienta para la extracción de ontologías a partir de texto (ontology learning). Dicha herramienta cumple los requisitos del modelo de ontology learning propuesto en los capítulos anteriores. A continuación resumimos las propiedades de la herramienta de ontology learning presentada en este capítulo:

- **Usuarios**: La herramienta contempla dos tipos de usuarios: Expertos y Normales. Los usuarios expertos pueden hacer uso de toda la funcionalidad disponible en el sistema, como la construcción semisupervisada de la ontología y la opción de entrenar el sistema a través de las sesiones. Los usuarios normales sólo pueden utilizar el sistema para construir ontologías de manera automática, y no pueden corregir al sistema durante el proceso de extracción de conocimiento.

- **Modos de uso**: La herramienta permite dos modos de uso, a saber, la extracción automática y la extracción semisupervisada. Mediante el modo automático, el sistema extrae todas las entidades de conocimiento que aparecen en el texto, sin posibilidad de interactuar con la herramienta para modificar ese conocimiento. La semisupervisada permite al usuario corregir al sistema, y por lo tanto, entrenarlo en un determinado dominio. Como se ha dicho anteriormente, se asume que sólo un usuario experto puede realizar análisis de textos en modo semi-supervisado de forma eficaz, ya que un usuario normal no está capacitado generalmente para entrenar al sistema.

- **Idiomas de la aplicación**: La herramienta se ha diseñado para que sea independiente del idioma y se ha personalizado para dos idiomas determinados, castellano e inglés.

- **Posibilidad de insertar nuevos tipos de relaciones**: El sistema parte de un grupo de relaciones ontológicas predeterminado que puede ser modificado y

personalizado por cada usuario para representar relaciones necesarias que no se engloben en las relaciones que tiene el sistema por defecto.

- **Entrenamiento del sistema a través de sesiones**: El proceso de extracción de conocimiento se basa en sesiones, formadas por un único texto de un dominio determinado que es procesado mas tarde por un usuario. Estas sesiones semisupervisadas permiten al experto entrenar al sistema en el dominio en cuestión, indicando las entidades de conocimiento que aparecen en cada frase y verificando o eliminando las entidades de conocimiento inferidas por el sistema.

- **Visualización y edición de la ontología**: La ontología resultante del proceso de ontology learning tiene un formato propio XML que puede ser traducido a los lenguajes ontológicos más utilizados, como OWL o RDF. Además, se dispone de una opción para editar y visualizar la ontología de manera gráfica, para que pueda ser completada con nuevas entidades de conocimiento, o bien refinada. Para ello se hace uso de un editor de ontologías implementado por el grupo de investigación al que pertenece el autor de este trabajo.

Por último, se ha presentado una breve comparación de esta herramienta con otros sistemas de ontology learning.

CAPÍTULO 7

CONCLUSIONES Y LÍNEAS FUTURAS

7.1 CONCLUSIONES

Este trabajo representa un intento de simplificación de los procesos de creación de ontologías. En la actualidad, hay, principalmente, dos tendencias en la construcción de ontologías, a saber, la construcción manual de ontologías y la construcción automática o semiautomática, la cual es denominada ontology learning. El mayor problema para la construcción de ontologías manual es el cuello de botella que supone la adquisición de conocimiento y la cantidad de tiempo necesario para la construcción de ontologías en distintos dominios (Shamsfard & Barforoush, 2004). Además, con la creciente necesidad de enriquecer la Web con grandes cantidades de ontologías que capturen el conocimiento del dominio, se han realizado numerosos estudios para favorecer la construcción de ontologías a partir de documentos escritos en lenguaje natural (Omelayenko, 2001).

En general, los mayores factores que se consideran en los sistemas de ontology learning son los siguientes (Shamsfard & Barforoush, 2004):

1) Las entidades de conocimiento que se extraen (conceptos, relaciones, axiomas, instancias, categorías sintácticas y roles temáticos).

2) El punto de partida: la prioridad del conocimiento a adquirir y el tipo, así como el lenguaje de entrada (natural, estructurado, etc.).

3) El preprocesamiento realizado (Procesamiento lingüístico como POS Tagging, análisis morfológico, análisis sintáctico, búsqueda de términos, etc.).

4) Características relacionadas con el método en el que se basa el proceso:

    a) Método de aprendizaje que se ha utilizado: Lingüístico, estadístico o basado en machine-learning.

b) Tipo de aprendizaje (supervisado vs. no supervisado, on-line vs off-line).

c) Grado de automatización (manual, semiautomático, cooperativa, completamente automática).

d) Tipo y cantidad de intervención del usuario.

5) El tipo de ontología extraida (ontologías del domino, ontologías terminológicas o de lenguaje natural, instancias de ontologías, etc)

6) Otras características de la ontología resultante: propósito del uso, tipo de contenido, la estructura y topologías, o el lenguaje de representación.

En base a estos factores, el modelo de ontology learning presentado en esta memora puede clasificarse de la siguiente manera:

1) En lo que respecta a las entidades de conocimiento extraídas, KAText extrae principalmente conceptos y relaciones semánticas, no sólo relaciones taxonómicas. Además, gracias a su motor de inferencia, puede también extraer atributos que pertenezcan a estos conceptos y los valores que puedan tomar estos atributos. Sin embargo, KAText no contempla la extracción de axiomas.

2) Desde la perspectiva del punto de partida, el sistema prioriza la extracción de conceptos y relaciones sobre la de atributos y valores. El texto del que se parte es un texto en lenguaje natural no estructurado, y nuestro sistema hace uso de unas bases de conocimiento que deben ser entrenadas previamente en un determinado dominio para la construcción satisfactoria de ontologías.

3) En cuanto al modo de preprocesamiento realizado, la primera fase de nuestro sistema consiste en realizar un POS-Tagging. Para ello, se desarrolló un POS-Tagger (explicado en el Capítulo 2) que lleva a cabo dicha tarea.

4) En lo concerniente a las características relacionadas con el método en el que se basa el proceso, se puede afirmar lo siguiente:

   a) El método de aprendizaje utilizado por KAText puede clasificarse dentro de los sensibles al contexto, ya que se basa en un sistema de reglas basado en casos MCRDR.

   b) El tipo de aprendizaje es supervisado, debido a que es el propio experto quien entrena el sistema, introduciendo asociaciones entre expresiones lingüísticas y las entidades de conocimiento que va extrayendo a partir del texto.

   c) Con respecto al grado de automatización, es semi-automática por el entrenamiento que requiere el sistema, aunque el modelo puede ser utilizado de forma totalmente automática sin necesidad de la interacción con ningún usuario.

   d) Debido a que se parte de las bases de conocimiento vacías, el grado de interacción con el sistema durante las primeras sesiones es bastante alto, siendo menor conforme el sistema aprende en un determinado dominio.

5) Los tipos de ontologías extraídas por el sistema son ontologías del dominio.

6) El lenguaje de representación utilizado por este sistema se basa en ficheros XML que describen el contenido de la ontología, en base al modelo ontológico del cual se parte en esta tesis, pero estos ficheros pueden ser traducidos a OWL y RDF, por ejemplo, mediante parsers desarrollados por el grupo de investigación al que pertenece el autor de esta memoria.

Actualmente, no existe ningún sistema de ontology learning que sea considerado óptimo, ni mejor, pero la mayoría de estos sistemas se centran en la extracción de jerarquías de conceptos. Además, en la actualidad existen muy pocos sistemas orientados al idioma castellano para la construcción de ontologías, y el número de investigaciones en este campo es cada día más importante. Esta tesis pretende contribuir al avance en los conocimientos dentro del area de investigación de ingeniería ontológica a través de un nuevo método para la construcción de ontologías validada en los idiomas castellano e inglés y que permite la definición de cualquier tipo de relación, además de proporcionar unas relaciones predeterminadas.

7.2 LÍNEAS FUTURAS

Actualmente, se está pensando en el uso de ontologías de lenguaje natural WordNet (Miller, 1990) para verificar que las relaciones inferidas por el sistema sean correctas, así como obtener nuevas relaciones y completar la ontología como se hace en (Navigli *et al*, 2003). Estos autores utilizan Wordnet para interpretar los términos semánticos e identificar principalmente relaciones taxonómicas y de similaridad. WordNet podría ayudar también a la extracción de términos que tengan relación con el dominio y la ontología que se esté pretendiendo construir. El uso de otras ontologías de terminología relacionada con algún dominio específico (como UMLS y GeneOntology), serían muy útiles para detallar la ontología adquirida por el sistema. Es por eso por lo que en un futuro el sistema deberá disponer de la opción de consulta en determinadas ontologías de manera que se puedan enriquecer las posibles ontologías creadas por el sistema.

Otra posible línea futura de actuaciones es la de la mejora de la fase de búsqueda de conceptos añadiendo técnicas de extracción de términos (Jacquemin, 2001), para la obtención de un conjunto de términos candidatos a ser conceptos del dominio.

En un futuro se pretende ampliar el sistema para cubrir los axiomas. El mayor problema de la extracción de axiomas es que el número de elementos de conocimiento involucrados en aquellos es desconocido a priori. Sin embargo, la cantidad de axiomas contenidos en un texto no

es muy significativa comparado con la de las demás entidades de conocimiento contenidas en un texto.

Por último, se está empezando a validar el sistema en varios dominios utilizando métodos estadísticos.

En resumen, la metodología de ontology learning presentada en esta tesis se puede situar en el contexto de Ingeniería Ontológica y Gestión del Conocimiento puesto que los resultados obtenidos son relevantes para ambas áreas de investigación. En primer lugar, es relevante para la Ingeniería Ontológica porque proporciona una metodología para la construcción de ontologías de manera semi automática. Como se ha dicho anteriormente el desarrollo de ontologías es una tarea complicada, por lo que una metodología de ayuda a la construcción de ontologías es altamente beneficiosa para esta área. Finalmente, esta metodología de extracción de conocimiento a partir de texto contribuye a mejorar el desarrollo de sistemas para la gestión de conocimiento al ayudar la consecución de módulos de conocimiento reutilizables y compartibles, los cuales son elementos fundamentales de dichos sistemas.

CAPÍTULO 7

CONCLUSIONES Y LÍNEAS FUTURAS

7.1 CONCLUSIONES

Este trabajo representa un intento de simplificación de los procesos de creación de ontologías. En la actualidad, hay, principalmente, dos tendencias en la construcción de ontologías, a saber, la construcción manual de ontologías y la construcción automática o semiautomática, la cual es denominada ontology learning. El mayor problema para la construcción de ontologías manual es el cuello de botella que supone la adquisición de conocimiento y la cantidad de tiempo necesario para la construcción de ontologías en distintos dominios (Shamsfard & Barforoush, 2004). Además, con la creciente necesidad de enriquecer la Web con grandes cantidades de ontologías que capturen el conocimiento del dominio, se han realizado numerosos estudios para favorecer la construcción de ontologías a partir de documentos escritos en lenguaje natural (Omelayenko, 2001).

En general, los mayores factores que se consideran en los sistemas de ontology learning son los siguientes (Shamsfard & Barforoush, 2004):

1) Las entidades de conocimiento que se extraen (conceptos, relaciones, axiomas, instancias, categorías sintácticas y roles temáticos).

2) El punto de partida: la prioridad del conocimiento a adquirir y el tipo, así como el lenguaje de entrada (natural, estructurado, etc.).

3) El preprocesamiento realizado (Procesamiento lingüístico como POS Tagging, análisis morfológico, análisis sintáctico, búsqueda de términos, etc.).

4) Características relacionadas con el método en el que se basa el proceso:

    a) Método de aprendizaje que se ha utilizado: Lingüístico, estadístico o basado en machine-learning.

b) Tipo de aprendizaje (supervisado vs. no supervisado, on-line vs off-line).

c) Grado de automatización (manual, semiautomático, cooperativa, completamente automática).

d) Tipo y cantidad de intervención del usuario.

5) El tipo de ontología extraida (ontologías del domino, ontologías terminológicas o de lenguaje natural, instancias de ontologías, etc)

6) Otras características de la ontología resultante: propósito del uso, tipo de contenido, la estructura y topologías, o el lenguaje de representación.

En base a estos factores, el modelo de ontology learning presentado en esta memora puede clasificarse de la siguiente manera:

1) En lo que respecta a las entidades de conocimiento extraídas, KAText extrae principalmente conceptos y relaciones semánticas, no sólo relaciones taxonómicas. Además, gracias a su motor de inferencia, puede también extraer atributos que pertenezcan a estos conceptos y los valores que puedan tomar estos atributos. Sin embargo, KAText no contempla la extracción de axiomas.

2) Desde la perspectiva del punto de partida, el sistema prioriza la extracción de conceptos y relaciones sobre la de atributos y valores. El texto del que se parte es un texto en lenguaje natural no estructurado, y nuestro sistema hace uso de unas bases de conocimiento que deben ser entrenadas previamente en un determinado dominio para la construcción satisfactoria de ontologías.

3) En cuanto al modo de preprocesamiento realizado, la primera fase de nuestro sistema consiste en realizar un POS-Tagging. Para ello, se desarrolló un POS-Tagger (explicado en el Capítulo 2) que lleva a cabo dicha tarea.

4) En lo concerniente a las características relacionadas con el método en el que se basa el proceso, se puede afirmar lo siguiente:

   a) El método de aprendizaje utilizado por KAText puede clasificarse dentro de los sensibles al contexto, ya que se basa en un sistema de reglas basado en casos MCRDR.

   b) El tipo de aprendizaje es supervisado, debido a que es el propio experto quien entrena el sistema, introduciendo asociaciones entre expresiones lingüísticas y las entidades de conocimiento que va extrayendo a partir del texto.

   c) Con respecto al grado de automatización, es semi-automática por el entrenamiento que requiere el sistema, aunque el modelo puede ser utilizado de forma totalmente automática sin necesidad de la interacción con ningún usuario.

   d) Debido a que se parte de las bases de conocimiento vacías, el grado de interacción con el sistema durante las primeras sesiones es bastante alto, siendo menor conforme el sistema aprende en un determinado dominio.

5) Los tipos de ontologías extraídas por el sistema son ontologías del dominio.

6) El lenguaje de representación utilizado por este sistema se basa en ficheros XML que describen el contenido de la ontología, en base al modelo ontológico del cual se parte en esta tesis, pero estos ficheros pueden ser traducidos a OWL y RDF, por ejemplo, mediante parsers desarrollados por el grupo de investigación al que pertenece el autor de esta memoria.

Actualmente, no existe ningún sistema de ontology learning que sea considerado óptimo, ni mejor, pero la mayoría de estos sistemas se centran en la extracción de jerarquías de conceptos. Además, en la actualidad existen muy pocos sistemas orientados al idioma castellano para la construcción de ontologías, y el número de investigaciones en este campo es cada día más importante. Esta tesis pretende contribuir al avance en los conocimientos dentro del area de investigación de ingeniería ontológica a través de un nuevo método para la construcción de ontologías validada en los idiomas castellano e inglés y que permite la definición de cualquier tipo de relación, además de proporcionar unas relaciones predeterminadas.

7.2 LÍNEAS FUTURAS

Actualmente, se está pensando en el uso de ontologías de lenguaje natural WordNet (Miller, 1990) para verificar que las relaciones inferidas por el sistema sean correctas, así como obtener nuevas relaciones y completar la ontología como se hace en (Navigli *et al*, 2003). Estos autores utilizan Wordnet para interpretar los términos semánticos e identificar principalmente relaciones taxonómicas y de similaridad. WordNet podría ayudar también a la extracción de términos que tengan relación con el dominio y la ontología que se esté pretendiendo construir. El uso de otras ontologías de terminología relacionada con algún dominio específico (como UMLS y GeneOntology), serían muy útiles para detallar la ontología adquirida por el sistema. Es por eso por lo que en un futuro el sistema deberá disponer de la opción de consulta en determinadas ontologías de manera que se puedan enriquecer las posibles ontologías creadas por el sistema.

Otra posible línea futura de actuaciones es la de la mejora de la fase de búsqueda de conceptos añadiendo técnicas de extracción de términos (Jacquemin, 2001), para la obtención de un conjunto de términos candidatos a ser conceptos del dominio.

En un futuro se pretende ampliar el sistema para cubrir los axiomas. El mayor problema de la extracción de axiomas es que el número de elementos de conocimiento involucrados en aquellos es desconocido a priori. Sin embargo, la cantidad de axiomas contenidos en un texto no

es muy significativa comparado con la de las demás entidades de conocimiento contenidas en un texto.

Por último, se está empezando a validar el sistema en varios dominios utilizando métodos estadísticos.

En resumen, la metodología de ontology learning presentada en esta tesis se puede situar en el contexto de Ingeniería Ontológica y Gestión del Conocimiento puesto que los resultados obtenidos son relevantes para ambas áreas de investigación. En primer lugar, es relevante para la Ingeniería Ontológica porque proporciona una metodología para la construcción de ontologías de manera semi automática. Como se ha dicho anteriormente el desarrollo de ontologías es una tarea complicada, por lo que una metodología de ayuda a la construcción de ontologías es altamente beneficiosa para esta área. Finalmente, esta metodología de extracción de conocimiento a partir de texto contribuye a mejorar el desarrollo de sistemas para la gestión de conocimiento al ayudar la consecución de módulos de conocimiento reutilizables y compartibles, los cuales son elementos fundamentales de dichos sistemas.

CAPÍTULO 8 RESUMEN EN INGLÉS / SUMMARY IN ENGLISH

Extracting knowledge directly from natural language text is a challenging task in Artificial Intelligence and Knowledge Engineering, as it would allow to extract knowledge easily and, what is more, without the intervention of knowledge engineers.

Besides, the need for enriching the Web with large amounts of ontologies has increased. This need for domain models has generated several studies and research on methodologies capable of overcoming the bottleneck provoked by the manual construction of ontologies (Omelayenko, 2001; Shamsfard and Barforoush, 2004). This need has led towards a new research area to obtain semiautomatic methods to build ontologies, which is called, *Ontology Learning*.

Nowadays, most techniques for learning domain ontologies from free natural language text only generates hierarchies of concepts or use a very reduced set of relations (Maedche & Staab, 2001).

The reasons explained in the previous paragraphs were the basic motivations for the accomplishment of this research work. The solution proposed in this work is based on the development of a new environment for incremental knowledge extraction from natural language texts. For this purpose, an ontological engineering perspective has been adopted. Hence, the knowledge acquired through the developed environment is represented by means of ontologies. This work presents a new method for the semiautomatic construction of ontologies from natural language texts. This method is not only based on obtaining hierarchies of concepts, but it uses a set of semantic relations between concepts.

To achieve this goal, the following methodology has been used:

- Analysis of the state of the art in Ontological Engineering. Different definitions of the term ontology and the possible classifications and types of ontologies have been studied.

- Analysis of the current methodologies to design and construct ontologies.

- Analysis of the current methodologies for the automatic construction of ontologies from natural language texts.

- Definition and formalization of an environment for incremental knowledge extraction from natural language text. The knowledge representation structure chosen has been the ontology. This environment is comprised of three sequential phases: POS-Tagging, concepts search phase, and inference phase.

- Design and implementation of a software application for the extraction of ontologies from natural language texts in a semisupervised and incremental manner. In this system, the experts interact with the system by selecting from the texts the explicit knowledge entities that are relevant for the domain. In this way, the system is trained for a certain domain.

ONTOLOGIES – STATE OF THE ART

The state of art in ontological engineering has been analysed. The first part of such analysis was centred on the origin of the notion of ontology. Some definitions of the term ontology, made from an Artificial Intelligence (AI) point of view, have been analysed. In AI, two main approaches for defining ontologies are distinguished. On the one hand, some computing researchers consider that ontologies define the basic terms and relations of a particular domain. On the other hand, the philosophical AI community uses the original philosophical notion of ontology and uses ontologies in a more formal manner. Ontologies are currently the most widespread technology for representing knowledge in AI because they provide structured and formal representation

Moreover, different classifications of ontologies have been proposed, according to different criteria. One of such classifications is based on the knowledge contained in ontologies. This classification distinguishes between the following types of ontologies:

- Linguistic ontologies: They are also called Natural Language Ontologies (NLO); they specify the terms used to represent the knowledge in a determined domain. An example of these types of ontologies are UMLS and WordNet.

- Information ontologies: These ontologies specify the structure of the database records. An example of this type of ontology is the database scheme.

- Ontologies for knowledge modelling: They specify knowledge conceptualizations. The internal structure of this class of ontologies is much richer than the previous classes of ontologies.

Another classification is based on the motivation of the ontology. Two different classifications have been studied attending to this criterion. The first one classifies ontologies in

four classes: Ontologies for knowledge presentation, generic ontologies, domain ontologies and application ontologies. The other classification distinguishes six types of ontologies: general ontologies, categorical ontologies, domain ontologies, generic ontologies, regional ontologies and applied ontologies.

The last classification distinguishes between descriptive and formal ontology. While descriptive ontologies are concerned with the collection of information, formal ontologies distill, filter, codify and organise the results of descriptive ones. In AI, ontologies are heavyweight in case they include axioms or lightweight when no axioms are included. Axioms are one of the elements used to specify ontologies. Other important ontological elements are: classes (or concepts), relations, functions and instances. Different ontological models have been proposed and the differences between such models have a two-folded nature: (1) type of ontology to model; and (2) richness of the ontology model. Relations are an important element in an ontology model provided that they are the instrument for relating ontological entities. Amongst the possible relations, we have distinguished the taxonomy (theory of classification), mereology (theory of parts), and the equivalence, dependence, topological, causal, functional, chronological, conditional and purpose relations.

Due to the increasing importance of ontologies, different mechanisms and methodologies for designing and building ontologies have been proposed. Some of them were developed to obtain particular ontologies whereas other methodologies had a more general purpose. Furthermore, methodologies for automatic building of ontologies (ontology learning) from free text have been studied. The major problems in ontology building are the bottleneck of knowledge acquisition and time-consuming construction of various ontologies for various domains/applications. These methodologies try to avoid that bottleneck helping the user to build the ontology in a semiautomatic manner.

The ontology learning approaches can be divided into three main groups depending on the method used: symbolic approaches, statistic approaches and machine learning-based approaches. Furthermore, each approach can be divided depending on the type of ontology acquired: Natural Language Ontologies (NLO), domain ontologies and ontology instances.

One of the objectives of this thesis is to develop a new methodology for ontology learning from free text in a semiautomatic manner to avoid the problems of the bottleneck that supposes the manual building of ontologies.

PART OF SPEECH TAGGERS

The importance of linguistic engineering and Part of Speech Taggers (POS Taggers) in most natural language processing systems has been analysed. The first part of this analysis was centred in the definition of POS Tagging. Therefore, the state of the art on POS Taggers has been analysed. There exists a possible classification of taggers:

- Linguistic Taggers: They were the first taggers and they are based in some linguistic knowledge expressed by manually-built rules. Amongst these systems, TAGGIT can be highlighted, that is based on context pattern rules. TAGGIT uses a set of grammatical tags with 71 elements and a disambiguation grammar of 3,300 rules. The current linguistic taggers still represent the knowledge as a rule set written by linguistics and obtained by studying the corpus. These taggers are very high precision systems that require a high development cost.

- Statistical Taggers: They obtain models of the language and generalizations from the empirical evidence obtained from large linguistic corpus. The development cost of this class of taggers is much less than the linguistics. These systems are language independent and their main problem resides in the estimation of the parameters of the statistical model used. For example, the disambiguation in CLAWS is based on the choice of the correct category in a corpus statistical database. Nowadays, this type of tagger is widely used as the TnT Tagger, based in Hidden Markov Models (HMM), and

the MXPOST tagger based on a maximum entropy model and some contextual measures.

- Machine Learning based taggers: They incorporate more sophisticated methods for information acquisition, which are based on the paradigm of machine learning. Inside this type of taggers, we can point out the Brill's tagger, based on transformation rules, and the TreeTagger, based on C4.5 decision trees.

Due to the non existence of specific taggers for the Spanish language, and due to the fact that language independent algorithms need tagged corpus which is not avaible for its use, a machine learning based tagger was developed. This tagger, called eTiKeT@, uses the paradigm of decision trees C4.5. This tagger is composed by two phases:

Search phase: In this phase, the most likely grammar category of each word, using a lexicon, is obtained.

Morphologic disambiguation phase: In this phase the grammatical category of each word is evaluated in its context. This phase uses a C4.5 decision tree to infer the correct categories of the word using n-grams formed by the grammar categories of the previous and next words in the current sentence.

Finally, an experiment with the tagger is described.

AN ONTOLOGY LEARNING APPROACH

An ontology learning from free text approach is described in this work. Such an approach is based on the MCRDR methodology for building knowledge-based and expert systems. An implicit assumption (Assumption 1) is that ontologies can be used to represent knowledge. As the whole text can be very long, it seemed convenient to divide it into minor *fragments* in order to facilitate its processing. Furthermore, the approach presented here is based on the idea that relationships are usually associated to verbs in natural language. Also, we decided, as it is usually done, to divide texts into sentences. In this approach, the expert is in charge of building the ontology. This gives rise to another assumption (Assumption 2): experts can extract semantic relationships from text. This expert must have expertise on the specific task described along the text, and the expert is somehow associated with the system and to the text by the task itself. In a way, we can say that knowledge resides inside the text. So, there is another implicit assumption (Assumption 3) in this sentence, namely, text can contain knowledge. Ontologies permit to divide knowledge into categories such as concepts, attributes, relationships, rules, axioms, etc. These knowledge entities can appear explicitly in the text, although sometimes knowledge is only referred to implicitly. The system attempts to find only explicit knowledge from the text.

By taking into account the above assumptions, the starting point is a system composed by three modules, namely, an empty *concept knowledge base*, an empty *verb knowledge base* and an empty *MCRDR sub-system*. At this phase, the system is hence unable to find any knowledge in the input text and the expert has to introduce for knowledge manually. The expert's task is then to identify the relationship associated to the main verb (if any) of the current sentence and all the other knowledge entities of the current fragment of text. He/she will also tell the system the *expressions* in which they appear. The expressions-knowledge associations for the concepts and relations are stored thereafter in the system (the expressions associated with concepts into the conceptual knowledge base, and the expressions associated with relationships into the verbal knowledge base) in order to be used for new knowledge findings thereafter. An MCRDR sub-

system is then created and maintained by the system in order to be used for acquiring the knowledge entities, which participate in the relationship under question.

The MCRDR sub-system rules are based on the relationship that represents the main verb in the current sentence, the grammar category of the knowledge entities, which appear in that sentence, and the position of this knowledge entities with respect the verb in the sentence.

The system shows the relationship and the knowledge entities in the current sentence, and the expert will just have to confirm the results output by the system. If the real conclusion is different from the conclusion inferred by the system, then the expert has to introduce the correct conclusion. After that, the system will add the new case into the MCRDR sub-system and will update the rules.

The knowledge acquisition process is divided into three sequential phases, which have been implemented in three separate modules:

POS-Tagging.

The main objective of this process is to obtain the grammatical category of each word in the current sentence. For this purpose, the C4.5 based POS-tagger described in this thesis is used. This is the first subprocess to be carried out.

For example, if the system gets the sentence "Cardiac glycosides have been used in the treatment of cardiac disease for more than 200 years", after the morphological analysis phase the system will obtain all the words in the sentence labelled with the grammar category in that sentence:

"Cardiac [Adjective] glycosides [Noun] have [Verb auxiliary] been [Verb auxiliary] used [Verb lexical] in [Preposition] the [Determiner] treatment [Noun] of [Preposition] cardiac

[Adjective] disease [Noun] for [Preposition] more [Adverb] than [Conjunction] 200 [Numeral] years [Noun]"

Concept search phase

Through this process, linguistic expressions representing concepts are identified. The associations between linguistic expressions and concepts have been stored in the conceptual knowledge base through a previous training of the system. This process is quite simple and as a result, all the expressions of the fragment which are already in the conceptual knowledge base are obtained.

In this approach, it is assumed that there exist some semantically meaningless words. These words usually have the following grammar categories: Preposition, Conjunction, Interjection, Particle, Pronoun and Determiner. So the system will only search for concepts associated to Nouns, Adjectives and Adverbs.

With all, the system works as follows. First, it takes each word, whose grammar category is noun, adjective or adverb, in the current sentence and looks for ***similar*** words in the already existing expressions in the concept knowledge base. Then, for each expression of the concept knowledge base which is similar to the current word, if it is considered to be an ***acceptable*** expression, these actions are performed: (1) obtain and sort the associated knowledge to the expression present in the knowledge base; (2) create a new expression that matches the concept knowledge-base expression and associates previously sorted associated knowledge to it as possible knowledge; and (3) add the new expression to the list of fragment expressions with its associated knowledge.

Obviously, there might be cases where no good options are found. In that case, the user has to be provided with the possibility of defining new knowledge associated to the expression. Alternatively, these expressions might also be straightforwardly ignored. This implies that the system needs to provide that possibility to the user.

The above referred ***similar*** function is in charge of identifying which expressions of the knowledge base are similar to the current word of the fragment. In its simplest case, it would be an "equal" function. Nevertheless, this function cannot deal with compound expressions by itself; therefore a function of the type "isPrefix" is needed. The "isPrefix" function checks whether the current word is a substring of another word or not.

It would also be desirable that the function could deal with word families (types associated to a single lemma/lexeme) and other language peculiarities. For instance, if the expression "causes" already exists in the knowledge base and the current fragment contains the word "caused", it would be desirable that the system realised that both words actually allude to the same verb (lemma). This issue might be partially implemented using parts-of-speech taggers and lemmatisers. Here, a word in the current fragment is "similar" to an expression in the knowledge base if the expression starts with the current word.

The also above referred ***acceptable*** function is an extension of the "similar" one. As the "similar" function can be very permissive, the "acceptable" function is introduced in order to determine whether the current word and a similar expression are not just "similar by chance". The "isPrefix" function has an important drawback: if the current word is the adjective "cardiac", any expression starting with "cardiac", as "cardiac arrest", "cardiac disease", or "cardiac glycosides" will be (candidates to be) considered as similar.

Therefore, this function limits the number of acceptable options amongst the similar ones. This function has been designed with strong requirements: an existing expression in the database is acceptable if it actually appears in the current fragment.

Inference Phase

In natural language, relationships between concepts are usually associated to verbs (Ruiz-Sanchez *et al* 2003, Valencia-Garcia, *et al* 2004). In the present work, MCRDR is mainly used to find relationships between concepts, but it can also be used to get other knowledge categories like concepts, attributes, or values. This MCRDR component is formed by a knowledge base

that contains linguistic expressions representing generic conceptual relationships, and by an MCRDR subsystem that infers the participants of these relationships. Next, the modus operandi of this process is described. First, the main verb of the current sentence is identified. Then, in case the knowledge base is not empty, the user searches for the type of semantic relationship associated to that verb. Once the type of relation associated to the main verb in the current sentence has been found, the MCRDR sub-system is applied to extract knowledge. This is carried out by using the grammatical category of the words, their position in the current sentence, and the type of relation associated to the verb, if any.

PRACTICAL EXAMPLES OF THE ONTOLOGY LEARNING APPROACH

In this chapter, three detailed examples that describe the modus operandi of the approach, with an emphasis put on the knowledge acquisition process, are shown.

More concretely, the first example is a simple one in the computer components domain. The second is an example of the heart diseases domain, and the language of the text used in this example is English. The last example, belong to the cancer domain, and the language of the text is Spanish.

Through this chapter, the knowledge acquisition process from free text and its use in different languages like Spanish and English have been illustrated.

APPLICATIONS OF THE ONTOLOGY LEARNING APPROACH

In this chapter, the usefulness and applicability of the ontology learning methodology presented in Chapter 3, have been mainly evaluated in two areas. The first area is ontology building for the Semantic Web. In this evaluation, a study in three different domains (Breast cancer, Leukaemia and data encrypting) was done to analyse whether the system is able to learn and suggest knowledge through different sessions. A measure called accuracy has been defined to demonstrate the hypothesis. The accuracy score is the result of the division between the amount of knowledge entities suggested by the system and that are accepted by the expert user, and the total amount of knowledge entities existing in the text. Three hypotheses have been formulated:

- Hypothesis 1 (Similarity): “The system obtains a similar accuracy for each expert”.

- Hypothesis 2 (Usefulness): "The system is useful for the ontology building".

- Hypothesis 3 (Incremental Learning): “The system learns trough the experiment”.

In order to complete the conditions in which the experiment was run, it should be noticed that the knowledge base was initially empty, that is, each person started with an empty knowledge base which was augmenting its content as the knowledge acquisition process evolved.

Through the experiments some of the hypotheses are confirmed. More concretely, the first hypothesis is confirmed in the first and last experiment because the experts have obtained similar results.

The second hypothesis is demonstrated in all the experiments, obtaining differents accuracy rates, obtaining the first experiment the best result and the second experiment the worst result.

The incremental learning hipothesis (hypothesis 3) is demonstrated in all the experiments too, where the system learns through the sessions, in the same text and through different texts.

The other application area is related to command recognition. Here, commands are expressed in natural language. This research work has been applied to medical robotics. For this purpose, the knowledge extraction model presented in chapter 3 has been extended to recognise spoken commands in natural language. A simulator for surgical operations, which interacts with the surgeon through speech, has been developed.

This system tends to translate the surgeon's commands expressed in natural language to a sequence of robot steps to perform these orders. The system is composed by the following modules:

- Speech Processing Module: There exist some commercial speech recognizers (e.g., IBM ViaVioce, Dragon Naturally Speak, etc.) which usually work basically in one of the following two modes. One is command- or control-based recognition, which employs predefined grammars to recognize a user query, Others are dictation-based, which obtain a list of the most probable words of the user query. We have used IBM ViaVoice as the Speech Recognition of the systems using the dictation system to obtain all words which the surgeon pronounces. To integrate IBM ViaVoice into the full system, we have used the Java Speech API, which specifies a cross-platform interface to support command and control recognizers, dictation systems and speech synthesizers. IBM ViaVoice has been used for speech recognition in other implemented systems in thee AI field like the work presented in (K.Dorai G. and Yacoob Y., 2002) where *Embedded Grammar Tags* are used to represent the possible queries that may be launched by users to retrieve particular content into a Semantic Web portal.

- Natural Language Processing Module: This module is the knowledge extraction model presented in this work. The model is based in the three sequential phases explained above: POS-Tagging, Concept Search Phase and Inference Phase.

- Mapping the ontologies acquired by the NLP system onto orders module: Ontologies have been used in this work for representing the knowledge the system possesses, so we have created an ontology related to the surgical domain. Such ontology contains a set of axioms to check for the ontology integrity and completeness. Once the ontology is consistent and complete, it has to be transformed into the final state of the linear planner implemented. Another set of axioms is used to do this translation. Once the partial ontology has been translated into the final state, the system has to plan the steps necessary to get that final state using an STRIPS-based planner.

- STRIPS Planner Module: STRIPS (Fikes and Nilsson, 1971) is a linear planner that attempts to find a sequence of operators in a space of world models to transform a given initial world model into a model in which a given goal formula can be proven true. It represents a world model as an arbitrary collection of first-order predicate calculus formulae and works with models consisting of a large number of formulae. This planner is used to planning the robot steps.

A robotic simulator has been implemented using the Java language, so that the operator can check for the robot motion in a virtual environment prior to the execution of the operation in the real world.

AN ONTOLOGY LEARNING APPLICATION SOFTWARE

In this chapter, a system for ontology extraction from text (ontology learning) has been presented. This system fulfils the requirements of the ontology learning model proposed in previous chapters. The ontology learning system properties presented in this chapter can be summarised as follows:

- **Users**: The system contemplates two types of users: expert and normal. Expert users can use the whole functionality available in the system, like a semisupervised ontology construction and the training the system through the sessions. Normal users are only able to use the system to build ontologies in an automatic way, and they are not able to make corrections to the system.

- **Modes of use**: The system allows two modes of use, namely, an automatic extraction and a semi-supervised extraction. In the automatic way, the system extracts all the knowledge entities that appear in the text. In this case there is no possibility of interacting with the system to modify this knowledge. The semi-supervised one allows the user to make corrections to the system, and therefore, to train it in a certain domain. Only an expert user is able to analyse the texts in a semi-supervised way, since a normal user, generally, is not qualified to train the system.

- **Languages of the system**: The system has been designed to be language independent and it has been tested for Spanish and English.

- **New types of relations**: The system has a predetermined group of ontological relations that can be modified by every user. This group allows to represent necessary relations that are not contained in the set of default ones.

- **System training through sessions**: The knowledge extraction process is based on sessions formed by a text of a certain domain. This text has been tried by a user. These (semisupervised) sessions allow the expert to train the system in a given domain. The expert must indicate the knowledge entities that appear in every sentence. He or she must also verify or eliminate the knowledge entities inferred by the system.

- **Ontology visualization**: The ontology obtained as a result of the ontology learning process is expressed in an XML format that can be translated into the most used ontological languages, such as OWL or RDF. Besides, the system has an option to edit and visualize the ontology in a graphical way, so it can be completed by adding new knowledge entities or by refining the existing ones.

Finally, a brief comparison between this tool and other ontology learning systems has been presented in this chapter. This comparison is based in two main criteria: the semantic relationships supported, and the automation of the knowledge extraction process.

CONCLUSIONS AND FUTURE WORK

This work represents an attempt to simplify ontology building processes. Nowadays, there are two main trends in ontology building: the manual ontology building, and the automatic or semiautomatic one, named ontology learning. The major problems in ontology building are the bottleneck of knowledge acquisition and time-consuming construction of various ontologies for various domains applications. (Shamsfard and Barforoush, 2004). Besides, due the increasing need for enriching the Web, lots of ontologies that capture the domain theories and several studies to make the knowledge acquisition processes automatic have been developed (Omelayenko, 2001).

In general, the major factors that distinguish ontology learning systems are the following (Shamsfard & Barforoush, 2004):

1) The learned elements (concepts, relations, axioms, instances and syntactic categories and thematic roles).

2) Starting point: prior knowledge and the type and language of input (natural, structured, …).

3) Pre-processing (linguistic processing such as POS Tagging, morphological analysis, syntactic analysis, spotting terms, …).

4) Learning method consisting of

    a) Learning approach: linguistic, statistical or machine-learning based.

    b) Learning category (supervised vs. non supervised, on-line vs off-line).

c) Degree of automation (manual, semi-automatic, cooperative, fully automatic).

d) Type and amount of user intervention.

5) The resulting ontology (domain ontologíes, natural language ontologies, instances of ontologies, etc)

6) Other features of the resulting ontology: purpose of use, type of content, structure and tipologies, or the representation language.

Using these factors, the ontology learning model presented in this work can be classified as follows:

1) The learned elements: The system extracts mainly concepts and semantic relations, not only taxonomic relations. Besides, the system can extract attributes and values too.

2) Starting point: The system tries to extract concepts and relations mainly. The input text is a free natural language text, and the system uses knowledge bases that must be trained in advance in a certain domain to obtain good ontologies.

3) Pre-processing: The first phase of the system is to perform a POS-Tagging. The POS-Tagger (explained in Chapter 2) is used in this pre-processing phase.

4) Concerning the learning method:

a) The learning approach used by the system can be classified as a machine-learning based one, because it is based on a rule based system.

b) The type of learning is supervised. The experts train the system by introducing associations between linguistic expressions and the knowledge entities extracted from the text.

c) Degree of automation: This methodology is semi-automatic due to the existence of a training phase, though the model can be used in a full automatic manner without the intervention of any user.

d) Due the initial empty state of the knowledge bases used by the system, the intervention of the users in the system during the first sessions is important. However, the human intervention decreases with the amount of sessions.

5) The resulting ontology is a domain ontology.

6) The representation language used is based in an XML format that describes the content of the ontology. This format can be translated into OWL or RDF, for example, through parsers developed by our research group.

Nowadays, none of the existing ontology learning systems can be considered the best one. The majority of these systems are centred on the extraction of concepts hierarchies. In addition, there exist very few systems for Spanish language, whereas the amount of researches in this field is becoming more important day-by-day. This thesis tries to contribute to the ontological engineering community providing a new ontology learning method from free text validated in both Spanish and English languages. This method allows for the definition of any type of relation, besides providing a predetermined set of relations.

We are currently planning the use of WordNet (Miller, 1990) to verify the correctness of the relations inferred by our system and to obtain the new relations as it is done in (Navigli *et al*, 2003). These authors use Wordnet to interpret semantic terms and to identify mainly taxonomic and similarity relations. In particular, Wordnet can contribute to the domain related terms

extraction. The use of another natural language ontologies (such as UMLS in medical domains), would also be very useful to enrich the acquired ontology.

Another possible future work is to improve the concepts search phase using terms extraction technologies (Jacquemin, 2001), to obtain a set of terms candidates for being domain concepts.

We are planning to extend the system to cover axioms. The major problem of axioms extraction is that the number of knowledge elements involved in axioms is not known a priori. Nevertheless, we think the quantity of axioms contained in a text is not very significant in comparison to the amount of other knowledge entities contained in a text.

Finally, the system will be validated in several domains using statistical methods.

In summary, the ontology learning methodology presented in this work can be contextualised in the areas of Ontological Engineering and Knowledge Management since the obtained results are relevant for both research areas. First, it is relevant for the Ontological Engineering community because it provides a new methodology for semi automatic ontology construction. The ontology construction is a hard task, so to provide a methodology that helps in its construction is a relevant contribution for this area. Finally, this methodology improves the development of knowledge management systems helping in the development of reusable and shareable knowledge modules, which are fundamental elements of the above mentioned systems.

REFERENCIAS

Agirre E., Ansa O., Hovy E., Martinez D. (2000) Enriching very large ontologies using the WWW. In: S. Staab, A. Maedche, C. Nedellec, P. Wiemer-Hastings (eds.), Proceedings of the Workshop on Ontology Learning, 14th European Conference on Artificial Intelligence ECAI'00, Berlin, Germany

Alani H., Kim S., Millard D. E., Weal M. J., Hall W., Lewis P.H., and Shadbolt N. R. (2003). *IEEE Intelligent Systems*. January/February 2003 14-21

Albertrazzi, L (1996) "Material and Formal Ontology", in R. Poli, P. Simons (eds.), *Formal Ontology:199-232*, Kluwer, Dordrecht.

Assadi, H. (1999) Construction of a regional ontology from text and ist use within a ocumentary system. In *N. Guarino (ed.), Formal Ontology in Information Systems, Proceedings of FOIS-98, Trento, Italy, 1999*, pages 236–249.

Aussenac-Gilles, N., Biebow, B., Szulman, S (2002) Modelling the travelling domain from a NLP description with Terminae. *Workshop on Evaluation of Ontology Tools, European Knowledge Acquisition Workshop*, Sigüenza, Spain.

Austad, A., Elle, O.J., and Røtnes, J.S. (2001) Computer-aided planning of trocar placement and robot settings in robot-assisted surgery, *International Congress Series* Volume 1230, June 2001, pp. 1020-1026.

Autret M. (1996) Transfer of space technology to the field of medicine, *Annales de Chirurgie* 50, 159 - 161.

Beatty, J, Biggs, P.J., Gal, K., Okunieff, P., Pardo, F.S., Harte, K.J., Dalterio, M.J., Sliski, A. P. (1996) A new miniature x-ray device for interstitial radiosurgery: dosimetry, *Medical Physics* Vol. 23 n. 1 (January 1996), 53-62.

Bernaras, A., Laresgoiti, I., Corera, J. (1996) Building and Reusing Ontologies for Electrical Network Applications. In *Proceedings of the European Conference on Artificial Intelligence (ECAI96):298-302*.

Berners-Lee, T., Hendler, J., Lassila, O. (2001) The Semantic Web. *Scientific American*, May 2001.

Bisson, G. , Nedellec, C., Canamero, D. (2000) Designing Clustering Methods for Ontology Building - The Mo'K Workbench. In: S. Staab, A. Maedche, C. Nedellec, P. Wiemer-Hastings (eds.), Proceedings of the Workshop on Ontology Learning, 14th European Conference on Artificial Intelligence ECAI'00, Berlin, Germany.

Borigault, D. (1996) LEXTER, a Natural Language tool for terminology extraction. In Proceedings, 7th EURALEX International Congress, 771-779, Göteborg.

Borst, W.N (1997). Construction of Engineering Ontologies for Knowledge Sharing and Reuse. CTIT Ph.D-thesis series No.97-14. University of Twente. Enschede, The Netherlands.

Brants, T., (2000) TnT – a statistical part-of-speech tagger, Proceedings of the sixth conference on applied natural language processing, ANLP-2000, Seattle, WA

Brill, E. (1992) A Simple Rule-Based Part-of-Speech Tagger. In Proceedings of the 3rd Conference on Applied Natural Language Processing, ANLP, 152-155.

Brill, E. (1995) Transformation-based error-driven learning and natural language processing: A case study in part of speech tagging. Computational Linguistics, 21(4):543-565

Brill, E. and Resnik, P. (1994) A rule-based approach to prepositional phrase attachment disambiguation. In Proceedings of the Fiftennth International Conference on Computational Linguistics (COLING-1994), pages 1198-1204, Kyoto

Brill, E. and Wu., J. (1998) Classifier combination for improved lexical disambiguation In Proceedings of COLING-ACL'98, pages 191-19, August.

Buchanan, B. (1986). Expert systems: working systems and the research literature. *Expert Systems*, 3, 32-51.

Church, K.W. (1988). A stochastic parts program and noun phrase parser for unrestricted text. Proceedings of the 1st Conference on Applied Natural Language Processing, ANLP (pp. 136–143). ACL.

Compton, P., Horn, R., Quinlan, R. & Lazarus, L. (1989). Maintaining an expert system, In J. R. Quinlan (Eds.), *Applications of Expert Systems*, London, Addison Wesley, 366-385.

Compton, P., Ramadan, Z., Preston, P., Le-Gia, T., Chellen, V. & Mullholland, M. (1998). A trade-off between domain knowledge and problem-solving method power. *11th Banff knowledge acquisition for knowledge-based systems workshop*, Banff, SRDG Publications, University of Calgary. SHARE, 17, 1-19.

Couturat, L. (1903) Opuscules et fragments inédits de Leibniz, Paris, France.

Cutting, D., Kupiec, J., Pedersen, J., and Sibun P. (1992). A practical Part of Speech Tagger. In Proceeding of the Third Conference on Applied Natural Language Processing, 133-140

D'Attellis, N., Loulmet, D., Carpentier, A., Berrebi, A., Cardon, C., Severac-Bastide, R., Fabiani, J. and Safran, D. (2002). Robotic-assisted cardiac surgery: Anesthetic and postoperative considerations, *Journal of Cardiothoracic and Vascular Anesthesia* Volume 16 (August 2002), 397-400

Daelemans, W., Zavrel, J., Berck, P., & Gillis, S. (1996). MBT: A memory-based part-of-speech tagger generator. Proceedings of the 4th Workshop on Very Large Corpora (pp. 14–27). Copenhagen, Denmark.

Daille, B. (1996) Study and implementation of combined techniques for automatic extraction of terminology. In Judith L. Klavans and Philip Resnik, editors, *The Balancing Act: Combining Symbolic and Statistical Approaches to Language.* MIT Press, Cambridge, MA, 49-66.

David, S. et Plante, P. (1990). De la nécessité d'une approche morpho-syntaxique dans l'analyse de textes. *ICO*, 2(3):140-154.

Davis, R., Sorbe, H., Szolovits, P. (1993). What is a Knowledge Representation? *AI Magazine. Spring, 17-33*.

Davulcu, H., Vadrevu S., Nagarajan S., and Ramakrishnan I.V. (2003) OntoMiner:Bootstrapping and Populating Ontologies from Domain-Specific Web Sites. *IEEE Intelligent Systems*, September/October 2003, 24-33

DeRose, S.J. (1988). Grammatical category disambiguation by statistical optimization. Computational Linguistics, 14, 31–39.

Dinsmore, M., Harte, K. J., Sliski, A. P., Smith, D.O., Nomikos, P. M., Dalterio, M.J, Boom, A.J., Leonard, W.F., Oettinger, P.E., Yanch J.C. (1996) A new miniature X-ray source for interstitial radiosurgery: device description*, Medical Physics* vol. 23 n. 1 (January 1996), 53 - 62.

Dorai G. K. and Yacoob Y. (2002), Embedded Grammar Tags:Advancing Natural Language Interaction on the Web. *IEEE Intelligent Systems* January-February 2002, 48-53

Enguehard, C. and Pantera, L. (1995) Automatic natural acquisition of a terminology. *Journal of Quantitative Linguistics*, 2(1):27-32

Faure, D. and Nedellec, C. (1998) A corpus-based conceptual clustering method for verb frames and ontology acquisition. In *In LREC workshop on Adapting lexical and corpus resources to sublanguages and applications, Granada, Spain*.

Fearing R. S., Moy G., Tan E. (1997) Some Basic Issues in Teletaction*, IEEE Int. Conf. Robotics and Automation*, April 1997.

Fensel, D, and Musen, M. (2001): Special Issue on Semantic Web Technology, *IEEE Intelligent Systems (IEEE IS)*, 16(2), 2001.

Federici, S. and Pirrelli, V. (1994). Context-sensitibity and linguistic structure in analogy-based parallel networks. Current issues in Mathematical Linguistics. 353-362

Fellbaum, C. (1998) *WordNet: An Electronic Lexical Database*. The MIT Press, 1998.

Fernández, M., Gómez-Pérez, A., Juristo, N. (1997) METHONTOLOGY: From Ontological Art to Ontological Engineering. Workshop on Ontological Engineering. Spring Synposium Series de la AAAI (American Association for Artificial Intelligence):33-40. Stanford, USA.

Fernández-López, M., Gómez Pérez, A., Rojas Amaya, M. D. (2000). Ontologies crossed life cycles. *Workshop on Knowledge Acquisition, Modelling and Management (EKAW):65-79*. Editor Springer Verlag. Jean Les Pins (Francia).

Fernández-López, M., Gómez-Pérez, A., Pazos-Sierra, A., Pazos-Sierra, J. (1999) Building a Chemical Ontology Using METHONTOLOGY and the Ontology Design Environment. *IEEE Intelligent Systems & their applications. January/February Pages 37-46*.

Fikes, R. E. and Nilsson, N. J (1971) Strips: A new approach to the application of theorem proving to problem solving, *Artificial Intelligence*. 2, 189-208

Fujii, A. and Ishikawa, T. (1999) Utilizing the World Wide Web as an encyclopedia: Extractin term descriptions from semi-structured texts. In *Proocedings of ACL-99*. Association for Computational Linguistics, New Brunswick, NJ.

Garside, R., Leech, G., & Sampson, G. (Eds.) (1987). The computational analysis of English: A corpus-based approach. London: Longman.

Gerstl, P., Pribbenow, S. (1995). Midwinters, end games and body parts: A cassification of partwhole relations. *International Journal of Human-Computer Studies, 43, 865-889*.

Gómez-Pérez, A. Moreno, A., Pazos, J., Sierra-Alonso, A.. (2000) Knowledge Maps: An essential technique for conceptualisation, *Data & Knowledge Engineering, 33(2), 169-190*.

Gray B.L., Fearing R.S. (1996) A surface micromachined microtactile sensor array, *Proceedings of 1996 IEEE International Conference on Robotics and Automation*, 1996, vol.1, pp. 1-6.

Greene, B.B. & Rubin, G.M. (1971). Automatic grammatical tagging of English. Technical Report, Department of Linguistics, Brown University.

Gruber, T. R. (1993) “A translation approach to portable ontology specifications”. *Knowledge Acquisition Vol. 5:199-220*.

Gruber, T. R. (2004) "Interview to Tom Gruber", *AIS SIGSEMIS Bulletin 1(3) October 2004* Página 4

Gruninger, M. Fox, M.S. (1995). The logic of enterprise modelling. In J. Brown & D.O. Sullivan, Eds. *Reenginering the Enterprise: 83-98*. London: Chapman & Hall.

Guarino, N. (1995). Formal Ontology, Conceptual Analysis and Knowledge Representation. *International Journal of Human and Computer Studies, 43(5/6): 625-640*

Guarino, N. (1998). Formal Ontology and Information Systems. *Proceedings of FOIS'98: 3-15*

Guarino, N., Carrara, M., Giaretta, P. (1994). Formalizing Ontological Commitment. In *Proceedings of National Conference on Artificial Intelligence (AAAI-94):560-567*. Seattle, Morgan Kaufmann.

Guarino, N., Welty, C. (2000). Identity, Unity, and Individuality: Towards a Formal Toolkit for Ontological Analysis. In H. Werner (ed.) *ECAI-2000: The European Conference on Artificial Intelligence:219-223*. IOS Press, Berlin, Germany.

Guarino, N., Welty, C. (2001). Identity and subsumption. In R. Green, C. A. Bean, and S. Hyon Myaeng (eds.), *The Semantics of Relationships: An Interdisciplinary Perspective,* Kluwer.

Harris S.J., Arambula-Cosio F., Hibberd R.D., Davies B.L., Wickham J.E.A., Nathan M.S., Kundu B. (1997): The PROBOT: an active robot for prostate resection, *Proc. Instr. Mech. Engrs.* Vol. 211 part H , 317- 325.

Hahn, U. and Schnattinger K. (1998a) Ontology engineering via text understanding. In *IFIP'98 —Proceedings of the 15th World Computer Congress*, Vienna and Budapest.

Hahn, U., Schnattinger, K., (1998b). Towards text knowledge engineering. Proceedings of 15th National Conference on Arti.cial Intelligence (AAAI'98), Madison, WI, pp. 524–531.

Hastings, P. M. *Automatic Acquisition of Word Meaning from Context*. PhD thesis, University of Michigan, 1994.

Hearst, M.A. (1992) Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th International Conference on Computational Linguistics*. Nantes, France.

Heyer, G., L.auter, M., Quasthoff, U., Wittig, T., Wolff, C. (2001) Learning relations using collocations. In *Proceedings of the IJCAI 2001 Workshop on Ontology Learning (OL'2001)*, Seattle, USA.

Hobbs (1993) The generic information extraction system. In *Proceedings of the Fifth Message Understanding Conference (MUC-5)*, Morgan Kaufmann.

Hoffmann J. (2001) The Fast-Forward Planning System, *AI Magazine* (2001), 57-61

Horn, K., Compton, P. J., Lazarus, L. & Quinlan, J. R.(1985). An expert system for the interpretation of thyroid assays in a clinical laboratory, *Aust Comput J*, 17, 7-11.

Jacquemin C. (2001) Spotting and Discovering Terms through Natural Language Processing. MIT Press

Justeson, J. S. and Katz S. M. (1995) Technical terminology: some linguistic properties and an algorithm for identification in text. *Natural Language Engineering* 1(1):9-27

Kant, I. (2001) (Published) *Lectures on metaphysics - Part III Metaphysik L2*. Cambridge University Press.

Karlsson, F., Voutilainen, A., Heikkilä, J. & Anttila, A. (1995) *Constraint Grammar. A Language Independent System for Parsing Unrestricted Text*. Mouton de Gruyter.

Kaufman, L. and Rousseeuw, P.J (1990) Finding Groups in Data: An Introduction to Cluster Analysis. John Wiley.

Kietz, J., Maedche, A., Volz, R. (2000) A Method for Semi-Automatic Ontology Acquisition from a Corporate Intranet. *Workshop on Ontologies and Texts*. Juan-les-Pins**.**

Knight, K., Luk, S. (1994). Building a Large Base for Machine Translation. In Proceedings of the American Association of Artificial Intelligence Conference (AAA-94):773-778. Seattle, WA.

Koneckny C.: *AESOP lends a hand in the OR*, Today's Surgical Nurse 18 (Mar/Apr 1996), 28-32.

Kremer, S. (2001). Spatio-Temporal Connextionist Networks: A Taxonom and Review. *Neural Computation 13:249-306*.

Kukleta J.: Use of robotics in Surgery: AESOP 2000, Proceedings of XXXIII Congress of the European Society for Surgical Research, Padova (Italy), April 1998.

Lenat, D.B. and Guha, R.V. (1990) Building Large Knowledge Based Systems. Representation and Inference in the CYC Project. Addison-Wesley, Reading, MA.

Lindberg, D.A.B., Humphrey, B.L., McCray, A.T. (1993). The Unified Medical Language System. *Methods of Information in Medicine. 32, 281-291*.

Maedche A. and Staab, S. (2000). Mining ontologies from text. In *Proceedings of EKAW-2000, Springer Lecture Notes in Artificial Intelligence (LNAI-1937)*, Juan-Les-Pins, France, 2000. Springer, 2000.

Maedche A., and Staab S. (2001) Ontology Learning for the Semantic Web, *IEEE Intelligent Systems*, vol. 16, no. 2, pp. 72 – 79.

Mahesh, K. (1996). *Ontology Development for Machine Translation: Ideology and Methodology*. New Mexico State University, Computing Research Laboratory, MCC-96-292.

Manning, C.D. and Schuetze, H. (1999) *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, Massachusetts.

Marcus, M.P., Marcinkiewicz, M.A., & Santorini, B. (1993). Building a large annotated corpus of english: The penn treebank. Computational Linguistics, 19(2), pp. 313–330.

Márquez, L., Padró, L. and Rodriguez, H. (2000). A Machine Learning Approach to POS Tagging. *Machine Learning*, 39, 59–91,

Marti A. Hearst and Hinrich Sch¨utze (1993). Customizing a lexicon to better suit a computational task. In *Proc. of the ACL-SIGLEXWorkshop on Acquisition of Lexical Knowledge from Text, Columbus, Ohio, USA*, 1993.

Meinong. A (1921) Self-presentation - published in: Raymond Schmidt (ed.) *Die Deutsche Philosophie der Gegenwart in Selbstdarstellung - vol. I*

Merialdo, B. (1994). Tagging english text with a probabilistic model. Computational Linguistics, 20(2), 155–171.

Millar, A. (1990). WordNet: An On-line Lexical Resource. *Journal of Lexicography*, 3(4).

Mizoguchi, R., Vanwelkenhuysen, J., Ikeda, M. (1995). Task Ontology for Reuse of Problem Solving Knowledge. *Towards Very Large Knowledge Bases: KnowledgeBuilding and Knowledge Sharing: 46-59.*

Morales-Carrasco R. and Gelbukh A. (2003). Evaluation of TnT Tagger for Spanish. In Proceedings of the Fourth Mexican International Conference on Computer Science (ENC'03).

Morin. E. (1999) Automatic acquisition of semantic relations between terms from technical corpora. In *Proc. of the Fifth International Congress on Terminology and Knowledge Engineering - TKE'99*, 1999.

Navigli R. and Velardi P. (2004). Learning Domain Ontologies from Document Warehouses and Dedicated Web Sites. *Computational Linguistics*, vol 30(2), 151-179

Navigli R., Velardi P. and Gangemi A. (2003). Ontology Learning and Its Application to Automated Terminology Translation. *IEEE Intelligent Systems*, January/February 2003, 22-31

Nierenburg, S., Levin, L., 1992. Syntax driven and ontology driven lexical semantics. In: *Pustejovsky, J., Bergler, S. (Eds.), Lexical Semantics and Knowledge Representation. LNAI*, Vol. 627. Springer, Berlin, pp. 5–20.

Ogata N. and Collier N. (2004). Ontology Express: Statistical and Non-Monotonic Learning of Domain Ontologies from Text. *In Proceedings Workshop on Ontology Learning and Population ECAI 2004*. Valencia. Spain

Omelayenko, B. (2001). Learning of Ontologies for the Web: the Analysis of Existent Approaches. *Proceedings of the International Workshop on Web Dynamics*, London, UK

Pereira, F., Tishby, N. and Lee, L. (1993) Distributation Clustering of English Words. In *Proceedings of the ACL-93, 1993*, pages 183–199.

Pisanelli, D., Gangemi, A., Steve, G. (1998). An Ontological Analysis of the UMLS Methatesaurus. Proceedings of AMIA 98 Conference, 1998.

Poli, R. (2000) Levels of Reality. BISCA 2000: Bolzano International Schools in Cognitive Analysis "Dependence and Dynamic Categories".

Poli, R. (2001) Alwis: Ontology for Knowledge Engineers, PhD thesis, Utrecht

Quine, W.O. (1961). *From a Logical Point of View, Nine Logico-Philosophical Essays*. Cambridge, Massachusetts: Harvard University Press

Quinlan J.R. (1993). C4.5: programs for Machine Learning, San Mateo, *Morgan Kaufmann*.

Ratnaparkhi, A. (1996) A maximum entropy part-of-speech tagger. In Proceedings of the First Conference on Empirical Methods in NLP, pages 133-142, Philadelphia, PA

Rector, A., Solomon, W., Nowlan, W., Rush, T. (1995). A Terminology Server for Medical Language and Medical Information Systems. *Methods of Information in Medicine, Vol. 34, 147-157*.

Refanidis I. and Vlahavas I. (2001) The GRT Planner, *AI Magazine* (2001), 63-65

Reuthebuch, O., Ecknauer, E., Zünd, G. and Turina, M. (2002) Total robotic-enhanced pericardiectomy for effusive pericarditis, *Interactive Cardiovascular and Thoracic Surgery* Volume 1 Issue 2 (December 2002), 102-104.

Resnik, P. S.(1993) *Selection and Information: A Class-based Approach to Lexical Relationships*. PhD thesis, University of Pennsylania, 1993.

Resnik P. and Smith N.A. (2003) The Web as a Pararell Corpus. *Computational Linguistics*. Vol 29(3), 349-380

Ruiz-Sánchez, J. M., Valencia-García, R., Fernández-Breis, J. T., Martínez-Béjar, R. and Compton, P. (2003) An approach for incremental knowledge acquisition from text. *Expert Systems with Applications* 25(2): 77-86.

Santamaría C., Gonzalo J. and Verdejo F. (2003) Automatic Association of Web Directories with Word Senses. *Computational Linguistics*. Vol 29(3), 485-502

Schmid, H. (1994a). Part-of-speech tagging with neural networks. Proceedings of the 15th International Conference on Computational Linguistics, COLING (pp. 172–176). Kyoto, Japan.

Schmid. H, (1994b). Probabilistic part-of-speech tagging using decision trees. In Proceedings of the International Conference on New Methods in Language Processing, Manchester, UK.

Shamsfard M. and Barforoush A. (2004) Learning ontologies from natural language texts, *International Journal of Human-Computer Studies* 60(1): 17-63.

Simons, P. (1987). *Parts: A study in Ontology.* Clarendon Press, Oxford.

Slaney J. and Thiébaux S.(2001) Blocks World revisited, *Artificial Intelligence* 125

Sowa, J.F. (2000) Knowledge Representation: Logical, Philosophical, and Computational *Foundations*, Brooks Cole Publishing Co., Pacific Grove, CA.

Staab, S., Schnurr, H.P., Studer, R., Sure, Y.(2001) Knowledge Processes and Ontologies, *IEEE Intelligent Systems, 16(1)*.

Studer, R., Benjamins, R., Fensel, D. (1998) Knowledge Engineering: Principles and Methods. *Data and Knowledge Engineering 25(1-2):161-197*.

Sundblad, H., (2002) Automatic acquisition of hyponyms and meronyms from question corpora. *Proceedings of the ECAI 2002 Workshop on Machine Learning and Natural Language Processing for Ontology Engineering* (OLT'2002), Lyon, France.

Suryanto, H. and Compton, P. (2000) Learning Classification taxonomies from a classification knowledge based system. In: S. Staab, A. Maedche, C. Nedellec, P. Wiemer-Hastings (eds.), Proceedings of the Workshop on Ontology Learning, 14th European Conference on Artificial Intelligence ECAI'00, Berlin, Germany, August 20-25, 2000.

Swartout, B., Patil, R. Knight, K., Russ, T. (1997) Toward distributed use of large-scale ontologies. In *AAAI-97 Spring Symposium Series on Ontological Engineering*.

Tendick F., Cavusoglu M. C. (1997) Human Machine Interfaces for Minimally Invasive Surgery, *Proceedings of the 19th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBS'97),* Chicago (USA), October 30- November 2 1997.

Uschold, M., Groninger, M. (1996) Ontologies: Principles Methods and Applications. Knowledge. *Engineering Review. Vol. 2.*

Uschold, M., King, M. (1995) Towards a Methodology for Building Ontologies. *Workshop on Basic Ontological Issues in Knowledge Sharing*.

Valencia-García, R., Ruiz-Sánchez, J.M., Vivancos-Vicente, P.J., Fernández-Breis, J.T., Martínez-Béjar, R. (2004a) An incremental approach for discovering medical knowledge from texts. *Expert Systems with Applications* 26(3):291-299

Valencia-García, R., Castellanos-Nieves, D., Vivancos-Vicente, P.J., Fernández-Breis, J.T., Martínez-Béjar, R., García-Sánchez, F. (2004b) An approach for Ontology Building from text supported by NLP techniques. *Lecture Notes in Artificial Intelligence vol 3040: Current Topics in Artificial Intelligence*

Vivancos Vicente, P.J., Valencia-Garcia, R. Fernandez Breis, J.T., Martinez Bejar, R. (2004) An approach for multirelational ontology modelling. *Lecture Notes in Artificial Intelligence vol 3157: Proceedings of 8th Pacific Rim International Conference on Artificial Intelligence, Volume Editor(s): Chengqi Zhang, Hans W. Guesgen, Wai K. Yeap*

Van Heijst, G. Schreiber, A.T., & Wielinga, B. J. (1997). Using explicit ontologies in KBS development. *International Journal of Human Computer Studies*, 45, 183-292.

Webb, G., Wells, J. and Zheng, Z. (1999) An Experimental Evaluation of Integrating Machine Learning with Knowledge Acquisition. *Machine Learning*, 31(1): 5-23.

Zavrel J., and Walter D. (1999). Recent Advances in Memory-Based Part-of-Speech Tagging. In Proceedings of the Sixth International Symposium of Social Communication.