# Un Modelo Terminológico para el Problema de Recuperación de Datos

Alvaro Barreiro[1] Raúl Ramos[1,2] Pedro Cabalar[1]
[1]Dept. Computación. Fac. Informática
Universidade da Coruña.
Campus de Elviña 15071 La Coruña, España
e-mail: {barreiro,cabalar}@dc.fi.udc.es
[2]CERN. European Laboratory for Particle Physics
Information Technology Division
1211 Genève, Suiza
e-mail: Raul.Ramos@cern.ch

**Palabras Clave**: Representación del Conocimiento, Aplicaciones de la Inteligencia Artificial, Lógicas Descripcionales, Sistemas Terminológicos, Recuperación de Datos.

**Resumen**: *Este trabajo presenta una formalización basada en Lógicas Descripcionales del modelo de descubrimiento de recursos más comunmente utilizado en Internet. Las Lógicas Descripcionales son básicamente un subconjunto de la lógica de predicados en donde la expresividad y complejidad computacional han sido exhaustivamente estudiadas. Por otro lado los mecanismos de descubrimiento de recursos son utilizados para publicar y recoger información acerca de los recursos disponibles a través de la red. Nuestro enfoque pretende mostrar una aproximación formal con vistas a enriquecer la semántica de los mecanismos de descubrimiento de recursos y ampliar los campos de aplicación de las Lógicas Descripcionales.*

## 1 Introducción

Las **Lógicas Descripcionales** (LD) son formalismos que nos permiten expresar, organizar y manipular el conocimiento acerca de un dominio. En el marco de la Inteligencia Artificial, proporcionan métodos y procedimientos para representar, recuperar y reutilizar el conocimiento existente relevante a un cierto dominio. Por esto también se les llama *Lógicas Terminológicas*. Las LD proporcionan generalmente facilidades para mantener y explotar una taxonomía de conceptos. Pueden considerarse las sucesoras formales de las redes semánticas y más concretamente de la KL-ONE [BS85].
Durante los últimos 15 años se ha estudiado la expresividad y complejidad de las LD, así como sus propiedades y limitaciones computacionales [DLNN95], [LB85]. Existen diversos lenguajes de conceptos (los más conocidos son CLASSIC [BRMR89], Kris [BH91] y Loom [MB87]) que son implementaciones de las LD y aplicaciones prácticas de los mismos que han sido usadas en una cantidad considerable de dominios. Por otra parte la idea de combinar los formalismos lógicos con la WWW no es nueva y se han realizado estudios en diversos campos, tales como la programación en Internet usando sistemas lógicos [HC96] o el modelado de espacios WWW como bases de datos deductivas [LDS96].
En este trabajo presentamos un aplicación de las lógicas descripcionales en el ámbito de los recursos que proliferan hoy en día en Internet. Proponemos una representación en LD de los mecanismos típicos utilizados en la localización y descripción de recursos en este contexto. Con esto pretendemos exponer la posibilidad de usar las ideas y formalizaciones

que proporcionan las LD en un área en auge.
La sección 2 enmarca el contexto en el que consideramos los recursos de Internet que modelamos en LD. La sección 3 es una introducción a las LD desde un punto de vista formal y la sección 4 expone nuestra aproximación a la utilización de LD en el tratamiento de recursos de Internet. Este trabajo finaliza con una breve exposición de las ventajas de la aproximación terminológica.

# 2 El Problema de Recuperación de Datos

Con el crecimiento y la popularidad de los servicios basados en Internet cualquier usuario tiene acceso una cantidad considerable de datos e información. Este trabajo trata con el problema de **recuperación de datos**, nombre con el que pretende distinguirse del problema de **recuperación de información**. Aunque comparten algunos protocolos y herramientas, este último tiene un sentido más amplio y trata principalmente con datos no estructurados.
El problema de recuperación de datos es importante en organizaciones en las que se establecen colaboraciones entre distintos usuarios o grupos de trabajo. Los sistemas deben proporcionar servicios que ayuden a los distintos usuarios en la búsqueda de datos. Estos servicios pueden ofrecer muchas clases de datos, desde directorios de gente (páginas blancas) hasta almacenes de libros y documentación catalogada. Desde el punto de vista del usuario, debe saber dónde buscar y cómo buscar. Para los proveedores de datos el problema reside en la uniformidad con la cual distintos proveedores deberían ofrecer la misma clase de datos y la organización general de los mismos.
La solución al problema de recuperación de datos exige un protocolo que permita el intercambio de los datos ofrecidos y el establecimiento de los mecanismos para su construcción, mantenimiento y consulta. Sin embargo, en el presente trabajo pretendemos únicamente mostrar un modelo de los datos del problema y no entraremos en detalles de la arquitectura del modelo. Es suficiente tener en cuenta:

- Que existe una red de nodos. Desde un punto de vista lógico, existen dos clases de nodos: **Proveedores de Servicios de Directorio** (PSD) que ofrecen datos según se describe más abajo; y **Clientes** que acceden a los PSD a través de peticiones compartiendo algún tipo de método de acceso.
- Que cada PSD ofrece una colección de elementos, cada uno de los cuales está descrito por una serie de pares `atributo:valor`. A su vez, cada colección está descrita por una serie de características generales. Cada PSD contiene información acerca de las colecciones y los elementos que ofrece, de forma que los clientes u otros agentes (por ejemplo, robots de exploración de la WWW) puedan acceder a descripciones generales de sus contenidos (esquemas).

Tomamos la especificación de RDM (*Resource Description Messages*) propuesta en [Har96] como implementación de esta última idea. RDM define un conjunto de mensajes para intercambiar descripciones de esquemas y de elementos. Estas descripciones de recursos (DR) están contenidas en los PSD. Así, RDM es un mecanismo usado por agentes de WWW o usuarios para descubrir y obtener los recursos de los PSD.
RDM está basado en el formato SOIF (*Summary Object Interchange Format*, [HSW94]) de Harvest para codificar esquemas y DR. RDM también proporciona *esquemas* predefinidos para soportar ciertos tipos de interacción con los PSD. Por ejemplo, RDM define un *es-*

*quema* para especificar preguntas acerca de DR (*¿qué DR han cambiado desde la semana pasada?*) o para describir taxonomías de las colecciones que ofrecen. RDM es lo que llamamos un **mecanismo de descubrimiento de recursos**. La aproximación terminológica a los mecanismos de descubrimiento de recursos permite básicamente un enriquecimiento de la semántica de la que deben favorecerse los usuarios de los servicios.

# 3 Sistemas Terminológicos de Representación del Conocimiento

Los Sistemas Terminológicos de Representación del Conocimiento (STRC) organizan y representan el conocimiento por medio de taxonomías e incorporan mecanismos de inferencia específicos a dichas estructuras. Los STRC hacen una clara distinción entre el conocimiento intensional y el extensional. El primero denota el conocimiento global acerca de un dominio determinado y se le suele denominar **TBox** (*Terminological Box*) o Módulo Definicional. Un TBox es un conjunto de axiomas terminológicos en los que se utilizan las expresiones de un lenguaje conceptual (LC) para construir una taxonomía. El segundo denota el conocimiento particular a una situación determinada dentro de un dominio, se le suele denominar **ABox** o Módulo Asercional. Este módulo está formado por axiomas que usan los conceptos introducidos en el TBox. Los individuales introducidos en el ABox son *clasificados* respecto a la terminología previamente definida. Pero, al contrario que los sistemas de programación lógica, estos sistemas permiten también razonar únicamente con la información intensional. Las inferencias típicas son las de subsumisión, satisfacibilidad, equivalencia y disyunción; incluso en los lenguajes de conceptos con complementos las tres últimas inferencias pueden reducirse a la primera ([DLNN95], [SSS91]). Además, los STRC deben disponer de una interfaz para interrogar y explotar el conocimiento almacenado e inferido en el sistema. Un *lenguaje inferencial* suministra las facilidades necesarias para interrogar al sistema. El lenguaje conceptual es el núcleo de un STRC y su expresividad y propiedades computacionales determinan la eficiencia y utilidad práctica del mismo.
Para un STRC el mundo está compuesto por un conjunto de *individuales*. Un *concepto* es un subconjunto de individuales y un *rol* es un subconjunto de pares de individuales (relación binaria). Un STRC determinado está compuesto por un *TBox* y por un *ABox*. El TBox contiene las definiciones de conceptos y roles (terminología) y una taxonomía establecida por las relaciones de subsumisión entre ellos. El ABox contiene el conjunto de individuales y relaciones que forman parte de una determinada percepción del mundo. Un lenguaje asercional permite crear individuales y relaciones entre individuales que son ocurrencias de los conceptos y roles definidos en el TBox.

**Lenguajes Conceptuales** En los lenguajes conceptuales se parte de conceptos atómicos y roles atómicos, y se usa el lenguaje para crear nuevos conceptos y roles. Los conceptos se consideran como predicados unarios y los roles como predicados binarios y la semántica tarskiana los interpreta como conjuntos de individuales o relaciones binarias entre individuales respectivamente. Los distintos lenguajes difieren en los constructores usados en la construcción de expresiones. Para introducir la sintaxis y semántica de los lenguajes conceptuales o terminológicos nos serviremos de la familia de lenguajes $\mathcal{AL}$, notación usada en [DLNN95] y que procede del lenguaje $\mathcal{ALC}$ introducido en [SSS91].

**Sintaxis** Sea **A** un conjunto conceptos atómicos y **P** un conjunto de roles atómicos. En el lenguaje $\mathcal{AL}$ los conceptos ($C$, $D$) y roles (Q, R) se construyen inductivamente a partir de los conceptos y roles atómicos y de los conceptos universal y vacío:
(1) Un elemento de **A**, $A$, es un concepto y un elemento de **P**, $P$, es un rol (conceptos y roles atómicos)
(2) $\top$ (concepto universal)
(3) $\bot$ (concepto vacío)
(4) $\neg A$ (negación de un concepto atómico)
(5) $C \sqcap D$ (concepto conjunción)
(6) $\forall P.C$ (concepto cuantificación universal de rol)
(7) $\exists P.\top$ (concepto cuantificación existencial no cualificada)
Las fórmulas creadas segun esta sintaxis reciben el nombre de *expresiones de conceptos*. Obsérvese que en $\mathcal{AL}$ no hay constructores de roles, por tanto todos los roles son atómicos. La extensión del lenguaje $\mathcal{AL}$ con uno o un subconjunto de los constructores siguientes da lugar al lenguaje $\mathcal{AL}[\mathcal{U}][\mathcal{E}][\mathcal{C}][\mathcal{N}][\mathcal{R}]$:
(8) $C \sqcup D$ (Extensión $\mathcal{U}$, unión de conceptos).
(9) $\exists R.C$ (Extensión $\mathcal{E}$, cuantificación existencial cualificada).
(10) $\neg C$ (Extensión $\mathcal{C}$, complemento de conceptos no atómicos ).
(11) $\geq nR$, $\leq nR$ (Extensión $\mathcal{N}$, restricciones de número).
(12) $Q \sqcap R$ (Extensión $\mathcal{R}$, rol intersección de roles).
Otros constructores de interés son:
(13) $C \times D$ (rol producto)
(14) $\leq nR.C, \geq nR.C$ (restricciones de número cualificadas)
(15) $fills.i.R$ (rellena)
A pesar de que no es estrictamente necesario ya que se derivan de las restricciones de número cualificadas, usaremos también los constructores:
(16) $nR$ (restricción exacta de número)
(17) $nRC$ (restricción exacta de número cualificada).

**Semántica** El significado formal del lenguaje está dado por una interpretación de teoría de modelos $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$. La interpretación consiste en un conjunto arbitrario $\Delta^{\mathcal{I}}$ (el dominio de la interpretación) y una función de interpretación $\cdot^{\mathcal{I}}$ que proyecta cada concepto $A$ en un subconjunto de $\Delta^{\mathcal{I}}$ ($\mathrm{A}^{\mathcal{I}}$) y cada rol $P$ en un subconjunto de $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ ($\mathrm{P}^{\mathcal{I}}$). Los conceptos predefinidos $\top$ y $\bot$ tienen una interpretación fija, $\Delta^{\mathcal{I}}$ y $\emptyset$ respectivamente. El significado de las expresiones de conceptos y roles viene dado por:
$(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
$(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$
$(\forall R.C)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} \mid \forall b.(a,b) \in R^{\mathcal{I}} \longrightarrow b \in C^{\mathcal{I}}\}$
$(\exists R.\bot)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} \mid \exists b.(a,b) \in R^{\mathcal{I}}\}$
$(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$
$(\exists R.C)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} \mid \exists b.(a,b) \in R^{\mathcal{I}} \wedge b \in C^{\mathcal{I}}\}$
$(nR)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} \mid card\{b \mid (a,b) \in R^{\mathcal{I}}\} = n\}$
$(\geq nR)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} \mid card\{b \mid (a,b) \in R^{\mathcal{I}}\} \geq n\}$
$(\leq nR)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} \mid card\{b \mid (a,b) \in R^{\mathcal{I}}\} \leq n\}$
$(Q \sqcap R)^{\mathcal{I}} = Q^{\mathcal{I}} \cap R^{\mathcal{I}}$
$(C \times D)^{\mathcal{I}} = \{(a,b) \in \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \mid a \in C^{\mathcal{I}} \wedge b \in D^{\mathcal{I}}\}$

$(\leq nR.C)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} \mid card\{b \mid (a,b) \in R^{\mathcal{I}} \wedge b \in C^{\mathcal{I}}\} \leq n\}$
$(\geq nR.C)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} \mid card\{b \mid (a,b) \in R^{\mathcal{I}} \wedge b \in C^{\mathcal{I}}\} \geq n\}$
$(nR.C)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} \mid card\{b \mid (a,b) \in R^{\mathcal{I}} \wedge b \in C^{\mathcal{I}}\} = n\}$
$(fills.i.R)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} \mid i \in \Delta^{\mathcal{I}} \wedge (a,i) \in R^{\mathcal{I}}\}$
La semántica garantiza las equivalencias $C \sqcup D \equiv \neg(\neg C \sqcap \neg D)$ y $\exists R.C \equiv \neg\forall R.\neg C$. Por esta razón se supone que cualquier lenguaje $\mathcal{AL}$ con negación de conceptos posee también los constructores unión y cuantificación existencial cualificada y en los nombres de los lenguajes se usa $\mathcal{C}$ en vez de $\mathcal{UE}$, así por ejemplo $\mathcal{ALCN} \equiv \mathcal{ALUEN}$. Con respecto a los primeros lenguajes conceptuales $\mathcal{FL}$ y $\mathcal{FL}^-$ [LB85] puede verse en [SSS91] que son sublenguajes de $\mathcal{ALC}$ y $\mathcal{AL}$ respectivamente.

**Un lenguaje para el TBox** Si $D$ y $C$ son expresiones de conceptos y $S$ es una expresión de rol, un TBox o *Terminología* es un conjunto finito de *axiomas terminológicos* que definen el concepto $A$ y el rol $R$:

- Axiomas terminológicos de conceptos y roles definidos (también llamadas definiciones completas): $A = D, R = S$
- Axiomas terminológicos de conceptos y roles primitivos (también llamadas definiciones incompletas o especializaciones): $A \sqsubseteq D, R \sqsubseteq S$
- Axiomas terminológicos que establecen la disyunción de conceptos: $dis(C,D)$

y con dos restricciones:

- Ningún concepto o rol puede aparecer más de una vez en la parte izquierda de un axioma terminológico.
- El axioma de disyunción no debe contener axiomas definidos.

Obsérvese que si el lenguaje tiene el concepto $\top$, no es necesario distinguir entre conceptos no definidos (sin restricciones en la interpretación) y parcialmente definidos (con condiciones necesarias en la interpretación). Un concepto no definido puede verse como definido parcialmente ($A \sqsubseteq \top$).
Sea $A$ un concepto, $R$ un rol, $D$ una expresión de concepto y $S$ una expresión de rol. Se dice que una interpretación $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ satisface un axioma terminológico, si y solo si:
$A^{\mathcal{I}} = D^{\mathcal{I}}(R^{\mathcal{I}} = S^{\mathcal{I}})$ para el axioma terminológico de la forma $A = D(R = S)$
$A^{\mathcal{I}} \subseteq D^{\mathcal{I}}(R^{\mathcal{I}} \subseteq S^{\mathcal{I}})$ para el axioma terminológico de la forma $A \sqsubseteq D(R \sqsubseteq S)$
$A^{\mathcal{I}} \cap B^{\mathcal{I}} = \emptyset$ para el axioma terminológico de la forma $dis(A,B)$
Ahora estamos en condiciones de definir un *modelo*. Una interpretacion $\mathcal{I}$ es un modelo de un TBox si satisface todos los axiomas terminológicos.

**Un lenguaje para el ABox** Un ABox contiene los individuales y relaciones que forman parte del mundo definido en la TBox. Si $C$ es un nombre de concepto, $R$ es un nombre de rol, y $a$ y $b$ son nombres de individuales entonces los siguientes son axiomas asercionales.
$(a.C)$
$(a.b.R)$
Un ABox es un conjunto finito de axiomas asercionales. La función de interpretación $\cdot^{\mathcal{I}}$ se extiende a los nombres de los individuales y roles proyectándolos sobre $\Delta^{\mathcal{I}}$ y $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ respectivamente. Una interpretación $\mathcal{I}$ satisface un axioma asercional, si y sólo si:
$a^{\mathcal{I}} \in C^{\mathcal{I}}$ para el axioma asercional de la forma $(a.C)$
$(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$ para el axioma asercional de la forma $(a.b.R)$

Una interpretacion $\mathcal{I}$ es un modelo de un ABox si satisface todos los axiomas asercionales. Es interesante observar que los sistemas terminológicos suelen incluir la *suposición de nombres únicos* pero no la de *mundo cerrado.*

# 4 Un Modelo Terminológico de los Mecanismos de Descubrimiento de Recursos

Un modelo terminológico para un mecanismo de descubrimiento de recursos debe incluir:

- Axiomas terminológicos con conocimiento general acerca de los recursos.
- Axiomas asercionales con conocimiento particular sobre las ocurrencias de esos recursos.

La propuesta de un modelo terminológico es un paso previo a la traslación automática del mecanismo a un lenguaje conceptual y en el marco de un modelo propuesto pueden realizarse translaciones de menor o mayor contenido semántico. En el caso de RDM/SOIF empezaremos considerando la definición de un *esquema* según [Har96], en la que hay información adicional sobre el esquema mismo (`Schema-name`[1], nombre del esquema; `URL`, dónde encontrar el esquema; `LastModified`, última modificación del esquema) y la definición de los atributos que describen a los objetos que definen el esquema. La formalización en LD la realizamos por tanto en tres etapas: atributos, esquema y objetos.

Un *esquema* declara un conjunto de *atributos* que contiene cada objeto definido por ese esquema. Además, para cada atributo se definen una serie de características generales que determinan su interpretación externa, como por ejemplo, si el atributo es un índice, el tipo de dato que contiene, longitud, etc. Por ejemplo, si un atributo es definido en un esquema como índice, todos los objetos definidos según ese esquema tendrán a ese atributo como índice. Este es una idea similar a las variables estáticas de una clase de objetos en Programación Orientada a Objetos. Este es un ejemplo de un esquema en RDM/SOIF con dos atributos, el segundo de los cuales es un índice:

```
@SCHEMA{ -
  Schema-Name:              Paper
  URL:                      http://someplace
  LastModified:             12/2/2003

  SOIF-Attribute-1{x}:      Title
  Description-1{x}:         Contains the title
  Data-Type-1{x}:           String
  Enforce-Uniqueness-1{x}:  Yes

  SOIF-Attribute-2{x}:      Year
  Description-2{x}:         A number identifying the paper
  Data-Type-2{x}:           Number
  Index-Attribute-2{x}:     1
}
```

[1]En esta sección hemos mantenido los nombres de los ejemplos en inglés para facilitar la lectura. Esta fuente se usa en todos los elementos de la traslación terminológica y `esta fuente` en los de RDM/SOIF

**Atributos** En LD, típicamente, los atributos se modelan como relaciones entre individuales. Así, un objeto con un conjunto de atributos es un individual con un conjunto de roles. Los roles se rellenan con valores que a su vez son individuales. En muchas ocasiones estos valores son ocurrencias de conceptos tales como String, Integer, etc. La mayoría de los lenguajes de conceptos tienen estos conceptos predefinidos (conceptos huéspedes). Por ejemplo, para un esquema `Paper` que define los objetos que tienen un `Title:STRING` y un `Year:INTEGER`, necesitamos en un SRTC los siguientes axiomas terminológicos.
$\mathsf{Integer} \sqsubseteq \top$
$\mathsf{String} \sqsubseteq \top$
$\mathsf{Title} \sqsubseteq \top$
$\mathsf{Paper} \sqsubseteq \top$
$\mathsf{hasTitle} = \mathsf{Paper} \times \mathsf{Title}$
$\mathsf{hasYear} = \mathsf{Paper} \times \mathsf{Integer}$
Ya que un mecanismo de descubrimiento de recursos (como RDM) supone los objetos descritos con atributos y valores, una parte importante del modelo terminológico es el modelado de atributos. Si vemos un atributo como un objeto con propiedades (su descripción, valores, etc..) un atributo se convierte en un individual y un conjunto de atributos se convierte en un concepto Attribute. Esta conceptualización permite aprovechar el tratamiento de conceptos propio de los sistemas terminológicos: especializar y generalizar un concepto atributo, saber si un atributo subsume a otro, clasificar un individual atributo, etc.
La conceptualización de atributos permite modelar las características generales de un atributo (descripción, si es índice, longitud, etc.). Supóngase que queremos introducir una descripción del atributo hasTitle. Esta descripción será el String "ContainstheTitle". Necesitamos un concepto Title (conceptualización del atributo) y un rol Description. Este rol sirve para describir la propiedad "descripción" del concepto Title. Con el constructor numerado como (15) en la sección 3 podemos indicar que este rol se rellena con un individual particular (el individual "ContainstheTitle", que es una ocurrencia del concepto String).
Así, en un STRC podemos modelar atributos con axiomas terminológicos usando el constructor fills. Es importante observar cómo con este constructor, un axioma terminológico puede depender de un axioma asercional. Obsérvese que uso del constructor fills hace que la existencia de conceptos exija la existencia de individuales.
Se define primeramente un concepto Attribute, un concepto Schema, y un rol Value, mediante los siguientes axiomas terminológicos:
$\mathsf{Schema} \sqsubseteq \top$
$\mathsf{Attribute} \sqsubseteq \top$
$\mathsf{Value} \sqsubseteq \top \times \top$
Y luego una serie de roles para definir las características globales del atributo (aquellas que comparten todas sus casos):
$\mathsf{Description} = \mathsf{Attribute} \times \mathsf{String}$
$\mathsf{DataType} = \mathsf{Attribute} \times \mathsf{String}$
$\mathsf{ContentType} = \mathsf{Attribute} \times \mathsf{String}$
$\mathsf{EnforceUniqueness} = \mathsf{Attribute} \times \mathsf{Boolean}$
$\mathsf{IndexAttribute} = \mathsf{Attribute} \times \mathsf{Integer}$
$\mathsf{IsInternal} = \mathsf{Attribute} \times \mathsf{Boolean}$
A continuación, por cada atributo SOIF creamos un concepto subsumido en el concepto Attribute y creamos también individuales necesarios para rellenar los roles de características

asociados al concepto. La definición del atributo también incluye que este es univaluado. Suponiendo un esquema como el mostrado en el inicio de esta sección pero que no tuviese el atributo índice, se crean los siguientes individuales mediante axiomas asercionales:
(“ContainstheTitle”.String)
(True.Boolean)
también se define el concepto Title rellenando los roles de características del atributo con los individuales correspondientes, mediante el axioma terminológico siguiente:
Title $=$ Attribute $\sqcap$ fills.“Contains the Title”.Description $\sqcap$ fills.True.EnforceUniqueness $\sqcap$ 1ValueString
y por último el axioma terminológico que define el rol hasTitle hace uso del concepto Titulo:
hasTitle $=$ Paper $\times$ Title

**Esquemas** Una vez conceptualizados los atributos sólo queda completar la información adicional acerca de cada esquema, creando los roles necesarios. De nuevo volvemos a rellenar las características (roles) de un concepto (el esquema que estamos definiendo) con individuales (los valores con los que se rellenan esas características). Los axiomas terminológicos que necesitamos son:
URL $=$ Schema $\times$ String
LastModified $=$ Schema $\times$ Date

junto con los axiomas asercionales:
(“http://someplace”.String)
(12/02/2003.Date)

y finalmente definimos un Paper mediante otro axioma terminológico, combinando los anteriores.
Paper $=$ schema $\sqcap$ fills.“http://someplace”. URL $\sqcap$ fills.12/02/2003.LastModified $\sqcap$ 1hasTitle
Con esto queda definido el TBox. Observese que hemos necesitado introducir algúnos axiomas asercionales para poder completar la definición.

**Objetos** Ahora estamos en condiciones de introducir los objetos definidos por esa plantilla como casos de conceptos. Por ejemplo, para el siguiente objeto SOIF (que sigue el *esquema* definido en el ejemplo previo)

```
@Paper{ -
  Title:    New Worlds
}
```

se introducirían los siguientes axiomas asercionales (donde p1 y t1 son nombres únicos dados a los individuales):
(t1.Title)
(“New Worlds”.String)
(t1.”New Worlds”.Value)

(p1.Paper)
(p1.t1.hasTitle)

# 5 Ventajas de la Aproximación Terminológica

Desde el punto de vista del modelo inicial de RDM, nuestra aproximación al mismo enriquece su semántica permitiéndole adquirir las ventajas de cualquier implementación terminológica. Esto es, al expresar las construcciones de RDM en un sistema genérico de conceptos y roles lo que estamos haciendo es:

- Expresar RDM en un lenguaje usado en multitud de dominios. Esto permitiría compartir herramientas y métodos en el tratamiento de la información que contienen.
- Extender la expresividad de las lógicas descripcionales a RDM. De esta manera RDM es fácilmente extensible (adición de nuevos conceptos y roles) independientemente del mecanismo que lo manipule.
- Tratar, desde un punto de vista formal, las ideas introducidas por RDM, lo que promueve su crítica y la unificación de juicios sobre el mismo.

Por otro lado, se realiza una aplicación de las LD a un problema inherentemente distribuido (las terminologías acerca de las descripciones se almacenan de manera aleatoria por todo el mundo). Suponiendo que las descripciones de recursos se almacenan a través de la red como terminologías, aún hacen falta los mecanismos de recuperación de las mismas (protocolo de acceso a conceptualizaciones remotas). Así, podemos vislumbrar una nueva aproximación al tema de las LD distribuidas. Existen algunas aproximaciones en este sentido [MKIS96] así como en el ámbito de las bases de datos [BR93]. Si dicho protocolo es lo suficientemente genérico entonces podría acceder a conceptualizaciones sobre cualquier dominio.

**Agradecimientos** Los autores agradecen el soporte económico de la Xunta de Galicia por medio del proyecto ref. XUGA10503B96.

# Referencias

[BH91] F. Baader and B. Hollunder. KRIS: Knowledge representation and inference system. *SIGART Bulletin*, 2(3):8–14, 1991.

[BR93] A. Borgida and Brachman R. Loading data into description reasoners. In *SIGMOD Conference*, pages 217–226, 1993.

[BRMR89] A. Borgida, Brachman R.J., D.L. McGuiness, and L.A Resnick. CLASSIC: A structural data model for objects. In *Proc. 1989 ACM SIGMOD Conference on Management of Data*, pages 59–67, June 1989.

[BS85] R.J. Brachman and J.G. Schmolze. An overview of the KL-ONE knowledge representation system. *Cognitive Science*, 9(2):171–216, April, 1985.

[DLNN95] Francesco M. Donini, Maurizio Lenzerini, Daniele Nardi, and Werner Nutt. The complexity of conceptual languages. Research Report 95-07, German Research Center for Artificial Intelligence - DFKI GmbH, Saarbrücken, Germany, June 1995.

[Har96] D. Hardy. Resource Description Messages(RDM). Technical report, World Wide Web Consortium, 1996.

[HC96] Manuel Hermenegildo and Daniel Cabeza. The PiLLoW/CIAO library for Internet/WWW programming using computational logic systems. In *Proceedings of the 1st Workshop on Logic Programming Tools for Internet Applications*, Bonn, Germany, September 1996.

[HSW94] Darren R. Hardy, Michel F. Schwartz, and Duane Wessels. Harvest v1.4 user's manual: Effective use of Internet information. Technical report cu-cs-743-94, University of Colorado at Boulder, 1994.

[LB85] H.J. Levesque and R.J. Brachman. A fundamental tradeoff in knowledge representation and reasoning (revised version). In Ronald J. Brachman and Hector J. Levesque, editors, *Readings in knowledge representation*, pages 817–823. Morgan Kauffmann, Los Altos, CA, 1985.

[LDS96] S.W. Loke, A. Davison, and L. Sterling. Lightweight deductive databases on the World-Wide Web. In *Proceedings of the 1st Workshop on Logic Programming Tools for Internet Applications*, pages 91–106, Bonn, Germany, September 1996.

[MB87] R. MacGregor and R. Bates. The LOOM knowledge representation language. Technical Report ISI/RS-87-188, USC/Information Sciences Institute, Marina del Rey, CA, 1987.

[MKIS96] E. Mena, V. Kashyap, A. Illarramendi, and A. Sheth. Managing multiple information sources through ontologies: relationship between vocabulary heterogeneity and loss of information. In *Proc. of Knowledge Representation Meets Databases (KRDB'96) ECAI96 Conference*, pages 50–52, Budapest, Hungary, August, 1996.

[SSS91] M. Schmidt-Schauß and G. Smolka. Attributive concept descriptions with complements. *Artificial Intelligence*, 48(1):1–26, 1991.