Ontologies in F-logic

Jürgen Angele and Georg Lausen

1Ontoprise GmbH
Amalienbadstraße 36
D-76227 Karlsruhe
angele@ontoprise.de

2Universität Freiburg
D-79098 Freiburg
lausen@informatik.uni-freiburg.de

Summary. F-logic ("F" stands for "Frames") combines the advantages of conceptual high-level approaches typical for frame-based languages and the expressiveness, the compact syntax, and the well defined semantics from logics. The salient features of F-logic include signatures, object identity, complex objects, methods, classes, inheritance and rules. We give an overview of the syntax and the intuitive semantics of F-logic. We discuss the semantics behind and the different ways how F-logic has been implemented. The language primitives are further demonstrated by discussing a real world application resulting in the ontology based configuration tool OnKo developed for the German Telecom.

2.1 Introduction

Conceptual (or Ontology) modelling deals with the question of how to describe in a declarative and abstract way the domain information of an application, its relevant vocabulary, and how to constrain the use of the data, by understanding what can be drawn from it. Corresponding abstract representation languages support the understanding of such descriptions, their rapid development, their maintenance and their reuse.

Conceptual modelling has a long history in the area of database systems, starting with the Entity/Relationship(ER) model [Che76]. Entitytypes represent sets of homogeneous entities by specifying attributes and their range. Entitytypes are instantiated by concrete entities with individual values for the attributes.
Relationships types specify the domain and range of relations between entities. Relationships types are instantiated by concrete relations between entities. By this way a simple schema (or ontology) for a relational databases may be given. Later on, this restricted set of concepts has been extended to the EER model (Extended Entity Relationship) [RES94] by adding modelling primitives like specialisation, grouping etc. in a similar way to modelling languages like UML and ODMG which have been developed for object-oriented models in the context of software engineering.

From the beginning, for database systems languages have been developed which allow to express complex relationships (constraints) between entities/objects and to query the database. For efficiency reasons traditional database query languages (relational algebra and calculus, SQL) have a rather limited expressiveness. However, when database systems approached knowledge-based applications, rule-base languages became attractive. Datalog [AHV95] is the most famous language paradigm in this context. Datalog has a model-theoretic semantics and introduced general recursion while still allowing powerful optimization techniques. However, Datalog is a logical rule language without function symbols and restricted negation and thus still has a limited expressive power. In particular, when object-oriented database systems became popular, function symbols became mandatory because of their ability to give object identities a logical semantics. From a logical point of view, these languages are variants of Horn-Logics with function symbols and negation in the rule bodies. In this paper we will elaborate on F-logic [KLW95], which accounts in a clean and declarative fashion for most of the structural aspects of frame-based and object-oriented languages. A general overview of languages in this context is given in [L99]. Similar languages have also been introduced for semi-structured, respective XML-databases (e.g. [GPQ97], [M01]).

F-logic (“F” stands for “Frames”) combines the advantages of the conceptual high-level approach typical for frame-based languages and the expressiveness, the compact syntax, and the well defined semantics from logics. The original features of F-logic [KLW95] include signatures, object identity, complex objects, methods, classes and inheritance. Implementations of F-logic (Florid [LHL98], Ontobroker [DEFS99], Flora [YK00]) introduced extensions and restrictions of the original features to make F-logic a powerful and efficient language for the respective intended application domain.

In this paper we will look on F-logic as a language for ontology applications. In the following we first give an overview of the syntax and the intuitive semantics of F-logic by concentrating on examples which allow the presentation of the various concepts in a concise way. It follows a discussion of the semantics and the different ways how F-logic has been implemented. Finally we present an example of an ontology application based on F-logic. This example is based on OntoBroker™, an implementation of F-logic specific for the ontology domain.