The NAOS System

C. Collet and T. Coupaye
LGI-IMAG, University of Grenoble, BP 53 38041 Grenoble cedex 9, France.
{Christine.Collet, Thierry.Coupaye}@imag.fr

1 Motivations and Goals

It is now recognized that integrating a production rule facility into a database system provides a powerful mechanism for designing or implementing several features. The Native Active Object System(NAOS)[1] incorporates an active behavior within the object-oriented database management system O_2 [2]. It has been developed for the GOODSTEP platform[1][3] dedicated to support the construction of customized Software Development Environments(SDE) with different tools.

2 Our approach

Active behavior has been incorporated in O_2 as Event-Condition-Action rules or active rules. Rules are defined as the same level than a class or an application (a set of programs). This allows program execution and data manipulation, including method calls, to be driven on rules.

Events may be generated by manipulations of entities or parts of entities (objects or values) which may be persistent or not. The moment of generation can be either *before or after* the actual operation depending on the type of the event. Events may also be generated by executions of transactions, programs or applications.

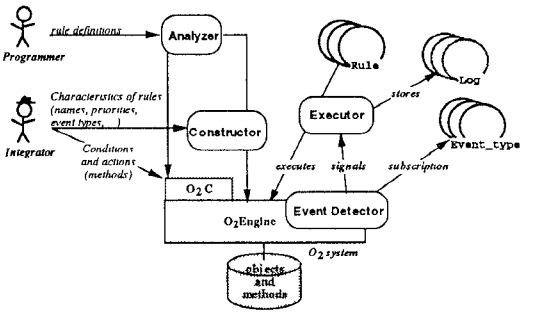
A condition is defined as an O_2SQL query and its result is visible to the action part. An action is an O_2C code which may operate on persistent and transient entities or activate/deactivate rules.

NAOS supports *immediate* and *deferred* rules. Immediate rules are scheduled for immediate execution while deferred rules accumulate events until the end of the current transaction. The action part of a rule is always executed immediately after the condition, given that the latter is satisfied. Triggered rules are executed in a new execution cycle distinct from the one in which the triggering events were produced: nested cycles for immediate rules and consecutive cycles for deferred rules. If several rules have to be executed in the same execution cycle, they are executed according to their priorities.

At runtime each rule is associated with a *delta structure* containing data related to the triggering operation(s). Such a delta structure can be named and used in the condition and action of the rule for manipulating the data.

3 Architecture

The following figure shows the main actors of NAOS. The `analyzer` is able to communicate with programmers who want to define and manipulate rules in an easy way using the rule language. It principally analyzes a rule definition and produces: (i) an intermediate representation of the rule sent to the constructor, and (ii) some O_2C code representing the condition and action of the rule.



The `constructor` creates a persistent representation of rules. It offers a low level interface well adapted for software integrators and developers who needs basic reactive capabilities for supporting some functions of the system they want to implement and do not need a high level rule definition language.

The `event detector` detects primitive events in an efficient way since it is incorporated in the O_2Engine. When an application A is launched the rules of the schema to which A belongs become active: a C++ snapshot of the rule and event type definitions is created in order to speed up rule execution and, subscriptions are sent to the `event detector` for each event types. For each subscription, the address of a handling function is supplied. This function wakes up (*signals*) the `executor` who processes the concerned rule(s) taking into account coupling modes, rule cascading, rule priorities, and the calculation of net effect.

4 Current status

NAOS is 20000 lines of C and C++ and runs as a library of O_2. A new release is expected for Spring 1995. For that we developped a composite event detector. Also studies on user-defined and temporal events are underway.

Acknowledgment

We would like to thank P. Habraken, T. Svensen, A. Chabert and T. Lenepveu for their invaluable contributions to the design and implementation of NAOS.

References

[1] C. Collet, T. Coupaye, and T. Svensen. NAOS efficient and modular reactive capabilities in an object-oriented database system. In *Proc. of the 20th Int. Conf. on Very Large Data Bases*, Santiago, Chile, September 1994.

[2] F. Bancilhon, C. Delobel, and P. Kanellakis. *Building an Object-Oriented Database - The story of O_2*. Morgan Kaufmann, 1992.

[3] GOODSTEP Team. The GOODSTEP Project: General Object-Oriented Database for Software Engineering Processes. In *Proc. of the Asia-Pacific Software Engineering Conference, Tokyo, Japan*, pages 410–420. IEEE Computer Society Press, 1994.

[1] GOODSTEP is the Esprit-III project No 6115