WIDE - A Distributed Architecture for Workflow Management

Stefano Ceri
Politecnico di Milano
Italy
ceri@elet.polimi.it

Paul Grefen
University of Twente
The Netherlands
grefen@cs.utwente.nl

Gabriel Sánchez
Sema Group sae
Spain
gsg@sema.es

Abstract

This paper presents the distributed architecture of the WIDE workflow management system. We show how distribution and scalability are obtained by the use of a distributed object model, a client/server architecture, and a distributed workflow server architecture. Specific attention is paid to the extended transaction support and active rule support subarchitectures.

1. Introduction to WIDE

Workflow management is currently considered a major application domain for information technology. To provide reliable data processing in workflow applications, database systems have become important as the basis for workflow management systems.

In the WIDE project, extended database technology is developed to serve as the basis for a commercial next-generation workflow management system. In WIDE, extending database technology focuses on extended transaction management and active rule support. Extended transaction management provides flexible and reliable workflow process semantics, active rule support provides reactive behavior to cope with workflow events. These advanced features are reflected in a rich workflow model [CG96] and specification language [CV96]. Design support is developed in WIDE to enable workflow application designers to effectively use these features.

WIDE (Workflow on Intelligent Distributed database Environment) is an ESPRIT project, the main contractor and industrial partner of which is Sema Group, a major European software firm. Politecnico di Milano and University of Twente are the academic partners. ING Bank, a major Dutch bank, and Hospital General de Manresa, a mid-sized Spanish hospital, are the end-user partners in the consortium.

For reasons of brevity, this paper concentrates on database technology aspects of WIDE. The organization of this short paper is as follows. Section 2 presents the overall WIDE architecture. Section 3 shows how distribution is handled in this architecture. Sections 4 and 5 discuss extended transaction management and active rule processing in the WIDE architecture. We end the paper with conclusions and a few words on future work.

2. The WIDE architecture

The WIDE architecture is designed to support next-generation workflow management functionality in a distributed environment. The architecture is based on a commercial database management system as implementation platform and extends this system with extended transaction management and active rule support. The design of the architecture is ruled by three major design decisions:

- the database management functionality should be orthogonal to the workflow management functionality,
- the transaction support functionality should be orthogonal to the rule support functionality,
- both extended database functionality and workflow management functionality should be independent from the underlying database management system.

The resulting architecture of the WIDE workflow management system is shown in Figure 1. The lowest layer of the architecture is formed by the commercial database management system (DBMS). In the project context, Oracle has been chosen as database platform. The DBMS layer is shielded from the upper layers by means of the basic access layer (BAL). The BAL provides an object-oriented database access interface to its clients and maps this to the relational interface of the DBMS to obtain data persistence. The mapping logic is generated by a translator that translates object-oriented data specifications into relational database manipulation operations.

Above the BAL, the server layer is located. In this layer, the database functionality of the DBMS is extended by a transaction support module and an active rule sup-

0-8186-7849-6/97 $10.00 © 1997 IEEE

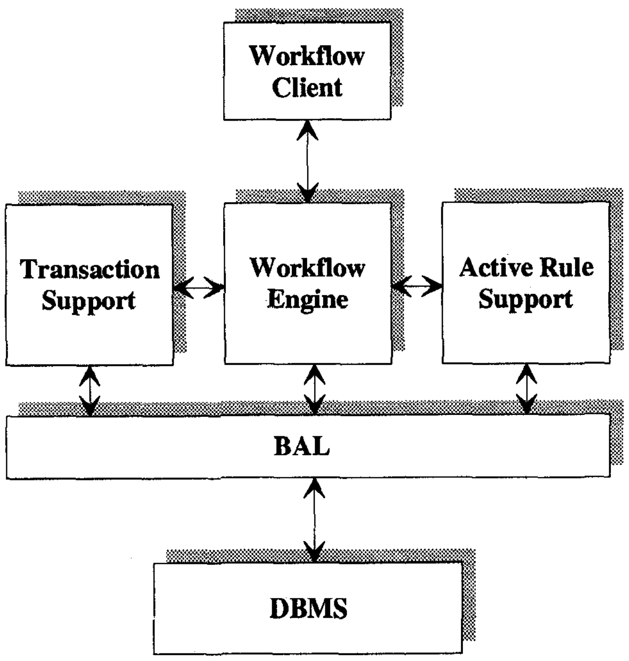

Figure 1: Global WIDE architecture

port module. These two modules are fully orthogonal and are discussed in more detail in Sections 4 and 5. Also located in the server layer, the workflow engine provides the 'heart' of the workflow management system. It uses the BAL for database access, the transaction support module to provide advanced transactional contexts for its operation, and the active rule support module to handle reactive workflow behavior.

Finally, in the client layer of the architecture, the workflow client module provides the interactive interface to the end-users of the workflow system. It communicates only with the workflow engine.

3. Distribution in WIDE

A major aspect of the WIDE architecture is distribution. Distribution is a main issue because of two reasons: workflow management is a distributed application by nature, and distribution opens the way to scalability of the architecture. Distribution is obtained in three different ways: a distributed object model, a client/server database architecture, and a distributed server architecture.

The distributed object model in WIDE is used to create workflow server modules and data objects that can transparently be accessed by multiple processes. These processes may be running on the same machine or on different machines in a network. The object model used conforms to the CORBA standard [OM95, Sie96]. The CORBA interface definition language (IDL) is used to describe the interfaces of the distributed objects. To obtain the mapping logic of the BAL (as discussed in Section 2), an IDL-to-SQL translator has been constructed.

The distributed object model allows for flexible clustering of functionality into processes and flexible allocation of processes to machines. An example is the management of case objects, the objects that contain the data of workflow cases (work items). These case objects can all be managed by one process, or they can be distributed among several processes to distribute the work load. Clearly, the object model provides a powerful means to achieve scalability in the architecture.

In the WIDE architecture, a client/server interface is used between the BAL and the DBMS (see Figure 1). This client/server architecture allows flexible allocation of the low-level database processes and the client processes that use these (located in the workflow engine, the transaction support module, and the active rule support module). In the configuration with Oracle as database platform, the Oracle Call Interface (OCI) has been chosen to realize the interface (see e.g. [Mc96]).

A WIDE workflow management infrastructure can consist of a hierarchy of workflow engines. In such a hierarchy, each engine is associated with a workflow domain. The hierarchy of workflow domains can be derived from the organizational structure in which the workflow is implemented, or from the 'geographical' structure of the organization. The decomposition of a workflow system into a hierarchy of workflow engines provides a means to obtain scalability of workflow applications for large organizations.

4. Transactions in WIDE

Workflow processes usually can be hierarchically decomposed into subprocesses down to the level of individual tasks. In the higher levels of the process hierarchy, process semantics are usually different from those in the lower levels of the hierarchy. For this reason, we have adopted a two-layer transaction model in WIDE [GV96]. The upper layer provides global transactions with 'loose' semantics and is based on concepts from the saga transaction model [GS87]. The lower layer provides local transactions with more 'strict' transactional semantics and is based on the nested transaction model (see e.g. [DH91]). The overall model is constructed such, that the two layers are completely orthogonal.

Given the orthogonal two-layer model, the extended transaction support module in the WIDE architecture consists of two orthogonal submodules supporting global transactions respectively local transactions (see Figure 2).

Global transaction support is provided by the global transaction manager (GTM) process. The GTM manipulates global transaction (GT) objects. Both GTM and GT are CORBA objects, such that they can be accessed trans-

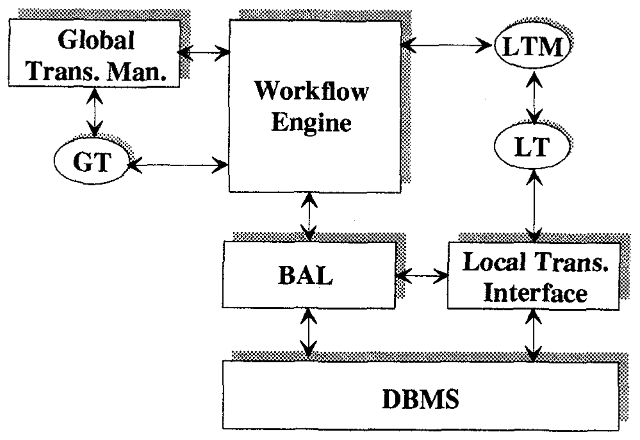

Figure 2: Transaction management architecture

parently from other processes. From a functional point of view, this means that one GTM process can serve multiple workflow engine processes. Global transaction support is completely independent from the underlying database platform.

Local transaction support is provided in two layers. The upper layer consists of the local transaction manager (LTM), which manipulates local transaction (LT) objects. The LTM can be seen as a transaction adapter [BP95]. The upper layer is independent from the underlying database management system, as it only assumes support for a standard 'flat' transaction model and uses logical transaction identifiers and operations. The lower layer of local transaction support consists of the local transaction interface (LTI), which maps logical transaction operations to physical transaction operations and logical transaction identifiers to physical transaction channels. With Oracle as database platform, the Oracle Call Interface (OCI) is used to realize the LTI-DBMS interface. OCI allows multiple concurrent transactions from one client using logon and cursor data areas [Mc96]. As the LTI addresses the DBMS directly, it can be considered part of the BAL layer in the layered architecture shown in Figure 1.

Further details on transaction management in WIDE can be found in [GV96].

5. Active rules in WIDE

Support of reactive behavior is of great importance in workflow management applications, e.g. to support exception handling. A convenient way to model reactive behavior is the use of active rules, i.e. event-condition-action (ECA) rules as they can be found in active database systems [WC96].

In the WIDE conceptual model, we distinguish four event classes: data events, external events, workflow events, and time events [CC96]. Data events are modifications to the workflow data, and can thus be considered

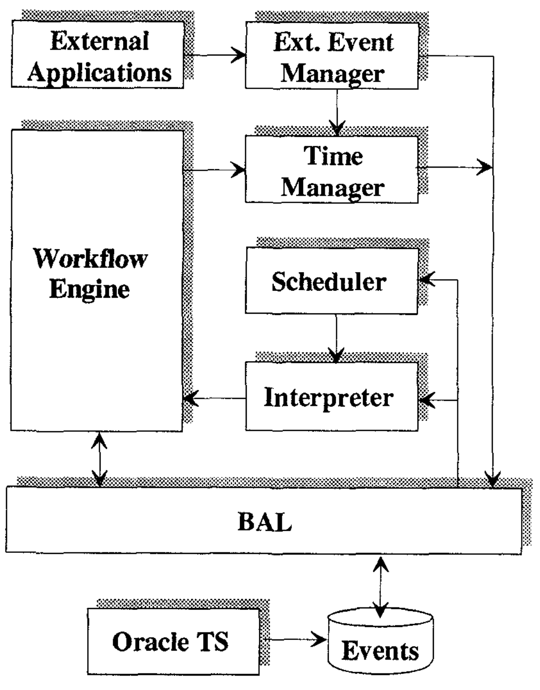

Figure 3: Rule support architecture

workflow process events. External events are raised by external applications used in the workflow context. Workflow events describe the workflow evolution, e.g., starts and ends of workflow tasks. Time events are related to absolute or relative time points in the execution of a workflow.

A decoupled rule execution model has been chosen in WIDE that is orthogonal to the transaction model. Detection of events and execution of rules are performed in a decoupled fashion, i.e. in the context of different local transactions provided by the LTI. This allows for flexible rule handling without too strict execution dependencies, as required in a workflow context.

A slightly simplified architecture of the active rule support module is shown in Figure 3. Event detection is performed by three different modules that record events in the events database (shown as Events in the figure). Data events are captured by low-level triggers that are installed in the database management system (Oracle TS). These triggers react on modifications to the workflow data relations and insert events into the events database. External events are captured by the external event manager. Time events are captured by the time manager, which is also used to generate timestamps for external and workflow events. Both external event manager and time manager insert event records into the events database using the BAL.

Rule execution is divided into scheduling and interpretation of rules. The scheduler inspects the events database (using the BAL) and matches the recorded events to rules. Selected rules are recorded in a to-execute list. This list is next emptied by invoking, for each rule, the rule interpreter. The interpreter is responsible for the evaluation of the condition of a rule and the execution of its action.

To enable distributed access, some submodules of the rule support subsystem are realized as CORBA objects. Further information on the WIDE rule system can be found in [CC96].

6. Conclusions

In this short paper, we have given an overview of the WIDE architecture. The aim has been to show how workflow management, extended transaction support, and active rule support have been combined in an orthogonal fashion that provides ample opportunities for distribution and scalability.

In the future course of the WIDE project, we will test the architecture at the end-user sites in the context of insurance and health care workflow applications. Also, we will further elaborate the transaction and rule model to provide fine-tuned support for diverse application contexts.

Acknowledgments

All members of the WIDE project are acknowledged for their contributions to the architecture described in this paper.

References

[BP95] R. Barga, C. Pu; *A Practical and Modular Method to Implement Extended Transaction Models*; Procs. 21st Int. Conf. on Very Large Data Bases; Zurich, Switzerland, 1995.

[CC96] F. Casati, S. Ceri, B. Pernici, G. Pozzi; *Deriving Active Rules for Workflow Enactment*; Int. Conf. on Database and Expert System Applications; Zürich, Switzerland, 1996.

[CG96] F. Casati, P. Grefen, B. Pernici, G. Pozzi, G. Sánchez; *WIDE: Workflow Model and Architecture*; CTIT Technical Report 96-19; University of Twente, 1996.

[CV96] D. Chan, J. Vonk, G. Sánchez, P. Grefen, P. Apers; *A Conceptual Workflow Specification Language*; CTIT Technical Report 96-47; Submitted for Publication; University of Twente, 1996.

[DH91] U. Dayal, M. Hsu, R. Ladin; *A Transactional Model for Long-Running Activities*; Procs. 17th Int. Conf. on Very Large Databases; Barcelona, Spain, 1991.

[GS87] H. Garcia-Molina, K. Salem; *Sagas*; Procs. 1987 ACM SIGMOD Int. Conf. on Management of Data; USA, 1987.

[GV96] P. Grefen, J. Vonk, E. Boertjes, P. Apers; *Two-Layer Transaction Management for Workflow Management Applications*; In Preparation; University of Twente, 1997.

[Mc96] D. McClanahan; *Oracle Developer's Guide*; Osborne McGraw-Hill; Berkely, USA, 1996.

[OM95] Object Management Group; *The Common Object Request Broker: Architecture and Specification, Version 2.0*; Object Management Group, 1995.

[Sie96] J. Siegel; *CORBA Fundamentals and Programming*; Wiley & Sons; New York, USA, 1996.

[WC96] J. Widom, S. Ceri; *Active Database Systems: Triggers and Rules for Advanced Data Processing*; Morgan Kaufmann; San Mateo, USA, 1996.