A Deductive, Object-Oriented Model as a Formal Framework for Active Database Environments

Susan D. Urban* Suzanne W. Dietrich
Department of Computer Science and Engineering
Arizona State University
Tempe, Arizona 85287-5406
{urban | dietrich}@asuvax.eas.asu.edu

Abstract

The successful integration of active, deductive and object-oriented databases for the basis of future generation databases is a challenging research objective. The advantages of an active, deductive, object-oriented database (A DOOD) are illustrated through a motivational example, which highlights the expressiveness of the declarative specification of constraints, updates and active rules over intensional data. Using a DOOD framework for the expression and execution of active database rules relies on the solution of a number of open research problems, which are outlined in this paper. The paper then discusses the specific research directions of a project at ASU, known as *A DOOD RANCH*, having as its goal an object-oriented database system that efficiently and correctly processes queries, constraints, and active rules that involve extensional and intensional data.

1 Introduction

The areas of deductive, object-oriented, and active databases have primarily developed as separate areas of research. Although each type of database has made its own contributions to the advancement of database technology, future database applications will require an environment that supports features provided by each type of database. Recent work on deductive, object-oriented models (DOOD's) has demonstrated the feasibility of using logic as a formal basis for object-oriented systems [1]. Other related work has also demonstrated the feasibility of integrating rule processing and object-oriented concepts [11]. Unfortunately, systems that fully integrate object-oriented concepts with deductive *and* active rule processing have not yet been adequately defined.

This paper describes the database research challenges that exist in the investigation of a DOOD model as a declarative framework for the expression and execution of active database rules. The presentation is based on our own initial investigation of this area [10] and specifically addresses 1) the definition of an active DOOD model, 2) the development of efficient and complete language evaluation strategies, 3) the development of execution models that accommodate different types of rules, and 4) the development of efficient condition monitoring strategies for active rule processing. We also address the need for design and debugging tools that support the specification of active rules and the analysis of active rule behavior.

The remainder of this paper is structured as follows. A motivational example of an active DOOD model and application is provided in Section 2. Research issues for an active DOOD environment are then outlined

*This research was partially supported by NSF Grant No. IRI-9109195

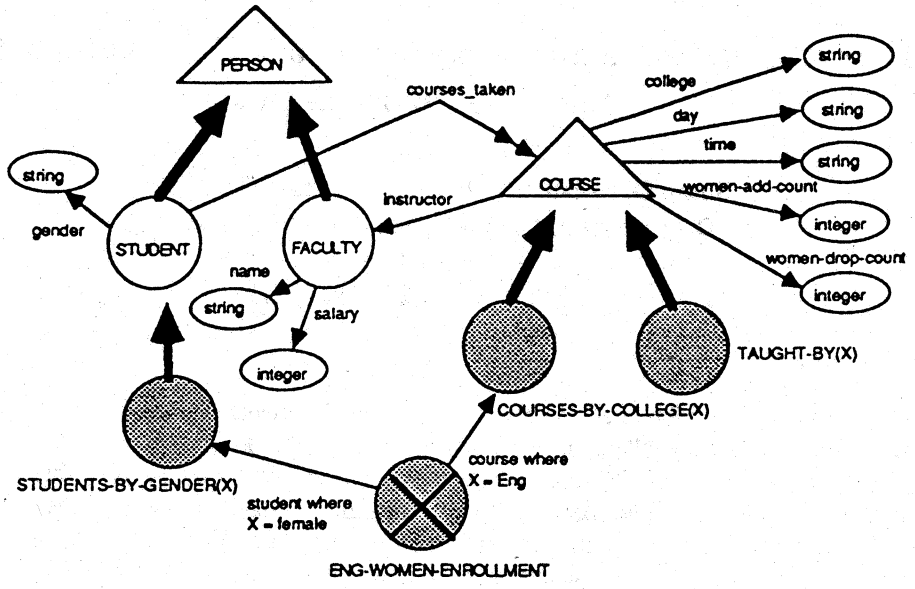

Figure 1: A University Schema

in Section 3. Section 4 provides an overview of our research efforts in this area, followed by a summary of the research challenge in Section 5.

2 A Motivational Example

As an example of the expressive modeling capabilities that are supported by the integration of deductive, active, and object-oriented concepts, consider the semantic schema in Figure 1 for a University enterprise. The graphical notation is that of the IFO semantic data model [4]. Triangles and circles represent abstract objects while ovals represent simple, printable values. Triangles specifically represent the root classes of superclass/subclass hierarchies while circles represent subclasses. Bold arrows are used to define superclass/subclass relationships. Labeled arrows represent attribute definitions, where single-headed arrows define single-valued attributes and double-headed arrows define set-valued attributes. Aggregation classes (i.e., classes that represent relationships between abstract objects) are denoted as circles enclosing an X. We have added the following additional modification to the notation: shaded subclasses and shaded aggregation classes indicate derived classes, where derived subclasses represent a subset of a base class and derived aggregation classes represent a class of newly-generated objects that are a result of a relationship between objects from other base or derived classes. The schema definition therefore consists of an extension and an intension.

The Extensional Schema Definition

Since the structural aspects of object models are fairly well-defined, the extensional definition of the University schema can be expressed using an object logic model such as that described in [8]. As shown in Figure 1, Student and Faculty are defined to be subclasses of Person. Students take many courses as indicated by the courses-taken set-valued attribute. Each Course has a Faculty object as an instructor as well as several other simple-valued attributes. The women-add-count and women-drop-count attributes of a Course are attributes that keep track of the number of women that enroll in a Course and the number of women that drop a Course, respectively. Individual abstract objects are assumed to have unique identifiers.

The Intensional Schema Definition

Based on the extensional schema definition, a rule-based language can be used to express derived attributes and subclasses. For this particular example, we are using a language associated with a model known as CDOL (Comprehensive, Declarative Object Language) [16]. Formal details of the model are omitted here due to space limitations (and since complete formal details are part of the research challenge!).

As an example, if we frequently need to form subclasses of the Course class based on the instructors that teach courses, we can specify the class definition as follows:

Taught-by(Inst-name:N):C ← Course:C[instructor:F], Faculty:F[name:N]

In this case, the Taught-by derived subclass is parameterized as in [2] so that, given a specific faculty member, the function will derive a subclass of courses taught by that faculty member. The direct specification of such a function within the extension would be impractical given the large number of faculty members that typically exist in a university. A similar case holds for derived subclasses of the Student class that form subclasses of male and female students and for derived subclasses of the Course class that form subclasses based on the college in which courses are offered:

Students-by-gender(gender:G):S ← Student:S[gender:G]

Courses-by-college(college:D):C ← Course:C[college:D]

Note that derived subclasses can also have derived attributes. Specific examples can be found in [16].

The final intensional aspect of the schema involves a derived aggregation class. Specifically, we need to form a view that defines a relationship between female students and the engineering courses in which they enroll. Using the notion of *imaginary objects* from [2], we can express such a derived class in CDOL as follows:

Eng-Women-Enrollment:gen_O(S,C)[student:S, course:C] ←
Students-by-gender("female"):S[courses-taken:T], Courses-by-college("Eng"):C, C isin T

The above statement defines the Eng-Women-Enrollment derived class to be a set of generated objects (generated by the gen_O object generating function) that represents the specified relationship between the derived subclasses of Student and Course.

Constraints, Updates, and Active Database Rules

The above schema provides a basis for an important point with regard to the modeling strength gained through the use of a deductive, object-oriented model: the use of the deductive features of the model provides an expressive framework that can enhance the specification of constraints, updates, and active rules. For example, it is important to insure that an instructor is not scheduled to teach two courses that are offered on the same day at the same time. Although the constraint can be directly stated over the extensional part of the schema, the constraint is more concisely stated over the intensional schema:

CONSTRAINT: for all C1 in Taught-by(X), there does not exist a C2 in Taught-by(X) such that
(C1 $\neq$ C2) implies (C1.day = C2.day and C1.time = C2.time)

The use of intensional data therefore provides a modularization feature for the expression of constraints. The advantage of such a modularization feature in a semantic modeling framework were originally described with the use of derived data in the functional data language DAPLEX [20].

The use of intensional data in the specification of active rules also demonstrates useful modeling capabilities. Assume that we need to maintain a count of the number of women that enroll in engineering courses and the number of women that drop engineering courses. We also need to notify the Society of Women Engineers (SWE) when women enroll in engineering courses so that membership information can be mailed to potential SWE members. When women drop engineering courses, SWE sends a questionaire to those women regarding their reasons for dropping an engineering course. Ideally, we would like to have active database capabilities that can automatically monitor this situation using rules such as the following:

EVENT: insert Eng-Women-Enrollment(O[student:S, course:C])
ACTION: C:[women-add-count:A $\leftarrow$ (A+1)],
Send message to SWE that student S has enrolled in an Eng. course

EVENT: delete Eng-Women-Enrollment(O[student:S, course:C])
ACTION: C:[women-drop-count]:D $\leftarrow$ (D-1)],
Send message to SWE that student S has dropped an Eng. course

The above active rules invoke changes to the extension of the database that are initiated by monitoring a change to an intensional part of the database.

The database intension can also be used to express object update conditions. In CDOL, for example, the following update example gives a 10% raise to all faculty that make less than $40K:

F[salary:S ← (S * 1.1)] ← Faculty:F[Salary:S], S < 40,000.

Note that update rules in CDOL are expressed as queries on a one-time basis, defining a set of objects to be updated and the type of update operation(s) to be performed. Updates are also required to be deterministic according to an admissability test described in [17].

3 Active DOOD Research Challenges

This section outlines some of the research challenges that exist in the development of an active DOOD model. Due to space limitations, this section does not present a complete survey of related work. The purpose of this section is to simply highlight research challenges that range from the definition of the model itself to the investigation of efficient techniques for language evaluation and active rule processing.

The Definition of an Active DOOD Model

The critical issues to address in the definition of an active DOOD model are:

1. *Declarative, rule-based expression of derived attributes and classes.*

   Object-oriented schemas should not be limited to the description of extensional data. The ability to incorporate deductive capabilities into the schema definition creates an expressive environment that allows the needs and views of different users to be addressed. Abiteboul and Bonner have addressed this issue in their elaboration of the need for view facilities in object models [2], addressing issues such as virtual classes and imaginary objects. Work in defining active DOOD models should focus on a declarative framework for the expression of intensional schema components, including recursive data definition. The definition of derived data should be as central to the schema definition as the definition of stored classes, opening up additional research issues to be addressed with respect to maintaining materialized views, performing view updates, and using view definitions in the expression of constraints, rules, and updates.

2. *Declarative expression of constraints.*

   The expression of constraints is an issue that is often not addressed in object models. The general view is that constraints should be encapsulated within the operations associated with object classes. Constraints are an important source of knowledge, however, about the relationships between objects. The declarative expression of constraints is important for establishing a sound notion of a consistent database state, for addressing optimization issues in constraint checking, and for reasoning about

relationships between objects, possibly for the purpose of constraint propagation. Constraints in an active DOOD model should be viewed as a central component of the schema definition, including the expression of constraints involving derived data as illustrated in our example.

3. *Declarative expression of active database rules.*

   An active DOOD model should provide a language that supports the expression of different types of rules. Rules that maintain constraints, for example, are inherently different from rules that serve as alerters and triggers. Integrity maintenance rules should also be closely associated with the constraints on which they are based and preferably derived directly from constraints as described in [21]. Furthermore, if virtual classes and attributes are to be viewed as first-class schema components, rule conditions involving derived data and rule event detection involving changes to intensional data should be important components of active rule specification.

4. *Declarative, ad-hoc expression of object updates.*

   One of the most critical omissions in current proposals for object logics is the issue of object updates. The update issue for objects is closely related to the specification of constraints and the way in which those constraints are maintained when updates occur. We feel that active rules for integrity maintenance provide an appropriate way of handling the occurrence of ad-hoc update commands and that an active DOOD model must provide a language framework in which updates work together with constraints and active rules. One such proposal is provided in our own work in this area [17]. A thorough investigation of update semantics in an active DOOD model is, however, an open issue.

Language Evaluation Issues

Although language evaluation and optimization is well-understood in both relational (algebraic) and deductive database (logical) frameworks, these issues are still relatively unexplored in the DOOD framework. For example, an investigation of the optimization of object-oriented recursive queries [19] demonstrates that the traditional relational approach of performing selections before performing joins and/or recursion may not be an appropriate optimization technique in object-oriented queries due to the complexity of methods and path expressions. DOODs inherit the complexity of deductive database evaluation of recursive rules in the presence of negation and aggregation, and add the additional complexity of set-valued properties, which can exhibit non-monotonicity problems similar to that of negation. The deductive database optimization for recursive rules may be applicable to the DOOD context [3] but the development of new techniques for the evaluation of complex objects, in the presence of object identity and sets, in a logic programming environment may likely be necessary.

Execution Model Issues

Most current prototypes of active databases have not provided formal semantics for the the execution of active rules. Furthermore, existing execution models do not make a clear distinction between the execution of different types of rules, other than to assign rule priorities. The work in [6] provides one of the most comprehensive descriptions to date of a rule execution framework that subsumes most previous work on rule execution models. We feel that an important line of research with respect to an active DOOD environment is to establish a formal definition of an execution model. Since constraints and integrity maintenance rules are an important part of the active DOOD model, the execution model should provide clear semantics for how integrity maintenance rules work together with other rules such as alerters and triggers. The use of rule sets and complex rule triggering specifications [11] should also be incorporated into the complete formal definition of active rule processing.

Condition Monitoring Issues

The condition monitor in an active database system is responsible for detecting which conditions, such as alerters, triggers and integrity constraints, are satisfied. Researchers developing active database systems have recognized the necessity for an efficient condition monitor and have proposed several approaches for condition monitoring in relational databases or relational databases extended with objects, limiting conditions to SPJ (select, project, join) expressions defined over extensional relations. Other researchers have addressed condition monitoring in active deductive databases [12, 13], which allow conditions expressed over intensional relations that may be defined in terms of select, project, join, union, stratified negation and recursion. The investigation of condition monitoring in the context of an active DOOD environment will involve monitoring declarative conditions expressed over both extensional and intensional data that includes the rich data structuring capability of object-orientation.

Development Environment Issues

One of the most important aspects in the acceptance of active database systems, in general, will be the use of tools that can support the development of active database applications. Research such as that in [7, 21] has already addressed the need for the automatic generation of integrity maintenance rules from constraints. An even more important issue, however, lies in the investigation of active rule behavior. Rule triggering patterns can often be unpredictable and can lead to anomalous rule behavior such as infinite rule triggering, rule triggering that causes conflicting updates, and rule triggering that produces non-deterministic results (i.e., the rule confluence problem). The work in [5] has provided initial results in this area. There is significantly more work that can be done, however, by examining the semantics of rule conditions. The development of tools that can guarantee some degree of correctness and predictable behavior of active

database rules represents a challenging aspect of active DOOD research.

4 A DOOD RANCH

In support of the above research issues, we are currently investigating the integration of active, deductive and object-oriented databases within a project that we appropriately (here in Arizona) refer to as *A DOOD RANCH* (Active, Deductive, Object-Oriented Databases - Relating Action, Negation, Constraints and Horn rules). The joint research project is the synergism of our own initial investigations in the area of updates, integrity maintenance and rule analysis in a DOOD environment [15, 21, 22], complete evaluation strategies for declarative logic programs[9] and efficient condition monitoring in active deductive databases[13] that also supports the maintenance of materialized views [14].

The goal of *A DOOD RANCH* is the development of an active database environment that uses a deductive, object-oriented model as a formal basis for the declarative specification and efficient execution of active rule processing. In particular, we are investigating the extension of a DOOD model [16] to provide a more complete framework for support of active and deductive database applications, developing a rule language that supports alerters and triggers in addition to integrity maintenance rules. We are also investigating efficient evaluation strategies for the DOOD rule language, deriving techniques from deductive database optimizations and enhancing those techniques with object-oriented considerations. Existing bottom-up and top-down evaluation techniques must be re-evaluated in the context of object-oriented optimization issues, such as object identity, set-valued attributes, and the use of path expressions in the rule language.

We are also extending and formalizing the rule execution model to create an active processing environment that supports deductive rules and active database rules in general. The research is specifically focused on the integration of a nested transaction processing model for active rules with an update propagation approach to condition monitoring [13]. The condition monitoring algorithm, which was originally developed for active deductive databases, must be re-examined in the context of an active DOOD environment. This investigation may provide additional tools for optimization. For example, the introduction of object identity with the use of object generating functions may provide useful techniques for detecting changes. The final result of our efforts in this aspect of the research will produce a formal execution model for the use of integrity maintenance rules together with general active database rules, supported by efficient techniques for detecting changes in conditions involving extensional and intensional data.

The final aspect of our work involves the development of a supportive environment for the design, testing, and analysis of active database applications. This aspect of the research will redefine, consolidate,

and extend our current research base on the analysis of constraints and the detection of anomalous rule behavior to create an environment that can assist in the design and use of rules in the active DOOD framework. The development of such an environment will be important to the acceptance of active databases as a viable technology.

5 Summary

Although significant results have been achieved within the individual areas of active, deductive, and object-oriented database systems, database applications of the future will require the successful integration of all three database paradigms. We have provided a motivational example of an active DOOD application to illustrate the expressive capabilities and features of such a rich environment. Achieving a declarative, object-based framework that efficiently processes queries, constraints and active rules, however, will require the investigation of many new research directions as outlined in this paper. We then discussed the specific goals for our research efforts in the integration of active, deductive and object-oriented databases to provide an object-oriented database system that efficiently and correctly processes queries, constraints, and active rules that involve extensional and intensional data.

References

[1] S. Abiteboul. Towards a Deductive Object-Oriented Database Language. In [18], pp. 453-472.

[2] S. Abiteboul and A. Bonner. Objects and Views. In *Proceedings of the 1991 ACM SIGMOD Conference*, 1991, pp. 238-247.

[3] S. Abiteboul, S. Grumbach, A. Voisard and E. Waller. An Extensible Rule-Based Language with Complex Objects and Data-Functions. *Proc. of the 2nd Int. Workshop on Database Programming Languages*, 1989, pp. 298-314.

[4] S. Abiteboul and R. Hull. IFO: A Semantic Database Model. In *ACM TODS*, 12, 4, December 1987.

[5] A. Aiken, J. Widom, and J. M. Hellerstein. Behavior of Database Production Rules: Termination, Confluence, and Observable Determinism. In *Proceedings of the 1992 ACM SIGMOD Conference*, 1992, pp. 59-68.

[6] C. Beeri and T. Milo. A Model for Active Object-Oriented Database. In *Proceedings of the 17th International Conference on Very Large Data Bases*, 1991, pp. 337-349.

[7] S. Ceri and J. Widom. Deriving Production Rules for Integrity Maintenance. In *Proceedings of the 16th International Conference on Very Large Data Bases*, 1990, pp. 566-577.

[8] W. Chen and D. S. Warren. C-Logic for Complex Objects. In *Proceedings of the ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, 1989, pp. 369-378.

[9] S. W. Dietrich. Extension tables: memo relations in logic programming. In *IEEE Symposium on Logic Programming*. 1987, pp. 264-272.

[10] S. W. Dietrich, S. D. Urban, J. V. Harrison and A. P. Karadimce. A DOOD RANCH at ASU: Integrating Active, Deductive and Object-Oriented Databases. *IEEE Data Engineering Bulletin: Special Issue on Active Database Systems*, December 1992, pp. 40-43.

[11] N. Gehani and H. V. Jagadish. Ode as an Active Database: Constraints and Triggers. In *Proceedings of the 17th International Conference on Very Large Data Bases*, 1991, pp 327-336.

[12] J. Harrison. *Condition Monitoring in an Active Deductive Database*. Ph.D. Dissertation, Dept. of Computer Science and Engineering, Arizona State University, August 1992.

[13] J. Harrison and S. W. Dietrich. Condition Monitoring in an Active Deductive Database. Technical Report 91-022, Dept of Computer Science and Engineering, Arizona State University. Submitted for journal publication.

[14] J. Harrison and S. W. Dietrich. The Maintenance of Materialized Views in a Deductive Database: An Update Propagation Approach. *Proc. of the Deductive Database Workshop* in conjunction with the Joint International Conference and Symposium on Logic Programming, Washington, D.C., November 1992, pp. 56-65.

[15] A. Karadimce and S. Urban. Diagnosing Anomalous Rule Behavior in Deductive Databases with Integrity Maintenance Rules. *Proc. of the Third International Workshop on Foundations of Models and Languages for Data and Objects*, Aigen, Austria, Sept. 1991, pp. 77-102.

[16] A. Karadimce and S. Urban. CDOL: A Declarative Platform for Developing OODB Applications. *Proc. of the International Phoenix Conference on Computers and Communications*, Tempe, AZ, March 1993, pp. 224-230.

[17] A. Karadimce and S. Urban. A Framework for Declarative Updates and Constraint Maintenance in Object-Oriented Databases. *Proc. of the Ninth Int. Conf. on Data Eng.*, Vienna, April 1993, pp. 391-398.

[18] W. Kim, J. Nicolas, and S. Nishio (editors). *Deductive and Object-Oriented Databases*. North-Holland, The Netherlands, 1990.

[19] R. Lanzelotte, P. Valduriez and M. Zait. Optimization of Object-Oriented Recursive Queries using Cost-Controlled Strategies. *Proc. of the ACM SIGMOD Conf.*, San Diego, June 1992, pp. 256-265.

[20] D. Shipman. The Functional Data Model and the Data Language DAPLEX. *ACM TODS*. vol. 6, no. 1, March 1981, pp. 140-173.

[21] S. D. Urban and M. Desiderio. CONTEXT: A Constraint Explanation Tool. In P. Chen, editor, *Data and Knowledge Engineering*, volume 8, North-Holland, 1992, pp. 153-183.

[22] S. D. Urban and B. L. Lim. An Intelligent Framework for Active Support of Database Semantics. to appear in *Advances in Databases and Artificial Intelligence: The Landscape of Intelligence in Database and Information Systems*, JAI Press, L. Delcambre and F. Petry (editors), 1993.

Proceedings of the Workshop on Combining Declarative and Object-Oriented Databases

(In cooperation with ACM SIGMOD)

Editor: Inderpal Singh Mumick
AT&T Bell Laboratories

May 29, 1993
Washington D. C., USA.

Special Interest Group for the Management of Data
(ACM SIGMOD)