Inference Web: Portable Explanations for the Web

Deborah L. McGuinness and Paulo Pinheiro da Silva
Knowledge Systems Laboratory
Stanford University

Abstract

The World Wide Web lacks support for explaining information provenance. When web applications return answers, many users do not know what information sources were used, when they were updated, how reliable the source was, what information was looked up versus derived, and if something was derived, how it was derived. In this paper we introduce the Inference Web (IW) that addresses the problems associated with opaque query answers by providing portable, combinable, and distributed explanations. The explanations include information concerning where answers came from and how they were deduced (or retrieved). The IW solution includes: an extensible web-based registry containing details on information sources and reasoners, a portable proof specification, and an explanation browser.

1 Introduction

Inference Web (IW) aims to enable applications that can generate portable and distributed explanations for any of their answers. There are many reasons that users and agents need to understand the provenance of information that they get back from applications. The main motivating factors for us are interoperability, reuse, and trust. Interoperability is essential if agents are to collaborate. Trust and reuse of retrieval and deduction processes is facilitated when explanations are available. Ultimately, if users and/or agents are expected to trust information and actions of applications and if they are expected to use and reuse application results potentially in combination with other information or other application results, they may need to have access to many kinds of information such as source, recency, authoritativeness, method of reasoning, term meaning and interrelationships, etc.

This work builds on experience designing explanation components for reasoning systems [McGuinness, 1996; McGuinness-Borgida, 1995; Borgida, et. al, 1999, and 2000] and experience designing query components for frame-like systems [McGuinness, 1996; Borgida-McGuinness, 1996] to generate requirements. We also obtained requirements input from contractors in DARPA-sponsored programs concerning knowledge-based applications (the High Performance Knowledge Base program[1], Rapid Knowledge Formation Program[2], and the DARPA Agent Markup Language Program[3] and more recently, the ARDA AQUAINT[4] and NIMD[5] programs). We also obtained requirements from literature on explanation for expert systems, (e.g., [Swartout, et. al., 1991]), and usability of knowledge representation systems (e.g., [McGuinness-Patel-Schneider, 1998 and 2003]), and theorem proving explanation (e.g., [Felty-Miller, 1987]).

Our goal is to address needs that arise with use of systems performing reasoning and retrieval tasks in heterogeneous environments such as the web. Users may obtain information from individual or multiple sources and they may need to determine which information to trust. Users may also obtain conflicting information and they may need additional information to help evaluate what to believe. They may also gather information from complex and hybrid sources and they need help integrating answers and solutions. As web usage grows, a broader and more distributed array of information services are available for use and the needs for explanations that can be shared across distributed environments grow.

In this paper, we include a list of explanation requirements gathered from past work and from surveying users. We present the Inference Web architecture and provide a description of the major IW components including the portable proof specification, the registry (containing information about inference engines, proof methods, and ontologies), and the justification browser. We also provide some simple

[1] http://reliant.teknowledge.com/HPKB/
[2] http://reliant.teknowledge.com/RKF/
[3] http://www.daml.org
[4] http://www.ic-arda.org/InfoExploit/aquaint/
[5] http://www.ic-arda.org/Novel_Intelligence/

usage examples. We conclude with a discussion of our work in the context of explanation work and state our contributions in the areas of application interoperability, reuse, and trust.

2 Requirements

If humans and agents need to make informed decisions about when and how to use answers from applications, there are many things to consider. Decisions will be based on the quality of the source information, the suitability and quality of the reasoning engine, and the context of the situation. Particularly for use on the web, information needs to be available in a distributed environment and needs to be interoperable across applications.

First, we consider issues concerning the source information. Even when search engines or databases simply retrieve asserted or "told" information, users (and agents) may need to understand where the source information came from at varying degrees of detail. This information sometimes called *provenance*, may be viewed as meta information about told information. Provenance information may include:

- Source name (e.g., CIA World Fact Book)
- Date and author(s) of last update
- Author(s) of original information
- Authoritativeness of the source (is this knowledge store considered or certified as reliable by a third party?)
- Degree of belief
- Degree of completeness (Within a particular scope, is the source considered complete. For example, does this source have all of the employees of a particular organization up until a some date? If so, not finding a particular employee would mean that they are not employed, counting employees would be an accurate response to number of employees, etc.)

The information above could be handled with meta information about content sources and about individual assertions. Additional types of information may be required if users need to understand the meaning of terms or implications of query answers. If applications make deductions or otherwise manipulate information, users may need to understand how deductions were made and what manipulations were done. Information concerning derived or manipulated information may include:

- Term or phrase meaning (in natural language or a formal language)
- Term inter-relationships (ontological relations including subclass, superclass, part-of, etc.)
- The source of derived information (reasoner used, reasoner method, reasoner inference rule, etc.)
- Reasoner description (is the reasoner used known to be sound and complete?)
- Term uniqueness (is J. Smith the same individual as John Smith?)
- Term coherence (is a particular definition incoherent?)
- Source consistency (is there support in a system for both A and ~A)
- Were assumptions used in a derivation? If so, have the assumptions changed?

3 Use Cases

Every combination of a query language with a query-answering environment is a potential new context for the Inference Web. We provide two motivating scenarios.

Consider the situation where someone has analyzed a situation previously and wants to retrieve this analysis. In order to present the findings, the analyst may need to defend the conclusions by exposing the reasoning path used along with the source of the information. In order for the analyst to reuse the previous work, s/he will also need to decide if the source information used previously is still valid (and possibly if the reasoning path is still valid).

Another simple motivating example arises when a user asks for information from a web application and then needs to decide whether to act on the information. For example, a user might use a search engine interface or a query language such as DQL[6] for retrieving information such as "zinfandels from Napa Valley" or "wine recommended for serving with a spicy red meat meal" (as exemplified in the wine agent example in the OWL guide document[Smith et. al., 2003]). A user might ask for an explanation of why the particular wines were recommended as well as why any particular property of the wine was recommended (like flavor, body, color, etc.). The user may also want information concerning whose recommendations these were (a wine store trying to move its inventory, a wine writer, etc.).

In order for this scenario to be operationalized, we need to have the following:

- A way for applications (reasoners, retrieval engines, etc.) to dump justifications for their answers in a format that others can understand. To solve this problem we introduce a portable proof specification.
- A place for receiving, storing, manipulating, annotating, comparing, and returning meta information used to enrich proofs and proof fragments. To address this requirement, we introduce the Inference Web Registry for storing the meta information and the Inference Web Registrar web application for handling the Registry.
- A way to present justifications to the user. As one solution to this problem, we introduce a proof browser.

4 Inference Web

We begin with a short description of different categories of Inference Web users. These users along with the usage

[6] http://www.daml.org/2002/08/dql/.

examples above motivate the main components of Inference Web: portable proofs and their parsers, registry and its registrar, and proof browsers.

The prime users of inference web are:

- Application developers (authors of reasoners, search engines, database systems, etc.) who would like to justify why their answers to queries should be believed or who would like to state under what conditions their systems are best used. These people are interested in allowing their system to not only answer queries but also provide meta information about the answer. The portable proof specification in Inference Web allows application developers to store this information in a sharable format.
- Authors of hybrid solutions programs interested in combining multiple answering systems and/or knowledge bases. These people need to understand how terms relate to each other and how answers were derived and might be integrated. Examples of such people include ontology builders who are merging ontologies or extending ontologies, crawler or wrapper authors, people combining databases or knowledge based systems, etc. The registry in Inference Web provides a store of information about inference methods, inference engines, ontologies, and sources that helps address these issues.
- Humans or agents needing to decide if they can trust either retrieved information or inference processes used to retrieve information. The browser in inference web addresses these issues by allowing users to view partial or complete justifications for answers.

Inference Web contains both *data* used for proof generation and presentation and *tools* for building, maintaining, presenting, and manipulating proofs. Inference Web data includes proofs and proof fragments published anywhere on the web. Inference Web data also includes a centralized repository of meta-data including sources, inference engines, inference rules and ontologies. Inference Web tools include a registrar for interacting with the registry, a parser for proof I/O, a browser for displaying proofs, and planned future tools such as proof web-search engines, proof verifiers (possibly utilizing tools such as Specware, etc). In this paper, we limit our discussion to the portable proofs (and an associated parser), the registry (and the associated registrar tools), and the browser.

4.1 Portable Proofs

Systems that may be asked to return a justification for an answer along with an answer need to expose provenance information along with their deductive process possibly including meta information about the system itself. We provide a specification written in the web markup language DAML+OIL [Connolly et. al., 2001]. Proofs dumped in the portable proof format become a portion of the Inference Web data used for presenting proofs. Our portable proof specification includes four major components of IW proof trees: inference rules, inference steps, well formed formulae (WFFs), and referenced ontologies. Inference rules (such as modus ponens) can be used to deduce a consequent (a well formed formula) from any number of antecedents (also well formed formulae). An inference step is a single application of an inference rule. The inference step will be associated with the consequent WFF and it will contain pointers to the antecedent WFFs, the inference rule used, and any variable bindings used in the inference rule application. The antecedent WFFs may come from other inference steps, existing ontologies, extraction from documents, or they may be assumptions. Figure 1 presents a typical dump of a WFF.

```
<?xml version='1.0'?> <rdf:RDF (...)>
<iw:WFF>
    <iw:WFFContent> (a WFF is stored as a predicate logic
                        sentence)
      <daml:List rdf:about='IW/spec/fopl.daml#Clause'>
        <daml:first>
          <fopl:Negated-Predicate-Of-Terms
            fopl:SymbolName='holds'>
          <fopl:hasArgumentList
rdf:parseType='daml:collection'>
            <iw:Constant>
<fopl:SymbolName>type</fopl:SymbolName> </iw:Constant>
            <fopl:Variable fopl:SymbolName='?inst'/>
           (...)
      </daml:List>
    </iw:WFFContent>
    <iw:isConsequentOf rdf:parseType='daml:collection'>
     (a WFF can be associated to a set of Inference steps)
     <iw:InferenceStep>
        <iw:hasInferenceRule
                rdf:parseType='daml:collection'>
          <iw:InferenceRule
                rdf:about='../registry/IR/GMP.daml'/>
        </iw:hasInferenceRule>
        <iw:hasInferenceEngine
                rdf:parseType='daml:collection'>
          <iw:InferenceEngine
                rdf:about='../registry/IE/JTP.daml'/>
         </iw:hasInferenceEngine>
           (...)
        <iw:has Antecedent
                rdf:parseType='daml:collection'>
        (inference step antecedents are IW files with
         their own URIs)
          <iw:WFF rdf:about='../sample/IW3.daml'/>
          <iw:WFF rdf:about='../sample/IW4.daml'/>
        </iw:hasAntecedent>
        <iw:hasVariableMapping
rdf:type='http://www.daml.org/2001/03/daml+oil#List'/>
           (...)
      </iw:InferenceStep>
    </iw:isConsequentOf>
  </iw:WFF>
</rdf:RDF>
```

Figure 1. An Inference Web Proof

There we can see an instance of a WFF, an inference step, and an inference rule. There is no ontology associated with this WFF since it is derived. If it had been asserted, it would require an association to the ontology that contains it. A proof can then be defined as a tree of inference steps explaining the process of deducing the consequent WFF. In Inference Web, proofs are *trees of proof fragments* rather than single monolithic proofs. With respect to a query, a logical starting point for a proof in Inference Web is a proof fragment that contains the last inference step used to derive a WFF that is an answer for the query. Any inference step can be presented as a stand alone, meaningful proof fragment as it contains the inference rule used with links to its antecedents and variable bindings. The generation of proof fragments is a straightforward task once inference engine data structures storing proof elements are identified as IW components. To facilitate the generation of proofs, the Inference Web provides a parser in Java that dumps proofs from IW components and uploads IW components from proofs. The development of an IW parser in LISP is under consideration.

The IW infrastructure can automatically generate follow-up questions for any proof fragment by asking how each antecedent WFF was derived. The individual proof fragments may be composed together to generate a complete proof, i.e., a set of inference steps culminating in inference steps containing only asserted (rather than derived) antecedents.. When an antecedent WFF is asserted, there are no additional follow-up questions required and that ends the complete proof generation.

A WFF may be the consequent of any number of inference steps. IW can be used to support multiple justifications for any particular WFF. WFFs may not be the consequent of an inference step if they are assumptions or merely asserted information in an ontology that the user is referencing. The specification of IW concepts used in Figure 1 is available at http://www.ksl.stanford.edu/software/IW/spec.

4.2 Registry

The IW registry is currently a centralized repository of information used to enrich explanations with details about *authoritative sources*, *ontologies*, *inference engines*, and *inference rules*. In the future, we may store only pointers to registry entries published elsewhere on the web. Our registry includes template information about each of the classes in the registry. For example, inference engines may have the following properties associated with them: name, URL, author(s), date, version number, organization, etc. The current demonstration registry is available at: http://belo.stanford.edu:8080/iwregistry/BrowseRegistry.jsp

Information in the registry contains the information linked to in the proofs. Every inference step should have a link to at least one inference engine that was responsible for instantiating the inference step itself, as shown in Figure 1.

The description of inference rules is one of the most important features of the Registry. Registered rules can be atomic or can be derived from other registered rules.

In order to interact with the IW Registry, there is a Registrar web application allowing users to update or browse the registry. A screen shot from the Registrar interface for inference rules is included in Figure 2. This displays a listing of the atomic inference rules for the JTP model-elimination reasoner at Stanford. Each of the inference rules includes a name, description, optional example, and optional formal specification.

Many reasoners also use a set of derived rules that may be useful for optimization, for example. One individual reasoner may not be able to provide a proof of the derived rules but one reasoner may point to another reasoner's proof of a rule. Thus, reasoner-specific rules can be explained in the Registry before the reasoner is actually used to generate IW proofs. Inference web thus provides a way to use one reasoner to explain another reasoner's inference rules. (This was the strategy used in [Borgida et al, 1999] for example.) This strategy may be useful for explaining heavily optimized inference engines. Inference Web's registry, when fully populated, will contain inference rule sets for many common reasoning systems. Users may view inference rule sets to help them decide whether to use a particular inference engine.

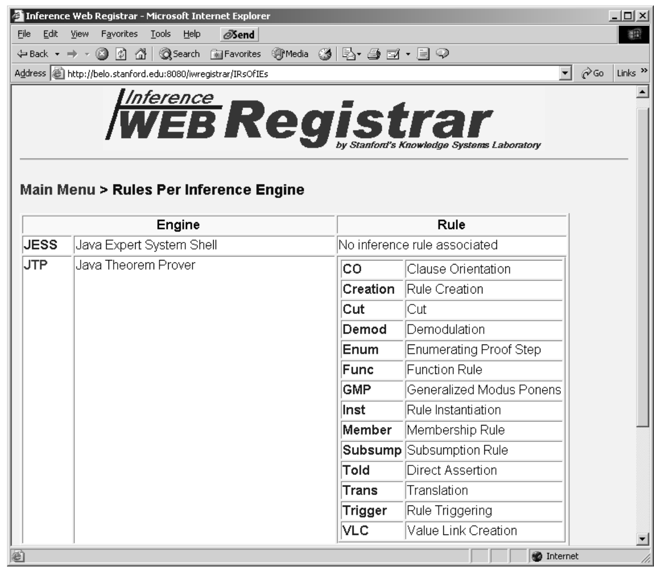

Figure 2: Sample Inference Web Registrar Entry

Ontologies are another component in the IW registry. Ontologies are stores of assertions that may be used in proofs. It can be important to be able to present information such as ontology source, date, version, URL (for browsing), etc. Figure 3 contains a sample ontology registry entry for the ontology used in our wine examples.

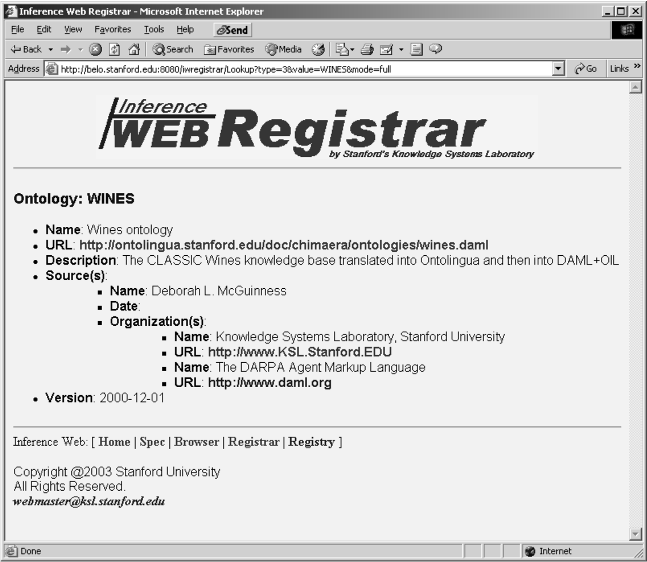

Figure 3: Sample Inference Web Ontology Entry

4.3 Browser

Inference Web includes a browser that displays proof fragments in a number of formats. Initially, we include formats for restricted English, KIF[7], and conjunctive normal form. We also expect that some applications may implement their own displays using the IW API.

The prototype browser allows a user to see an inference rule used along with the derived sentence and the antecedent sentences. The browser implements a lens metaphor responsible for rendering a fixed number of levels of inference steps depending on the lens magnitude setting. Figure 4 presents an inference step for one wine use case in Section 3. Prior to this view, the program has asked what wine to serve with a spicy-red-meat course. In Figure 4, one can see that NEW-COURSE12, which is the selected meal course, requires a drink that has a full body since it is a spicy red meat course. The sentences are formatted in KIF and the lens magnitude is one, thus the browser displays the inference step used to derive it including its antecedents. A lens setting of two would also include the antecedent's derivations.

We believe that one of the keys to presentation of justifications is breaking proofs into separable pieces. Since we present fragments, automatic follow-up question support is a critical function of the IW browser. Every element in the viewing lens can trigger a browser action. The selection of an antecedent that is derived re-focuses the lens on an antecedent's inference step. For other lens elements, associated actions present Registry meta-information in the *Trust Disclosure Panel*. The selection of the consequent presents details about the inference engine used to derive the actual theorem. The selection of an inference rule presents a description of the rule. The selection of a sentence that is asserted information presents details about ontologies where the axiom is defined. In Figure 4, selecting the consequent would present information about JTP- the inference engine used to derive it. Selecting GMP – the inference rule, would present information about JTP's Generalized Modus Ponens rule. Selecting a statement such as "beef curry is a spicy red meat" or "spicy red meat courses require full-bodied wines" presents information about the wines ontology. Selecting a derived or cached inference rule presents information about the inference rule. (JTP uses a set of special purpose axioms for more efficiently reasoning with the DAML+OIL language and those inferences may be used in an explanation). An example of this process can be seen from the Inference Web web pages at: http://www.ksl.stanford.edu/software/iw/Ex1/.

[7] http://logic.stanford.edu/kif/kif.html.

5 Related Work and Contributions

Recognition of the importance of explanation components for reasoning systems has existed in a number of fields for many years. For example, from the early experiences with MYCIN[Shortliffe,1976], expert systems researchers understood the need for systems that understood their reasoning processes and could generate explanations in a language understandable to its users. Inference web attempts to stand on the shoulders of past work in expert systems, such as MYCIN and the explainable expert system on generating explanations using both their leanings on how to generate explanations and interoperating with next generation systems that generate explanations. IW also builds on the learnings of explanation in description logics (e.g., [McGuinness, 1996; Borgida, et.al, 2000]) that attempts to provide a logical infrastructure for separating pieces of logical proofs and automatically generating follow-questions based on the logical format. It also looked to the theorem proving community with work such as [Felty-Miller, 1987]) that attempts to provide understandable and flexible explanations of theorem provers and foundational systems such as [Boyer, et. al, 1995] that provides some explanations of deductions along with WFFs not proven.

Figure 4: An Inference Web Browser Screen

We are not aware of work that has attempted to provide an infrastructure for providing, storing, and manipulating interoperable explanations of heterogeneous reasoning systems. Beyond just explaining a single system, Inference

Web attempts to incorporate best in class explanations and provide a way of combining and presenting justifications that are available. It does not take one stance on the form of the explanation since it allows deductive engines to dump single or multiple explanations of any deduction in deductive language of their choice. It provides the user with flexibility in viewing fragments of single or multiple explanations in multiple formats. IW simply requires inference rule registration and portable proof format.

Inference Web provides the following contributions:

- An architecture supporting interoperability between agents using portable proofs. Portable proofs are specified in the emerging web standard DAML+OIL so as to leverage XML-, RDF-, and DAML-based information services. Proof fragments as well as entire proofs may be interchanged.
- Lightweight proof browsing using the lens-based IW proof browser supporting either pruned justifications or guided viewing of a complete reasoning path.
- Support for alternative justifications using multiple inference steps. This allows derivations to be performed by performance reasoners with explanations being generated by alternative reasoners aimed at human consumption.
- Registry of inference engines, ontologies, and sources.

We are currently extending the Stanford's JTP[8] theorem prover to produce portable proofs and simultaneously populating the IW registry with JTP information. Future work includes expanding to include other inference engines. We also intend to provide specialized support for why-not questions expanding upon [Chalupsky-Russ,2002] and [McGuinness,1996]. We are also looking at additional support for proof browsing and pruning.

6 Conclusion

Inference web enables applications that can generate portable explanations of their conclusions. We described the major components of IW – the portable proof specification based on the emerging web language-DAML (soon to be updated to OWL), the registry, and the IW proof browser. We facilitated use in a distributed environment by providing IW tools for registering and manipulating proofs, proof fragments, inference engines, ontologies, and source information. We also facilitated interoperability by specifying the portable proof format and providing tools for manipulating proofs and fragments. We have implemented the IW approach for one inference engine (JTP) and are in discussions with additional reasoner authors to include more reasoning engines. We have presented the work at some government sponsored program meetings(RKF, DAML, and AQUAINT) to gather input from other reasoner authors/users and have obtained feedback and interest. We are initiating additional reasoner, ontology, and source registrations.

[8] http://www.ksl.stanford.edu/software/jtp/.

Acknowledgements

Many people have provided valuable input to our work. Thanks in particular go to colleagues at KSL including Richard Fikes, Jessica Jenkins, Gleb Frank, Eric Hsu, and Yulin Li for input on JTP, our specification or applications. Also thanks go to a number of colleagues in some government programs who provided input including Hans Chalupsky, Peter Clark, Ken Forbus, Ken Murray, and Steve Reed, All errors, of course are our responsibility.

References

[Borgida et.al, 2000] Alex Borgida, Enrico Franconi, and Ian Horrocks. Explaining *ALC* subsumption. In *Proc. of the 14th European Conf. on Artificial Intelligence (ECAI'2000)*, pages 209-213. IOS Press, 2000.

[Borgida et al., 1999] Alex Borgida, Enrico Franconi, Ian Horrocks, Deborah McGuinness, and Peter Patel-Schneider. ``Explaining ALC subsumption" Proceedings of the International Workshop on Description Logics(DL-99), Linköping, Sweden, July 1999, pp 33-36.

[Boyer et.al, 1995] Robert Boyer, Matt Kaufmann, and J. Moore. The Boyer-Moore Theorem Prover and Its Interactive Enhancement, *Computers and Mathematics with Applications*, **29**(2), 1995, pp. 27-62.

[Chalupsky and Russ, 2002] Hans Chalupsky and Tom Russ. WhyNot: Debugging Failed Queries in Large Knowledge Bases. In *Proceedings of the Fourteenth Innovative Applications of Artificial Intelligence Conference (IAAI-02)*, pages 870-877.

[Connolly et. al, 2001] Dan Connolly, Frank van Harmelen, Ian Horrocks, Deborah L. McGuinness, Peter F. Patel-Schneider, and Lynn Andrea Stein. DAML+OIL (March 2001) Reference Description. W3C Note 18 December, 2001. http://www.w3.org/TR/daml+oil-reference.

[Dean et. al., 2002] Mike Dean, Dan Connolly, Frank van Harmelen, James Hendler, Ian Horrocks, Deborah McGuinness, Peter Patel-Schneider, and Lynn Andrea Stein. OWL Web Ontology Language 1.0 Reference. World Wide Web Consortium (W3C) Working Draft 29 July 2002. Latest version is available at http://www.w3.org/TR/owl-ref/.

[Felty and Miller, 1987] Amy Felty and Dale Miller. Proof Explanation and Revision. University of Penn, Tech. Report MSCIS8817, 1987 http://cm.bell-labs.com/who/felty/abstracts/proof87.html.

[Fikes et al., 2002] Richard Fikes, Pat Hayes, Ian Horrocks, editors. *DAML Query Language(DQL) Abstract Specification.* http://www.daml.org/2002/08/dql/.

[McGuinness, 1996] Deborah L. McGuinness. 1996. *Explaining Reasoning in Description Logics.* Ph.D. Thesis, Rutgers University, Technical Report LSCR-TR-277.

[McGuinness and Borgida, 1995] Deborah L. McGuinness and Alex Borgida. *Explaining Subsumption in Description Logics*. In *Proc.14th International Joint Conf. on Artificial Intelligence*, Montreal, Canada. 1995.

[McGuinness and Patel-Schneider, 2003] Deborah McGuinness and Peter Patel-Schneider. ``From Description Logic Provers to Knowledge Representation Systems''. Franz Baader, Deborah McGuinness, Daniele Nardi, and Peter Patel-Schneider, editors The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press, 2003.

[McGuinness and Patel-Schneider, 1998] Deborah McGuinness and Peter Patel-Schneider. ``Usability Issues in Knowledge Representation Systems''. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence*, Madison, Wisconsin, July, 1998. Updated version of ``Usability Issues in Description Logic Systems'' published in *Proc. of International Workshop on Description Logics*, Gif sur Yvette, (Paris), France, Sept, 1997.

[Shortliffe, 1976] Edward Hance Shortliffe. Computer-Based Medical Consultations: MYCIN. Elsevier/North Holland, New York, 1976.

[Smith et al., 2003] Michael Smith, Deborah L. McGuinness, Raphael Volz and Chris Welty. Web Ontology Language (OWL) Guide Version 1.0. World Wide Web Consortium (W3C) Working Draft. Available at http://www.w3.org/TR/owl-guide.

[Specware,2001] http://www.specware.org/

[Swartout et al., 1991] W. Swartout, C. Paris, and J. Moore. **"**Explanations in Knowledge Systems: Design for Explainable Expert Systems". In IEEE Intelligent Systems, June 1991. http://www.computer.org/intelligent/ex199/x3058abs.htm**.**