# Implementing Modal Extensions of Defeasible Logic for the Semantic Web

**Nikos Dimaresis and Grigoris Antoniou**

Computer Science Department, University of Crete, Greece
Institute of Computer Science, FORTH, Greece
{dimares,antoniou}@ics.forth.gr

## Defeasible Logic

Defeasible reasoning is a nonmonotonic reasoning approach in which the gaps due to incomplete information are closed through the use of defeasible rules. Defeasible logic (Nute 1994) and its variants are an important family of defeasible reasoning methods. It is a simple, efficient but flexible non-monotonic formalism that offers many reasoning capabilities embodies the concept of preference and it has low computational complexity. Recent theoretical work on defeasible logics has: (i) established some relationships to logic programming (Antoniou et al. 2006); (ii) analyzed the formal properties of these logics (Antoniou et al. 2001) and (iii) has delivered efficient implementations (Antoniou & Bikakis 2007).

Its use in various application domains has been advocated, including modelling of contracts (Grosof 2004), (Governatori 2005), legal reasoning (Governatori, Rotolo, & Sartor 2005), agent negotiations (Governatori et al. 2001), modelling of agents and agent societies (Governatori & Rotolo 2004).

Nonmonotonic rule systems are expected to play an important role in the layered development of the Semantic Web. Defeasible reasoning systems that are used on applications to the Semantic Web have already been implemented (Antoniou & Bikakis 2007). Semantic Web community has performed extensive research in the area of policies. It is a concept that encompasses many different notions and one of them are the business rules.

In the current work, we develop a nonmonotonic rulebased system that can reason in Semantic Web applications associated with policies and business rules. It is based on an extension of defeasible logic with modalities and supports reasoning with RDF/S ontologies.

## Extension of Defeasible Logic with Modalities

As stated in (Antoniou & Arief 2002), defeasible logic is an appropriate nonmonotonic approach for the modeling and reasoning with business rules. The expression power of the formal specification language that is required by the business rules community is high and includes deontic notions like obligation, permission and prohibition. Our work in this paper is based on the logical framework developed in (Governatori & Rotolo 2004), where Defeasible Logic is extended with modal operators.

This is a non-monotonic and computationally-oriented framework that combines perspectives from rational BDI agents and agent models that are based on social and normative concepts. It deals with the following modalities: i) *knowledge* (the agent's theory about the world); ii) *intention* (that is the agent's general policies); iii) *agency* (agent's intentional actions); iv) *obligation* (absolute obligations from the agent's normative system).

In our work we consider a fifth kind of modality, *permission*, which is a basic deontic operator. Defeasible Logic is the suitable non-monotonic formalism that can deal with the defeasible nature of informational attitudes, internal motivational attitudes like intention and agency, and external normative concepts like obligation. A rule-based nonmonotonic formalism was developed that extends defeasible logic and represents and reasons with these modal operators. This formalism has as main feature the introduction of the mode for every rule, which determines the modality of a rule's conclusion. It supports modalised literals that can be defined in defeasible theories as facts or as part of the antecedents of rules.

## Translation into Logic Programs

We use the approach of meta-program formalization to simulate the proof theory of the extension of defeasible logic to reason over a defeasible theory. The meta-program was implemented in the logic programming language of Prolog. It has similar structure to the meta-programs that have been developed for the propositional defeasible logic (Antoniou et al. 2006), with an additional argument in predicates that represent rules. This is a modal operator that determines the mode of the rule. For example, a strict rule is defined as:

```
strict(Name,Operator,Head,Body)
```

A modalised literal is represented as prefixed with the modal operator (*agency, intention, obligation, permission*). An unmodalised literal belongs to the knowledge of the environment.

The next clauses define definite provability: a literal is definite provable in the knowledge modality, if it is a fact `strictly(P,knowledge):-fact(P)`
and in other modalities, if the corresponding modal literal is a fact. A definite provable literal in intention is defined as `strictly(P,intention):-fact(intention(P)).`
Finally a literal is definite provable in a modality, if it is supported by a strict rule, with the same mode and its premises are definitely provable. A definite provable literal in agency is defined as
`strictly(P,agency):-strict(R,agency,P,A), strictly(A).`

## Extension of Defeasible Logic with Modalities

Our nonmonotonic rule-based system provides automated decision support, when running a specific case with the given logic programs and ontological knowledge to get a correct answer. Figure 1 presents the overall architecture of our system. The system works in the following way: An organization imports its rules (business rules, policies e.t.c.) as logic programs. They follow the structure of the extended meta-program with modalities. The logic programming system is YAP. This Prolog engine supports arithmetic built-in predicates that are embedded in the meta-program and facilitate our system to support arithmetic operations. The RDF Translator is used to translate dynamically the RDF/S data into logical facts and rules, which can be processed by the organization's rules. The Reasoning Engine compiles the meta-program and the logic programs and evaluates the answer to user's queries.

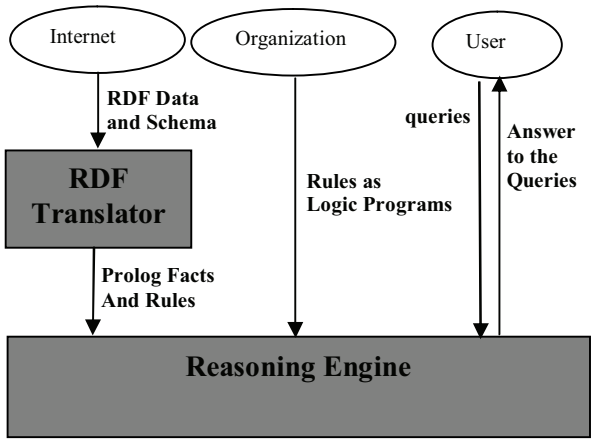


**Figure 1 : System Architecture**

## Use Case: University Regulations

We envisage applications of our system for modelling and reasoning with business rules. As a concrete application, we modelled a variety of university regulations from the Department of Computer Science at the University of Crete. The system offers automated support for reasoning with regulations. An example is the following typical rule from the department's policy in enrollment in courses for students:

*A student has the permission to enroll in a course during a semester if he has passed the course's prerequisites, unless he has enrollled in courses this semester with total number of course units more than 35. A student is also forbidden to enroll in a course in a spring semester, if he has enrollled in the same course just the fall semester the same academic year (the previous year).*

This is a typical rule with exceptions. In our logical framework, these rules can be represented with the use of defeasible rules, which introduce the deontic operators of permission and obligation in conclusions:

*r1 : prerequisites(Student,Lesson)* $\Rightarrow_{perm}$ *enroll(Student,Lesson,Semester,Year)*
*r2 : enroll(Student,Lesson,fall,Year)* $\Rightarrow_{obl}$ *enroll(Student,Lesson,Semester,Year+1)*
*r3 :total_semester_units(Student,Lesson,Year,Summer), Sum>35*$\Rightarrow_{obl}$ *enroll(Student,Lesson,Semester,Year) r2>r1, r3>r1*

Our system offers the capability to decide automatically if a student has the permission or not to enroll in a particular course given in a particular semester, by running the logic programs with the corresponding regulations and translating dynamically the university RDF data, which are related to this particular query, into logical facts.

## References


Antoniou, G., and Arief, M. 2002. Executable declarative business rules and their use in electronic commerce. In *SAC '02: Proceedings of the 2002 ACM symposium on Applied computing*, 6-10. New York, NY, USA: ACM Press.

Antoniou, G., and Bikakis, A. 2007. DR-Prolog: A System for Defeasible Reasoning with Rules and Ontologies on the Semantic Web. *IEEE Transactions on Knowledge and Data Engineering* 19(2):233-245.

Antoniou, G.; Billington, D.; Governatori, G.; and Maher, M. J. 2001. Representation results for defeasible logic. *ACM Transactions on Computational Logic* 2(2):255-287.

Antoniou, G.; Billington, D.; Governatori, G.; and Maher, M. J. 2006. Embedding defeasible logic into logic programming. *Theory Pract. Log. Program.* 6(6):703-735.

Governatori, G., and Rotolo, A. 2004. Defeasible logic: Agency, intention and obligation. In *DEON*, 114-128.

Governatori, G.; Dumas, M.; ter Hofstede, A. H. M.; and Oaks, P. 2001. A formal approach to legal negotiation. In *International Conference on Artificial Intelligence and Law*, 168-177.

Governatori, G.; Rotolo, A.; and Sartor, G. 2005. Temporalised normative positions in defeasible logic. In ICAIL '05: *Proceedings of the 10th international conference on Artificial intelligence and law*, 25-34. New York, NY, USA: ACM Press.

Governatori, G. 2005. Representing business contracts in RuleML. *Int. J. Cooperative Inf. Syst.* 14(2-3):181-216.

Grosof, B. N. 2004. Representing e-commerce rules via situated courteous logic programs in RuleML. *Electronic Commerce Research and Applications* 3(1):2-20.

Nute, D. 1994. *Handbook of logic in artificial intelligence and logic programming*, volume 3. Oxford University Press. Chapter Defeasible logic.