published as: S. Linckels, Christoph Meinel: A Simple Application of Description Logics for a Semantic Search Engine;
Proceedings of the IADIS International Conference on Applied Computing (AC 2005);
Lisbon, Portugal, Vol. II, pp. 306-311.

# A SIMPLE APPLICATION OF DESCRIPTION LOGICS FOR A SEMANTIC SEARCH ENGINE

Serge Linckels
*Hasso-Plattner-Institut für Softwaresystemtechnik, University of Potsdam*
*Postfach 900460, D-14440 Potsdam*
*linckels@hpi.uni-potsdam.de*

Christoph Meinel
*Hasso-Plattner-Institut für Softwaresystemtechnik, University of Potsdam*
*Postfach 900460, D-14440 Potsdam*
*meinel@hpi.uni-potsdam.de*

**ABSTRACT**

In this paper we present a simple application of Description Logics in order to improve the semantic search engine of a multimedia e-Learning tool. CHESt (*Computer History Expert System*) allows students to enter a freely formulated question about computer history. The system returns a very short commented list of multimedia clips in which the user finds the answer to his question. Finding the semantically pertinent clip(s) is the challenge. In this paper we illustrate how the semantic reasoning process can be improved by using Description Logics as a formal representation language for specifying documents and queries.

**KEYWORDS**

Description Logics, semantics, inference, search engine, e-Learning.

## 1. INTRODUCTION

We presented in [12] our new e-Learning tool CHESt (*Computer History Expert System*) that understands students' questions. Details about the pedagogical background of this e-Learning tool can be found in [11]. The tool disposes of a knowledge base with 300 multimedia *clips* that cover the main events in computer history. The user enters a question by means of natural language (NL) and the system returns a list of appropriated clips as an answer. Because the multimedia clips are recorded and stored as *RealMedia* files, their content is not available as text. However, to allow the search engine to understand the meaning of the documents, it is useful to describe the knowledge base with metadata [2]. In the former version of CHESt, we used RDF(S) to describe the semantics of the multimedia clips.

Here we will briefly recapitulate how the former version of CHESt works. The semantic search engine gets a NL question from the user and maps it to a general assertion. To do this, it firstly searches for semantically important words and translates them into RDF. We use a specific domain dictionary to retrieve the semantics for every word in the sentence. For example, the question "Who invented the operating system CP/M?" would be transformed into a set of well-known-words

$$\Phi = \{(\texttt{dc:creator};\text{"invented"}),(\texttt{chest:OS};\text{"CP/M"})\}.$$

Semantically unimportant words like {what, did} or too general words like {operating, system} are ignored. Secondly, this transformed question is mapped to a general assertion. The set of general assertions is given to the system, and generally contains only few elements. In the above example, the system would map the question to the general assertion "An invention was invented by an inventor", because of the predicate `dc:creator`. Based on that interpretation, an RDQL [13] query is generated and launched against the knowledge base. In the example, the query would look like this:

In Proceedings of the IADIS International Conference of Applied Computing 2005 (IADIS AC2005), 22-25 February 2005, Lisbon, Portugal, Vol. II, pp. 306-311.

```
SELECT WHERE (?x;dc:creator;"CP/M")
```

where `?x` is the missing part of the query. A commented list of pertinent clips is returned to the user.

Unfortunately, more advanced and complex reasoning is not possible in RDF, mainly due to the weak possibilities in expressing properties and rules over properties. In this paper, we report main improvements in the inference engine of our tool by using OWL (*Web Ontology Language*) instead of RDF(S). In section 2 we will show how the knowledge is represented formally with DLs. In section 3 we will present the reasoning process, which is the main advantage of this change. We will conclude in section 4 with some (dis)advantages of the proposed solution.

# 2. KNOWLEDGE REPRESENTATION FOR CHEST

In this section we will briefly introduce the main concepts for representing knowledge and reasoning about it by using DLs. The fact that OWL builds on RDF(S) simplifies the serialization task (which is not covered in this paper).

## 2.1 Description Logics Preliminaries

Description Logics is a formal language for representing knowledge and reasoning about it [1]. In DLs, the conceptual knowledge of an application domain is represented in terms of *concepts* (unary predicates) that are interpreted as sets of individuals, and *roles* (binary predicates) that are interpreted as binary relations between individuals. The semantics of a concept description is defined by the notion of interpretations as given below.

**Definition 1 (Interpretation):** An interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ *consists of a non-empty set* $\Delta^{\mathcal{I}}$*, the domain of the interpretation, and an interpretation function* $\cdot^{\mathcal{I}}$ *that maps each concept name to a subset of* $\Delta^{\mathcal{I}}$ *and each role name to a binary relation* $r^{\mathcal{I}}$*, subset of* $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$.

A typical DL knowledge base comprises two components: a terminology, also called a *TBox*, and assertions, also called an *ABox*. Both are described in this section.

**Definition 2 (Knowledge Base):** *A knowledge base (KB) is a pair* $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$*, where* $\mathcal{T}$ *is a TBox, and* $\mathcal{A}$ *is an Abox.*

## 2.1 Knowledge Terminology in a TBox

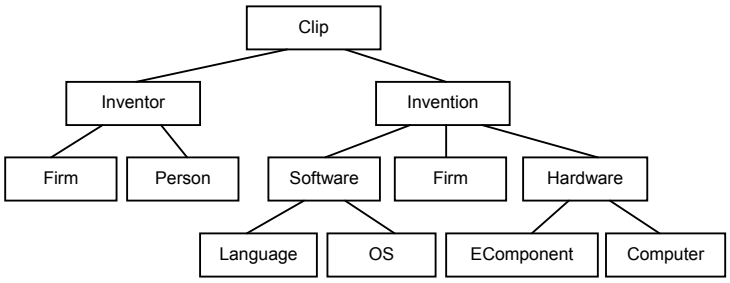
The TBox defines the vocabulary to use in the KB by terms of concepts and roles. The concepts are either defined as new concepts or by using previously defined concepts. The resulting terminologies can easily be serialized as OWL.

**Definition 3 (Terminology):** *Let A be a concept name and C a concept definition. Then* $A \doteq C$ *and* $A \sqsubseteq C$ *are terminological axioms. The first is a complete definition, the second an incomplete one. A terminology* $\mathcal{T}$ *is a finite set of terminological axioms such that no concept name appears more than once in the left-hand side of a definition. If a concept A occurs in the left-hand side of a definition, it is called defined concept. The other concepts are called primitive concepts.*

The concepts in CHESt are organized in a taxonomy. Figure 1 illustrates the translation of the hierarchy of concepts (HC) into the acyclic $\mathcal{ALC}$-concept description $\mathcal{T}_{\text{CHESt}}$. A special case in our taxonomy is the concept Firm, which can be an inventor (something was invented by that firm) or an invention (a firm was founded by an inventor). The language $\mathcal{ALC}$ [15] is sufficiently expressive for our purposes. It is in fact a subset of the logics implemented in most "state of the art" DL systems, for example those based on highly optimized tableaux algorithms, see for example [8, 6]. $\mathcal{ALC}$ concepts are built using a set of concept names (NC) and role names (NR). Valid concepts are defined by the following syntax:

$$C ::= A \mid \top \mid \bot \mid \neg A \mid C_1 \sqcap C_2 \mid C_1 \sqcup C_2 \mid \forall R.C \mid \exists R.C$$

with $A \in$ NC is a concept name and $R \in$ NR is a role name.




Person ≐ ∃hasName.String ⊓ ∃wasBorn.Date ⊓ ∃isDeceased.Date
Clip ≐ Inventor ⊔ Invention
Inventor ≐ Person ⊔ Firm
Invention ≐ Firm ⊔ Software ⊔ Hardware ⊓ ∃hasTitle.String ⊓ ∃wasCreated.Date ⊓ ∃hasInventor.Inventor
Software ≐ Language ⊔ OS
Hardware ≐ EComponent ⊔ Computer
Computer ≐ hasComponent.EComponent

Figure 1. Example of a hierarchy of concepts and the according $\mathcal{ALC}$-terminology for $\mathcal{T}_{\text{CHESt}}$.

## 2.1 Knowledge Assertions in an ABox

In the ABox, one introduces individuals, by giving them names, and one establishes properties for these individuals. Figure 2 shows some examples of CHESt assertions translated into DLs, noted $\mathcal{A}_{\text{CHESt}}$.

**Definition 4 (Model of Assertions):** *The interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ satisfies the concept assertion C(a) if $a^{\mathcal{I}} \in C^{\mathcal{I}}$, and it satisfies the role assertion R(a,b) if $(a,b)^{\mathcal{I}} \in R^{\mathcal{I}}$. An interpretation $\mathcal{I}$ satisfies the ABox $\mathcal{A}$ if it satisfies each assertion in A. In this case we say that $\mathcal{I}$ is a model of the ABox.*

| | | |
|---|---|---|
| Person(Kildall) | Firm(DR) | OS(CPM) |
| String("Gary Kildall") | String("Digital Research) | String("CP/M") |
| Date(1942) | Date(1973) | Date(1974) |
| Date(1994) | hasTitle(DR,"Digital Research") | hasTitle(CPM,"CP/M") |
| hasName(Kildall,"Gary Kildall") | wasCreated(DR,1973) | wasCreated(CPM,1974) |
| wasBorn(Kildall,1942) | hasInventor(DR,Kildall) | hasInventor(CPM,DR) |
| isDeceased(Kildall,1994) | | |

Figure 2. Examples of concept assertions for $\mathcal{A}_{\text{CHESt}}$. The person Gary Kildall (1942-1994) founded the firm Digital Research in 1973, which has published the operating system CP/M in 1974.

# 3. INTERPRETING A USER QUESTION

A DL system not only stores terminologies and assertions, but also offers services that reason about them, called logical inference. This allows to make explicit some implicit knowledge that is contained in the KB. We first present the main kinds of reasoning performed in DLs before showing their application in CHESt.

## 3.1 Main Kinds of Reasoning

Reasoning in a DL KB is mainly based on determining subsumption relationships and satisfiability with respect to the axioms in the TBox, and instance checking with respect to the assertions in the ABox. An exhaustive list is described in [1].

**Definition 5 (Subsumption):** *Let C and D be concept names, D subsumes C with respect to $\mathcal{T}$ (noted $\mathcal{T} \models C \sqsubseteq D$) iff $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ for all interpretation $\mathcal{I}$ that satisfies $\mathcal{T}$.*

**Definition 6 (Satisfiability):** *A concept C is satisfiable with respect to $\mathcal{T}$ if there exists a model $\mathcal{I}$ of $\mathcal{T}$ such that $C^{\mathcal{I}}$ is nonempty. In this case we say also that $\mathcal{I}$ is a model of C.*

**Definition 7 (Instance Checking):** *An assertion $\alpha$ (a concept assertion or a role assertion) is entailed by $\mathcal{A}$ (written $\mathcal{A} \models \alpha$) if every interpretation that satisfies $\mathcal{A}$, that is, every model of $\mathcal{A}$, also satisfies $\alpha$.*

## 3.2 Reasoning in CHESt

The interpretation of a user question in CHESt is performed in two steps: the mapping of concepts over the TBox, and the transformation of the user question into an ABox query. Both are explained below.

### 3.2.1 Mapping of Concepts

In [9] a matching algorithm is proposed. It is not the topic of this short paper to explain this algorithm in detail. However, for a better understanding of its use in our e-Learning tool, we will summarize its mechanism briefly. The algorithm takes a query description $\mathcal{T}_Q$ and a document description $\mathcal{T}_D$ and returns a mapping that identifies corresponding elements in the two descriptions. This mapping consists of a set of mapping elements indicating that certain concepts of the query are related to certain concepts in the document. A concept $A_i$ from $\mathcal{T}_Q$ is related to a concept $B_i$ from $\mathcal{T}_D$ if their names and their descriptions are similar. The algorithm uses the difference operation presented in [16] and improved in [10]; since the result of the difference operation is a description or a set of descriptions, it can be used for retrieving sets of individuals matching the difference between two descriptions. In other words, only similar documents from the KB are found. The algorithm works in three steps: computing the similarity of concepts (step 1 and 2), which is quantified by two coefficients (the name and the description coefficient), and mapping similar concepts (step 3).

**Step 1: Matching of names.** It is based on the notion of semantic relatedness that measures the extent to which two lexicalized concepts are close. This measure is based on the semantic relations of a knowledge source, for example WordNet. This result is called the *name similarity coefficient (nsim)*.

**Step 2: Matching of description.** It consists in comparing the concept descriptions occurring in the two terminologies. This phase uses name similarities between concepts appearing in the concept descriptions. This result is called the *description similarity coefficient (dsim)*.

**Step 3: Mapping of concepts.** The resulting weighted similarity (*wsim*) is a mean of *nsim* and *dsim*. A mapping $\rho$ is deduced from those coefficients by choosing pairs of elements with maximal similarity. A mapping $\rho$ from $\mathcal{T}_Q$ to $\mathcal{T}_D$ is computed as follows:

$\rho(Q_i) = D_j$ with $1 \le i \le n$, $1 \le j \le m$,
if $wsim(Q_i, D_j) > \varepsilon$ and $wsim(Q_i, D_j) > wsim(Q_i, D_k)$ for all $D_k \in \mathcal{T}_D$ , $k \neq j$
where $\varepsilon$ is the minimal tolerated difference.
$\rho(Q_i) = \top$ otherwise.

We had to modify this algorithm in two points in order to use it in CHESt.

- The reasoning mechanism must be improved in order to perform a query over a non-empty ABox.
- It must not consider the documents content, but the metadata, which describe the document. As already stated in the introduction, this is necessary because we are dealing with multimedia clips where a textual content is not available.

### 3.2.2 The Generating of a Semantic Query

We start from the assumption that all documents in the KB $\mathcal{K}_{\text{CHESt}}$ are represented by DL terminologies, but the user's question $Q$ is expressed in NL. Thus, the latter must be transformed into a query w.r.t. the ABox. The so generated query allows to retrieve all documents from the KB that satisfy the expression $R_Q$. This means that it must be checked if there exists at least one model $\mathcal{I}$ of $\mathcal{A}_{\text{CHESt}}$ such $(R_Q)^{\mathcal{I}} \neq \emptyset$. In other words, there must exist an individual $y$ in $\Delta^{\mathcal{I}}$ that is an element of $(R_Q)^{\mathcal{I}}$. Figure 3 shows an example where the system must find all objects in the KB that are individuals of the concept Inventor and are involved in the

invention of the operating system "CP/M". A model w.r.t. the ABox (see figure 2) is the Firm DR. Technically, the reasoning over the KB and the retrieval of individuals can be performed with most DL reasoners. We experimented with the Java interface of RACER [6], which builds on the OWL-API (http://owl.man.ac.uk). More information about the use of DLs as query language for retrieving sets of individuals matching a description from the KB can be found for example in [4, 5, 7].

| $Q$ = Who invented the operating system CP/M? |
| --- |
| $\mathcal{A}_{\text{CHESt}} \models R_Q$ = OS($x$) ^ hasTitle($x$,"CP/M") ^ hasInventor($x$,$y$?) ^ Inventor($y$?) |

Figure 3. Example of a NL user question $Q$ and the according $\mathcal{ALC}$ query expression w.r.t. the ABox. The variable $y$ is the missing part and should be the result of the query.

## 4. RELATED WORK

We found three very promising projects that have several concepts in common with our method. In [3] the algorithm CTXMATCH is presented, which allows to coordinate hierarchies of concept (HC). Semantic coordination, namely the problem of finding an agreement on the meaning of heterogeneous semantic models, is one of the key issues in the development of the Semantic Web. CTXMATCH is a new algorithm for discovering semantic mappings across hierarchical classifications based on a new approach to semantic coordination. This approach shifts the problem of semantic coordination from the problem of computing linguistic or structural similarities (what most other proposed approaches do) to the problem of deducing relations between sets of logical formulae that represent the meaning of concepts belonging to different models. The authors show how to apply the approach and the algorithm to an interesting family of semantic models, namely hierarchical classifications, and present the results of preliminary tests on two types of hierarchical classifications, web directories and catalogs.

In [14] a similar but maybe more specific project was presented, stating that the need for Natural Language Interfaces to databases (NLIs) has become increasingly acute as more and more people access information through their web browsers, PDAs, and cell phones. Yet NLIs are only usable if they map natural language questions to SQL queries correctly. People are unwilling to trade reliable and predictable user interfaces for intelligent but unreliable ones. With their work, the authors introduce a theoretical framework for reliable NLIs, which is the foundation for the fully implemented PRECISE NLI. They prove that, for a broad class of semantically tractable natural language questions, PRECISE is guaranteed to map each question to the corresponding SQL query. They report on experiments testing PRECISE on several hundred questions drawn from user studies over three benchmark data-bases. They find that over 80% of the questions are semantically tractable questions, which PRECISE answers correctly. PRECISE automatically recognizes the 20% of questions that it cannot handle, and requests a paraphrase. Finally, they show that PRECISE compares favorably with Mooney's learning NLI and with Microsoft's English Query product.

A larger system is the KIM Platform (http://www.ontotext.com), which provides a novel Knowledge and Information Management (KIM) infrastructure and services for automatic semantic annotation, indexing, and retrieval of unstructured and semi-structured content. The most direct applications of KIM are:

- Generation of meta-data for the Semantic Web, which allows hyper-linking and advanced visualization and navigation;
- Knowledge Management, enhancing the efficiency of the existing indexing, retrieval, classification and filtering applications.

As a base line, KIM analyzes texts and recognizes references to entities (like persons, organizations, locations, dates). Then it tries to match the reference with a known entity, having a unique URI and description. Alternatively, a new URI and description are automatically generated. Finally, the reference in the document gets annotated with the URI of the entity (semantic annotation). This sort of meta-data can be used for indexing, retrieval, visualization and automatic hyper-linking of documents. In order to allow the easy bootstrapping of applications, KIM is equipped with an upper-level ontology (KIMO) of about 250 classes and 100 properties. Further, a knowledge base (KIM KB), pre-populated with about 200 000 entity descriptions, is bundled with KIM. Its role is to provide as background knowledge (resembling a human's

common culture) a quasi-exhaustive coverage of the entities of general importance - those, which are considered well-known and because of this, typically, not introduced in the documents.

## 5. CONCLUSION

In this paper we have presented a simple application of DLs to improve the semantic search mechanism of a multimedia e-Learning tool. The advantages of this upgrade are firstly, that the serialization in OWL is still compatible with the former RDF(S) description. Secondly, the inference engine allows logical reasoning tasks that go beyond the heuristics of the earlier system. Thirdly, queries can be classified with respect to each other into a *subsumption hierarchy*. It is very useful to have the user questions organized so that the results of previous related queries can be reviewed, for example to implement a kind of learning mechanism. Unfortunately, one of the main problems of the solution presented here is that the matching algorithm mentioned in section 3.2 was created for being used with WordNet as knowledge source. We think that a large-scale dictionary like WordNet is not the best possible solution for a domain ontology about computer history. First of all, different meanings for the same word are possible. Hence, our information retrieval system must set the different interpretations in a context to find the best match. Secondly, large-scale dictionaries often lack specific domain expressions. For these reasons we propose either to use an existing domain specific dictionary or to create a dictionary of its own.

## REFERENCES


[1] Baader F. et al, 2003, *The Description Logic Handbook: Theory Implementation and Applications*. Cambridge University Press, Cambridge, UK.

[2] Baeza-Yates R., Ribeiro-Neto B., 1999, *Modern Information Retrieval*, Addison-Wesley, USA.

[3] Bouquet P.,Serafini L., and Zanobini S., 2003, Semantic coordination: a new approach and an application. *Proceedings Second International Semantic Web Conference (ISWC2003)*, Sanibel Island, Florida, USA, pp. 130-145.

[4] Buchheit M. et al., 1994, Subsumption between Queries to Object-Oriented Databases. *Information Systems, Special issue on extending database technology 19(1)*, pp. 33-54.

[5] Donini F.M. et al., 1998, AL-log: Integrating Datalog and Description Logics. *Journal of Intelligent Information Systems*,Vol. 10.

[6] Haarslev V., Möller R., 1999, RACE System Description, *Proceedings DL99*, Linköping, Sweden, pp. 130-132.

[7] Horrocks I., Tessaris S, 2000, A conjunctive query language for description logic aboxes. *Proceedings of AAAI2000*, Austin, Texas, USA, pp. 399-404.

[8] Horrocks I., Patel-Schneider P., 1998, *FaCT and DLP*, Lecture Notes in Artificial Intelligence No. 1397, Springer, Berlin, pp. 27–30.

[9] Karam N. et al, 2004, Semantic Matching of Natural Language Web Queries. *Proceedings of ICWE2004*, Munich, Germany, pp. 416-429.

[10] Küsters R., 2000, *Non-Standard Inferences in Description Logics*. Springer Lecture Notes in Artificial Intelligence LNAI 2100, Germany.

[11] Linckels S., Meinel Ch., 2004, An Educational Tool that Understands Students' Questions. *Proceedings of AECT/All That Jazz Conference*, Chicago, Illinois, USA.

[12] Linckels S., Meinel Ch., 2005, A Simple Application for an Intelligent Librarian System. *Proceedings of IADIS AC2005*, Lisbon, Portugal.

[13] Miller L. et al, 2002, Three Implementations of SquishQL, a Simple RDF Query Language. *Proceedings of ISWC2002*, Sardinia, Italy.

[14] Popescu A.-M., Etzioni O., Kautz H, 2003, Towards a theory of natural language interfaces to databases. *Proc. ACM IUI*, Miami, FL, USA.

[15] Schmidt-Schauss M., Smolka G., 1991, Attributive concept descriptions with complements. *Journal of Articial Intelligence*, 48(1), pp. 1-26.

[16] Teege G., 1994, Making the Difference: A Subtraction Operation for Description Logics. *Proceedings KR94*, Bonn, Germany, pp. 540-550.