Description Logic Systems with Concrete Domains: Applications for the Semantic Web

Volker Haarslev[†] and Ralf Möller[‡]

[†]Concordia University, Montreal
[‡]University of Applied Sciences, Wedel

Abstract

The Semantic Web initiative defines important challenges for knowledge representation and database systems. Recently, several standards for representation languages have been proposed (RDF, DAML+OIL, OWL). We briefly discuss the logical basis of these representation languages by referring to description logic (DL) inferences systems. By introducing the DL inference system Racer we demonstrate that practically usable query engines for Semantic Web representation languages are available. Then, we argue that current representation languages for the Semantic Web are not sufficient for simple and well-defined representation problems that naturally arise in the context of Semantic Web applications. In particular, we mention different kinds of algebraic constraints over various domains such as the reals or the natural numbers. We report on practical experiences with DL reasoning systems (e.g., Racer) already supporting these representation means and argue for extensions to DAML+OIL or OWL.

1 Introduction

The Semantic Web initiative defines important challenges for knowledge representation and database systems. Recently, one of the main standards for the Semantic Web has been proposed: the Resource Description Format (RDF [15]). Since RDF is based on XML it shares its document-oriented view of grouping sets of declarations or statements. With RDF's triple-oriented style of data modeling, it provides means for expressing graph-structured data over multiple documents (whereas XML can only express graph structures within a specific document). As a design decision, RDF can talk about everything. Hence, in principle, statements in documents can also be referred to as resources. In particular, conceptual domain models can be represented as RDF resources. Conceptual domain models are referred to as "vocabularies" or "ontologies". Specific languages are provided for defining ontologies. An extension of RDF for defining ontologies is RDF Schema (RDFS [6]) which only can express conceptual modeling notions such as generalization between classes (aka concepts) and properties (aka roles). For properties, domain and range restrictions can be specified. Thus, the expressiveness of RDFS is very limited. Much more expressive representation languages are DAML+OIL [17] and OWL [16].

In recent research efforts, these languages are mainly considered as ontology representation languages (see e.g. [2] for an overview). The languages are used for defining classes of so-called abstract objects. In this paper we argue that OWL can also be practically used for representing information about specific abstract objects of a certain domain as well because OWL is based on RDF. All information about specific objects (or entities) refers to an ontology (expressed in DAML+OIL or OWL). Thus, in contrast to, for instance, simple relational databases, queries for retrieving abstract objects described in RDF documents have to be answered w.r.t. to a conceptual domain model (the ontology). In this paper we demonstrate the main query answering services of the OWL inference system Racer, and discuss its relevance for practical applications.

It is common sense that dealing with abstract objects is not enough in the context of the Semantic Web. Although still in a very weak way, based on XML-Schema, OWL and DAML+OIL also provide means for dealing with data types known from programming languages. Data types describe sets of so-called data objects (also called data values) and encompass `integer`, `short`, `long`, `boolean`, `string` etc. For a Semantic Web representation language, a semantic characterization for data types might have been more appropriate in our opinion. Thus natural numbers, integers, reals, or complex numbers might have been selected as data types rather than, for example, `long` or `short` because for knowledge representation languages the storage format should not be of topmost concern. In addition, DAML+OIL and OWL do not support so-called constraints between data values. In many practical applications, for instance, linear polynomial inequations with order relations are required for expressing adequate domain models. This paper argues that Semantic Web languages such as DAML+OIL or OWL must be extended to meet practical concerns. In addition, we show that practical systems such as Racer exist that already cope with important representation techniques missing in contemporary languages for the Semantic Web.

2 DL Systems as Query Engines for the Semantic Web

The representation languages mentioned above are defined with a model-theoretic semantics. In particular, for the language OWL, a semantics was defined such that very large fragments of the language can be directly expressed using so-called description logics (DLs, see [1]). The fragment is called OWL DL. With some restrictions that are discussed below one can state that the logical basis of OWL or DAML+OIL can be characterized with the description logic $\mathcal{SHIQ}(\mathcal{D}_n)^-$ [2] (DAML+OIL documents are assumed to be interpreted in the spirit of OWL DL). For those readers not familiar with description logics it suffices to know that $\mathcal{SHIQ}(\mathcal{D}_n)^-$ is a decidable subset of first order predicate logic. Sets of formulae for expressing conceptual domain models (ontologies) are called T-boxes in the tradition of description logics. Sets of formulae for expressing information about specific abstract objects (aka individuals) are called A-boxes. A tuple consisting of a T-box and an A-box is also referred to as a knowledge base (KB).

With some restrictions, DAML+OIL documents can be automatically translated

to $\mathcal{SHIQ}(\mathcal{D}_n)^-$ T-boxes. The restrictions can be summarized as follows. DAML+OIL and OWL also allow for so-called individuals in concepts (and T-boxes). For example, expressing the fact that all humans stem from a single human called ADAM requires an individual in a concept (and a T-box): (implies HUMAN (some has-ancestor {ADAM}). This cannot be expressed in $\mathcal{SHIQ}(\mathcal{D}_n)^-$. However, a straightforward approximation exists (see [9]) such that in practice suitable $\mathcal{SHIQ}(\mathcal{D}_n)^-$ T-boxes can be generated from a DAML+OIL documents. It should be noted that, unfortunately, in a sense OWL is less expressive than its predecessor DAML+OIL because so-called qualifying number restrictions (e.g., for representing concepts such as "at least four related companies which are customers and are also international companies") are missing from OWL 1.0. This will probably be solved in OWL 1.1 since qualified number restrictions are already covered by $\mathcal{SHIQ}(\mathcal{D}_n)^-$ and are very important in particular for querying knowledge bases.

The RDF-Part of DAML+OIL documents can be translated to $\mathcal{SHIQ}(\mathcal{D}_n)^-$ A-boxes. The logic $\mathcal{SHIQ}(\mathcal{D}_n)^-$ is interesting for practical applications because highly optimized inference systems are available (e.g., Racer). Neglecting the well-known restrictions mentioned above, state of the art description logic systems such as Racer [7] can directly read DAML-OIL or OWL documents from local files or from remote web servers and represent the information contained in these documents as knowledge bases. Once the information in DAML+OIL documents is represented as a description logic knowledge base, description logic inference systems such as Racer can be used for practically answering queries (an XML-based query language is described in [4]).

- Concept consistency w.r.t. a T-box: Is the set of objects described by a concept empty?

- Concept subsumption w.r.t. a T-box: Is there a subset relationship between the set of objects described by two concepts?

- Find all inconsistent concepts mentioned in a T-box. Inconsistent concepts might be the result of modeling errors.

- Determine the parents and children of a concept w.r.t. a T-box: The parents of a concept are the most specific concept names mentioned in a T-box which subsume the concept. The children of a concept are the most general concept names mentioned in a T-box that the concept subsumes. Considering all concept names in a T-box the parent (or children) relation defines a graph structure which is often referred to as taxonomy. Note that some authors use the name taxonomy as a synonym for ontology. Computing the taxonomy is called classifying the T-box.

Note that whenever a concept is needed as an argument for a query, not only predefined names are possible. Rather, concepts can be composed to adequately describe the domain objects involved. For details see [1].

If also an A-box is given, among others, the following types of queries are possible:

- Check the consistency of an A-box w.r.t. a T-box: Are the restrictions given in an A-box w.r.t. a T-box too strong, i.e., do they contradict each other? Other queries are only possible w.r.t. consistent A-boxes.
- Instance testing w.r.t. an A-box and a T-box: Is the object for which an individual stands a member of the set of objects described by a certain query concept? The individual is then called an instance of the query concept.
- Instance retrieval w.r.t. an A-box and a T-box: Find all individuals from an A-box such that the objects they stand for can be proven to be a member of a set of objects described by a certain query concept.
- Computation of the direct types of an individual w.r.t. an A-box and a T-box: Find the most specific concept names from a T-box of which a given individual is an instance.
- Computation of the fillers of a role with reference to an individual.

Given the description logic background of Semantic Web representation languages it becomes clear that many application papers[1] demonstrate how these inference services can be used to solve actual problems with DAML+OIL or OWL knowledge bases. In particular, the instance retrieval inference service can be used in the Semantic Web scenario (see [12] for expressivity results). Before we discuss some examples, we consider an extension of DAML+OIL or OWL: constraints on data objects.

3 Concrete Domains

In description logics and data bases, representation techniques for expressing constraints on data objects have a long tradition (see [1, 14]). In the following we will adopt the description logic perspective: concrete domains. Racer supports reasoning over natural numbers ($\mathbb{N}$), integers ($\mathbb{Z}$), reals ($\mathbb{R}$), complex numbers ($\mathbb{C}$), and strings. For different sets, different kinds of predicates are supported.

$\mathbb{N}$	linear inequations with order constraints and integer coefficients
$\mathbb{Z}$	interval constraints
$\mathbb{R}$	linear inequations with order constraints and rational coefficients
$\mathbb{C}$	nonlinear multivariate inequations with integer coefficients
Strings	equality and inequality

3.1 An Example

The following example uses the concrete domains $\mathbb{Z}$ and $\mathbb{R}$. For convenience reasons, rational coefficients can be specified in floating point notation. They are automatically transformed into their rational equivalents (e.g., 0.75 is transformed into 3/4).

[1] See e.g. the workshops on Applications of Description Logics (http://dl.kr.org/adl2001/, http://dl.kr.org/adl2002/).

```
(in-tbox family)
(signature
   :atomic-concepts (... teenager)
   :roles (...)
   :attributes ((integer age)))
...
(equivalent teenager (and human (min age 16)))
(equivalent old-teenager (and human (min age 18)))
...
```

Asking for the children of `teenager` reveals that `old-teenager` is a `teenager`. A further extension demonstrates the usage of reals as a concrete domain.

```
...
(signature
   :atomic-concepts (... teenager)
   :roles (...)
   :attributes ((integer age)
                (real temperature-celsius)
                (real temperature-fahrenheit)))
...
(equivalent teenager (and human (min age 16)))
(equivalent old-teenager (and human (min age 18)))
(equivalent human-with-fever
            (and human (>= temperature-celsius 38.5))
(equivalent seriously-ill-human
            (and human (>= temperature-celsius 42.0)))
...
```

Obviously, Racer determines that the concept `seriously-ill-human` is subsumed by `human-with-fever`. For the reals, Racer supports linear equations and inequations. Thus, we could add the following statement to the knowledge base in order to make sure the relation between the two attributes `temperature-fahrenheit` and `temperature-celsius` is properly represented.

```
(implies top (= temperature-fahrenheit
                (+ (* 1.8 temperature-celsius) 32)))
```

While constraints such as `(min age 18)` can indeed be represented in OWL using datatypes and range restrictions, the latter constraint cannot be represented in OWL because it involves constraints based on linear equations. Note that for readability reasons we did not use the OWL syntax here even for those parts that can be represented in OWL (or RDF).

If a concept `seriously-ill-human-1` is defined as

```
(equivalent seriously-ill-human-1
            (and human (>= temperature-fahrenheit 107.6)))
```

Racer recognizes the subsumption relationship with `human-with-fever` and the synonym relationship with `seriously-ill-human`.

In an A-box, it is possible to set up constraints between single individuals. This is illustrated with the following examples.

```
...
(signature
   :atomic-concepts (... teenager)
   :roles (...)
   :attributes (...)
   :individuals (eve doris)
   :objects (temp-eve temp-doris))
...
(constrained eve temp-eve temperature-fahrenheit)
(constrained doris temp-doris temperature-celsius)
(constraints
   (= temp-eve 102.56)
   (= temp-doris 39.5))
```

For instance, these declarations state that the individual `eve` is related via the attribute `temperature-fahrenheit` to the object `temp-eve`. The initial constraint `(= temp-eve 102.56)` specifies that the object `temp-eve` is equal to 102.56. Now, asking for the direct types of eve and doris reveals that both individuals are instances of `human-with-fever`.

In the following A-box there is an inconsistency since the temperature of 102.56 Fahrenheit is identical with 39.5 Celsius.

```
(constrained eve temp-eve temperature-fahrenheit)
(constrained doris temp-doris temperature-celsius)
(constraints
   (= temp-eve 102.56)
   (= temp-doris 39.5)
   (> temp-eve temp-doris))
```

We assume that the assertion `(> temp-eve temp-doris)` is removed from the knowledge base. Then, various kinds of queries involving concrete domains are possible. For instance: Check if certain concrete domain constraints are entailed by an A-box and a T-box. The following query returns true.

```
(constraint-entailed? (= temp-eve temp-doris))
```

One can also use the instance retrieval inference service and ask for the instances of `human-with-fever`: `(concept-instances human-with-fever)`. Both individuals, `doris` and `eve` are included in the answer set.

The example demonstrates the importance of reasoning for query answering in the context of the Semantic Web. The temperatures of `doris` and `eve` are given using

different measures. Nevertheless, a DL system such as Racer can compute adequate answers by supporting concrete domains.

We present another example that might be important for the Semantic Web: dealing with dates. The following declarations can be processed with Racer. The predicates `divisible` and `not-divisible` are defined for natural numbers and are reduced to linear inequations internally.

```
(define-concrete-domain-attribute year :type cardinal)
(define-concrete-domain-attribute days-in-month :type cardinal)

(implies Month (and (>= days-in-month 28) (<= days-in-month 31)))

(equivalent month-inleapyear
            (and Month
                 (divisible year 4)
                 (or (not-divisible year 100)
                     (divisible year 400))))

(equivalent February
            (and Month
                 (<= days-in-month 29)
                 (or (not month-inleapyear)
                     (= days-in-month 29))
                 (or month-inleapyear
                     (= days-in-month 28))))
```

Next, we assume some instances of February are declared.

```
(instance feb-2003 February)
(constrained feb-2003 year-1 year)
(constrained feb-2003 days-in-feb-2003 days-in-month)
(constraints (= year-1 2003))

(instance feb-2000 February)
(constrained feb-2000 year-2 year)
(constrained feb-2000 days-in-feb-2000 days-in-month)
(constraints (= year-2 2000))
```

Note that the number of days for both months is not given explicitly. Nevertheless, asking `(concept-instances month-inleapyear)` yields `(feb-2000)` whereas asking for `(concept-instances (not month-inleapyear))` returns `(feb-2003)`. In addition, one could check the number of days:

```
(constraint-entailed? (<> days-in-feb-2003 29))
(constraint-entailed? (= days-in-feb-2000 29))
```

In both cases, the answer is true. In the context of the Semantic Web reasoning over implicit information will become more and more important.

3.2 Concrete Domains for Ontology Languages

The logic for DAML+OIL or OWL was called $\mathcal{SHIQ}(\mathcal{D}_n)^-$. The minus sign is motivated by some important restrictions to the original approach for concrete domains (to ensure decidability of consistency problem, see [1]). The net effect of the imposed restrictions is that at the conceptual level (i.e., in the T-box) constraints on concrete domain values can be imposed only for a single individual. It should be emphasized however, that at the A-box level, i.e., for named individuals, no such restrictions apply, i.e., general constraint systems can be expressed by the description logic that Racer supports. Thus, decidability results for languages that are more expressive than DAML+OIL or OWL exist. Although decidability is an important issue, another issue that is important for practical applications concerns implementation technology and optimizations such that practical problems can indeed be solved.

Besides evaluating Racer's concrete domain reasoning with various tricky but small examples, a first study was conducted that uses a very complex KB derived from a real-world application. A $\mathcal{SHIQ}$ version of the Tambis KB (for further details see [3]) was extended resulting in the logic $\mathcal{SHIQ}(\mathcal{D}_n)^-$. It contains ∼400 named concepts and over 50 general axioms. For our experiments, concrete domain constructs were added to the KB. All Tambis T-boxes can be classified within seconds.

In order to classify KBs of this type of complexity with sufficient runtime performance, various tableau optimization techniques are required, especially two main techniques, dependency-directed backtracking [11] and pseudo-model merging [11, 10]. For instance, if pseudo-model merging is switched off for classifying the Tambis KB, Racer's runtime is increased by a factor of ∼20. The adaptation of these techniques to concrete domain reasoning is discussed in detail in [8].

Given the examples discussed above, it seems obvious that ontology languages and the Semantic Web can profit from concrete domains and corresponding constraints. However, the examples discussed above indicate that DAML+OIL and OWL support only a subset of what is required for practical applications. Evaluation results obtained with Racer indicate that adequate average case performance for answering queries involving linear inequation with order constraints can be achieved. Nonlinear multivariate inequations over complex numbers and linear inequations over natural numbers are also supported by Racer.

4 Accessing the Retrieval Inference Service: A Systems Perspective

The main examples for the Semantic Web use information retrieval applications involving one or more agents [5]. In a full-fledged information retrieval scenario, an agent might consult a document management system provided by an agent host environment. The agent can ask for documents that match a certain query in a similar way as discussed above. This scenario can also be realized with Racer if documents are annotated with meta data formalized with RDF [13]. Information about documents can be represented using A-boxes. RDF annotations for documents are read by Racer and corresponding assertions are added to an A-box. Concrete domains play

an important role for describing documents (e.g., year, ISBN number etc.). Agents can retrieve documents by posing retrieval queries to these A-boxes w.r.t. to specific T-boxes in the way exemplified above.

4.1 Publish/Subscribe Interface

If we consider an instance retrieval query Q w.r.t. an A-box A, then it is clear that the solution set for Q could be extended if more information is added to A over time (whoever is responsible for that, another agent or the agent host environment). It would be a waste of resources to frequently poll the host environment with the same query (and repeated migration operations). Therefore, Racer supports the registration of queries at some server w.r.t. to some A-box (Publish/Subscribe Interface). With the registration, the agent specifies an IP address and a port number. The corresponding Racer Server passes a message to the agent if the solution set of a previously registered instance retrieval query is extended. The message specifies the new individuals found to be instances of the query concept Q. We call the registration of a query, a subscription to a channel on which Racer informs applications about new query results. For details see the Racer manual [9].

Rather than considering a single query in isolation, a practical system should be able to consider query sets (as database systems do in many applications). With the publish/subscribe interface, multiple queries can be optimized by Racer. Instance retrieval queries can be answered in a faster way if the set of candidates can be reduced. In a similar way as for databases, the idea is to exploit results computed for previous instance retrieval queries by considering query subsumption (which is decidable in the case of the query language that Racer supports). However, this requires computing index structures for the T-box (the process is known as T-box classification) and, therefore, query subsumption is enabled on demand only. On the one hand, there are some applications, in which A-boxes are generated on the fly with few queries referring to a single A-box. On the other hand, there are applications which pose many queries to more or less "static" T-boxes and A-boxes (which are maybe part of the agent host environment). The Racer Server supports both application scenarios. As a design decision, Racer computes answers for queries with as few resources as possible. Nevertheless, a Racer Server can be instructed to compute index structures in advance if appropriate to support multiple queries.

4.2 Realizing Local Closed World Assumptions in Applications

Feedback from many users of the Racer system indicates that, for instance, instance retrieval queries could profit from possibilities to "close" a knowledge base in one way or another. Due to the non-monotonic nature of the closed-world assumption and the ambiguities about what closing should actually mean, in description logic inference systems usually there is no support for the closed-world assumption. However, with the publish and subscribe interface of Racer, users can achieve a similar effect. Consider, for instance, a query for a book which does not have an author. Because of the open-world assumption, subscribing to a channel for $Book \ \& \ (\leq 0\, has\_author)$ does

not make much sense. Nevertheless the agent can subscribe to a channel for *Book* and a channel for $(\geq 1\, has\_author)$. It can accumulate the results returned by Racer into two variables A and B, respectively, and, in order to compute the set of books for which there does not exist an author, it can consider the complement of B wrt. A. We see this strategy as an implementation of a local closed-world (LCW) assumption.

However, as time evolves, authors for documents determined by the above-mentioned query indeed might become known. In other words, the set B will probably be extended. In this case, the agent is responsible for implementing appropriate backtracking strategies, of course.

The LCW example demonstrates that the Racer publish and subscribe interface is a very general mechanism, which can also be used to solve other problems in knowledge representation. Due to space restrictions, we can only give ideas for applications and services which can be implemented using logic. The examples we have given here should stimulate developers of agent systems and agent host environments to use the facilities of state of the art description logic inference engines. As a summary we discuss additional features of the Racer System in the next section.

4.3 Additional Features of the Racer System

Optimizations: Various optimization techniques for ontology-based query answering with respect to T-boxes, A-boxes, and concrete values have been developed, implemented, and investigated with the Racer System. One of the design goals of Racer is to automatically select state of the art optimization techniques that are applicable to the current input.

Persistency: In a similar way as in database systems, for query answering w.r.t. T-boxes and A-boxes complex data structures are computed and used internally by Racer. Internal structures of T-boxes and A-boxes being processed for query answering can be saved to disk for quick access and later reuse if the Racer Server is restarted.

Multi-User Support, Thread Safeness, Locking, Load Balancing: In a distributed systems context, there can be multiple agents connecting the a server at the same time. If they refer to the same A-boxes and T-boxes (which is very likely in the scenarios presented above), requests must be synchronized. Thus similar problems as with databases such as thread safeness, locking, and load balancing have to be dealt with. For instance, if multiple Racer Servers are started, queries can be automatically directed to "free" Racer Servers. These problems are tackled by the Racer Proxy, which is supplied as part of the Racer System distribution.

5 Summary

This paper states that description logic systems can be used as query engines for DAML+OIL or OWL ontologies. In particular, we argue that some aspects of individuals in concept terms can be adequately represented using A-boxes. The remaining aspects have to be approximated if current description logic inference systems are to be used for practical applications.

Then, we pointed out that from a knowledge representation point of view, the languages DAML+OIL and OWL are rather weak w.r.t. data constraints since simple and well-understood algebraic representation techniques are not supported – surprisingly.

Third, we demonstrated that an inference system for the Semantic Web comprises more than just a concept consistency tester or subsumption computation algorithm. A highly optimized architecture is required for dealing with practical application problems. As the evaluation results indicate, it seems that Racer is on the right way, but it definitely needs more research for many of the upcoming Semantic Web applications.

Acknowledgments

We are grateful to numerous users of the Racer system who used Racer to solve many different kinds of application problems. Detailed comments from our users helped to reach the level of maturity that Racer currently has. We hope that all answers to questions arrived in time although the delay might get larger due to high workload. Nevertheless, not all complaints, in particular those concerning performance, could be easily answered and solved, some application problems are simply hard to deal with.

References

[1] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider. *The Description Logic Handbook*. Cambridge University Press, 2002. In print.

[2] F. Baader, I. Horrocks, and U. Sattler. Description logics as ontology languages for the semantic web. In D. Hutter and W. Stephan, editors, *Festschrift in honor of J"org Siekmann*. LNAI. Springer-Verlag, 2003.

[3] S. Bechhofer, Ian Horrocks, C. Goble, and R. Stevens. OilEd: A reason-able ontology editor for the semantic web. In T. Eiter F. Baader, G. Brewka, editor, *Proceedings of KI 2001: Advances in Artificial Intelligence Joint German/Austrian Conference on AI*, volume 2174 of *LNAI*, page 396 ff., 2001.

[4] S. Bechhofer, R. Möller, and P. Crowther. The DIG description interface. In *Proc. International Workshop on Description Logics – DL'03*, 2003.

[5] T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *Scientific American*, May 2001.

[6] D. Brickley and R.V. Guha. RDF vocabulary description language 1.0: RDF Schema, http://www.w3.org/tr/2002/wd-rdf-schema-20020430/, 2002.

[7] V. Haarslev and R. Möller. Racer system description. In *International Joint Conference on Automated Reasoning, IJCAR'2001, June 18-23, 2001, Siena, Italy.*, 2001.

[8] V. Haarslev and R. Möller. Practical reasoning in RACER with a concrete domain for linear inequations. In I. Horrocks and S. Tessaris, editors, *Proceedings of*

the International Workshop on Description Logics (DL'2002), Apr. 19-21, 2002, Toulouse, France, pages 91–98, August 2002.

[9] V. Haarslev and R. Möller. The Racer user's guide and reference manual, 2003.

[10] V. Haarslev, R. Möller, and A.-Y. Turhan. Exploiting pseudo models for TBox and ABox reasoning in expressive description logics. In R. Goré, A. Leitsch, and T. Nipkow, editors, *Proceedings of the International Joint Conference on Automated Reasoning, IJCAR'2001, June 18-23, 2001, Siena, Italy*, Lecture Notes in Computer Science, pages 61–75. Springer-Verlag, June 2001.

[11] I. Horrocks. *Optimising Tableaux Decision Procedures for Description Logics*. PhD thesis, University of Manchester, 1997.

[12] Ian Horrocks and Sergio Tessaris. Querying the semantic web: a formal approach. In Ian Horrocks and James Hendler, editors, *Proc. of the 13th Int. Semantic Web Conf. (ISWC 2002)*, number 2342 in Lecture Notes in Computer Science, pages 177–191. Springer-Verlag, 2002.

[13] Adobe Systems Inc. Embedding XMP metadata in application files, 2002.

[14] G. Kuper, L. Libkin, and J. Paredaens (Eds.). *Constraint Databases*. Springer-Verlag, 1998.

[15] O. Lassila and R.R. Swick. Resource description framework (RDF) model and syntax specification. recommendation, W3C, february 1999. http://www.w3.org/tr/1999/rec-rdf-syntax-19990222, 1999.

[16] F. van Harmelen, J. Hendler, I. Horrocks, D. L. McGuinness, P. F. Patel-Schneider, and L. A. Stein. OWL web ontology language reference, http://www.w3.org/tr/owl-guide/, 2003.

[17] F. van Harmelen, P.F. Patel-Schneider, and I. Horrocks (Editors). Reference description of the DAML+OIL (march 2001) ontology markup language, http://www.daml.org/2001/03/reference, 2001.