A Deductive and Object-Oriented Database System: Why and How ?

Laurent Vieille
BULL Research
BP 68 - Rue Jean Jaurès
F 78 340 Les Clayes sous Bois - FRANCE
e-mail : L.Vieille@frcl.bull.fr

This talk will outline the *principles, the architecture and the potential target applications* of a Deductive and Object-Oriented Database System (DOOD). Such systems combine the novel functionalities (relying on the associated technology) developed in deductive database projects, the ability to manipulate the complex objects appearing in many applications and the architectural advances achieved by Object-Oriented DBMS's.

The deductive approach to information modelling appears to offer unmatched characteristics for the development of new information systems.

The ability to specify (recursive) deduction rules, integrity constraints and triggers on (declarative) conditions, brings so-called *knowledge-independence.* While traditional data base management systems provide data - independence, i.e. the ability to represent and manipulate (factual) data independently from the applications, deductive-based systems are able to represent and manipulate *intensional* information, such as *deduction and integrity rules*, independently from the applications. Knowledge independence (1) simplifies the task of the application programmer who does not need to include tests in his application (defensive programming) to guarantee the soundness of his transactions; (2) guarantees that different applications have the same notion of a derived relationship (e.g., of the notion of a good partner, as a company not making business with a bad partner), and enforce the same set of semantic constraints (e.g. that financial transactions enforce a set of regulations specified by an independent body); (3) makes much easier the modification of these deduction and integrity rules (e.g. to reflect regulation changes), in comparison to an imperative encoding.

SIGMOD /5/93/Washington, DC,USA

Deductive databases differ from extensions to relational systems (such as triggers and iteration operators) by the *declarativity* of their language, the *modularity* with which deduction and integrity rules can be specified and updated, and, thanks to the formal foundations on which they rely, their ability to support a *new class of knowledge manipulation tools.* Thanks to the declarativity of the language, the user can write one assertion as specifying one integrity constraint, instead of a set of triggers implementing this constraint (the correctness of this set being then questionable). Thanks to the modularity of the language, the rules computing the amount of tax to be paid by an individual may be updated independently from each other. Thanks to the foundation on predicate calculus, new tools for knowledge manipulation such as consistency checkers and explanation facilities can be defined, designed and developed.

The integration of this deductive approach together with the object-oriented modelling techniques and the architectural achievements of object-oriented databases opens up a new range of possibilities for the development of information systems. *Data mining activities* are enabled thanks to the combination of a high-level model with a declarative, logic-based language. *Concurrent Design Activities* (e.g., in civil engineering) benefits both from the object management capabilities and the ability to specify semantic integrity rules which have to be enforced by all actors of an engineering or logistic project. *Highly-regulated industries* benefit from the ability of the DOOD system to validate transactions against theses regulations.

The architecture and development of an integrated DOOD system raise a number of technology issues, even when starting from one of the most advanced deductive database prototypes. These issues include the provision of an integrated data model, the use of the deductive framework to optimize queries, the integration of integrity checking in transaction management, etc...