FLORID: A Prototype for F-Logic

J. Frohn R. Himmeröder* P.-Th. Kandzia G. Lausen C. Schlepphorst*
Institut für Informatik, Universität Freiburg i. Br., Germany
{frohn,himmeroe,kandzia,lausen,schlepph}@informatik.uni-freiburg.de

1 System Overview

FLORID[1] – F-LOgic Reasoning In Databases – is a deductive object oriented database system incorporating F-logic [3] as a data definition and query language and combining the advantages of deductive databases with the rich modelling capabilities of object oriented concepts.

F-logic provides complex objects, uniform handling of data and metadata, rule-defined class hierarchy and signatures, non-monotonic multiple inheritance, equating of objects by rules and variables ranging over methods and classes. Moreover, FLORID extends F-logic by *path expressions* [2] to facilitate object navigation.

Because of its powerful properties F-logic cannot be mapped onto a relational data model by transforming classes into relations. Hence, it was a major design goal to use an object oriented data representation in FLORID. All information about one object is stored in an *object frame* which has a unique OID. Several (external) object names may refer to the same OID, thus defining an equivalence relation between objects. Frames consist of *slots* for method applications or class membership enabling schema modifications during the evaluation by deriving new signatures, subclass relationships or class memberships. The collection of all frames forms an *object world*.

A program is a set of rules including facts and queries. It is read from a file, parsed and checked for safety rule by rule. Queries are answered immediately whereas facts and rules are collected internally. The evaluation of this internal program has to be started explicitly by a system command.

The evalution of rules in FLORID follows a bottom-up strategy which is extended by the trigger semantics from [3] to handle multiple inheritance. The T_P-Operator induced by an program P applies all rules of P once to the current object world and adds all derived facts. T_P is iterated until a fixpoint is reached. At a fixpoint, the evaluation looks for an active trigger, i.e. a class defining an inheritable method and a goal matching a member of that class for which this method is not overridden yet. One such trigger is fired (inserting the fact) and T_P is iterated again until the next fixpoint is reached. The model is established when no more active trigger exists. If several triggers are active, firing one can deactivate others. Hence, the model depends on the order triggers are selected, thus (non-deterministically) choosing one canonic model of P (see [3], chapter 14).

To support flexible extension and customization of FLORID, all important system components are available as *interface objects* at the FLORID interface. The user may deliberately create and combine these components to build a system matching his needs. System control and database language are integrated, all inputs follow F-logic syntax.

FLORID's basic interface is a Prolog-like text shell, where the user can control all system parameters by sending messages to interface objects. Alternatively, there is a Tcl/Tk GUI and an integration into the emacs environment [1]. FLORID can also be used on-line:
http://www.informatik.uni-freiburg.de/~dbis/flsys/

2 Demonstration of FLORID

We show the basic usage and some more involved features like rule-based handling of meta data, negation and maintaining equality between objects by a simple example from the field of integrating heterogeneous databases.

References

[1] J. Frohn, R. Himmeröder, P.-T. Kandzia, and C. Schlepphorst. FLORID, Version 1.0, user manual, 1996. Available from ftp://ftp.informatik.uni-freiburg.de/pub/florid/manual.ps.gz.

[2] J. Frohn, G. Lausen, and H. Uphoff. Access to objects by path expressions and rules. In J. B. Bocca, M. Jarke, and C. Zaniolo, editors, *Proc. Intl. Conference on Very Large Data Bases*, Santiago de Chile, 1994.

[3] M. Kifer, G. Lausen, and J. Wu. Logical foundations of object-oriented and frame-based languages. *Journal of the ACM*, 42(4):741–843, July 1995.

*The work of these authors is supported by the Deutsche Forschungsgemeinschaft, La 598/3-2.

[1] FLORID is available via ftp.informatik.uni-freiburg.de/pub/florid.

Proceedings of the 13th International Conference on Data Engineering
1063-6382/97 $10.00 © 1997 IEEE