EXPERIENCES WITH BUSINESS OBJECT-BASED WORKFLOW SUPPORT

Alexander Schill[1] and Christian Mittasch[2]

[1] Fakultät Informatik, TU Dresden D-01062 Dresden, Germany
[2] Institut für Informatik, TU Bergakademie Freiberg D-09596 Freiberg, Germany

[schill | chris]@ibdr.inf.tu-dresden.de

Abstract: Currently, the OMG standardises business objects as a paradigm for encapsulating components of business functionality. This paper reviews this approach and discusses its basic requirements concerning a scaleable middleware platform with associated services. We show how higher-level workflow services can be provided and implemented using business objects as their basic building blocks. A generic architecture, language and runtime systems for distributed and decentralised workflow execution are presented. Particular emphasis is dedicated to the use of business objects and also to the integration of services such as naming, events and transactions. Especially performance aspects are becoming a decisive factor for the success of such component-based applications. Therefore, configuration management aspects and performance results of our implementation are also discussed.

Keywords: Business objects, workflow management, distributed systems, middleware, CORBA

1 INTRODUCTION AND MOTIVATION

The OMG (Object Management Group) currently standardises business objects as a modelling basis for distributed business applications. A business object (BO) is a coarse-grained object and a software component with a well-defined interface that provides domain-specific, standardised and extensible functionality. It is described in a unique manner using Component Description Language (CDL [12]). Thus, it describes its behaviour very abstract, that is platform-independent. Then, an interoperability layer allows using platform-specific characteristics. This abstraction layer allows mapping BO-descriptions to different distributed platforms, too.

L. Kutvonen et al. (eds.), *Distributed Applications and Interoperable Systems II*

The use of business objects in a distributed environment requires a standardised and stable middleware platform, e.g. CORBA or DCOM. Such platforms offer uniform interface specification facilities, standardised inter-object communication mechanisms, and a set of additional services ranging from naming via security to transactions. If the platform CORBA is used, business objects are implemented with the exploitation of ORB-mechanisms and CORBAservices.

Business object instances are associated with object types that are organised in an inheritance hierarchy. Examples of business objects include bills, invoices, customer records, or resources and workflows. Major benefits of standardised business objects are heterogeneous interoperability at the domain-specific application level and easier reuse of components. But, the lack of process frameworks for BOs causes problems that effect the possible success of such software component architectures. Therefore, the contribution emphasises the impact of runtime aspects to a business object-based architecture in CORBA. It especially emphasises the need for performance dependent architectures for distributed applications.

While business objects present major building blocks for complex applications, even higher-level constructs for modelling and executing complex business processes are desirable for many application domains of business information systems. According to the Workflow Management Coalition's (WfMC) definition [25], a workflow consists of a set of linked activities that collectively realise a business objective within the context of an organisational structure. In other words, different activities are executed within a highly distributed environment. In an initial design or restructuring phase, BPR (business process reengineering) and associated tools can support the adaptation of existing, conventional business processes of a company towards a more systematic and partially automatic processing. Therefore, design interfaces for mapping the results of BPR onto workflow management infrastructures are also of particular importance.

The combination of business objects as major building blocks and workflows as higher-level control structures promises to enable rather flexible and extensible solutions [5, 24]. Developers and users expect two main advantages of business object-based Workflow Management Systems (WfMS): (1) better reuse in other business information systems, and (2) better scalability and adaptability.

In this paper, we first discuss aspects of the ongoing standardisation efforts concerning business objects and the requirements and solutions concerning underlying CORBA-middleware in section 2. Section 3 presents a generic workflow management environment based on CORBA and on business objects. Section 4 addresses specific aspects concerning the required services and presents performance results and experiences. Section 5 concludes with an outlook to future work.

2 CORBA BUSINESS OBJECTS FOR DISTRIBUTED APPLICATIONS

In this section, we introduce the foundations of our approach, that is BOs and an abstract software architecture.

2.1 Business Objects in CORBA

More recently, the notion of business objects [5, 12, 24] has been recognised by standardisation bodies such as the OMG. Business objects encapsulate application-specific functionality and provide standard interfaces. They are typically imbedded into an inheritance hierarchy and can be reused this way. Such objects can flexibly be integrated into a workflow processing environment for performing selected tasks within a workflow. Essential additional characteristics are *Autonomy:* Each business object is responsible for its own instantiation, atomicity, durability and access control. *Composition:* Business objects can consist of hierarchical components; for example a business process may include other business processes or sub-processes. They may be fractale.

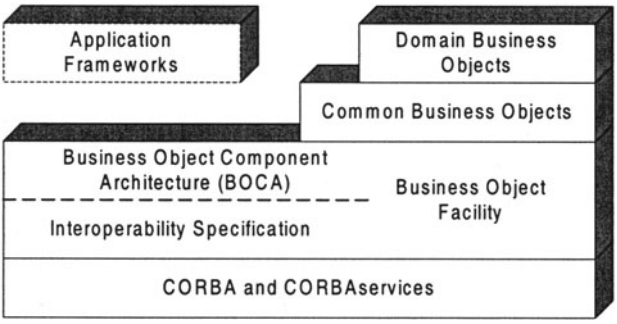

Figure 1. Abstract architecture of OMG Business Objects.

We exploit the notion of business objects for modelling resources (such as objects like orders or invoices [14], servers like order processing or shipping systems, and human interactions with employees) and also processes (workflows, activities).

The OMG also works towards the integration of business objects into so-called frameworks [3, 7]. A framework is a set of co-operating classes representing a reusable design solution for a specific application area. In addition to this definition we would like to stress further characteristics of frameworks: A framework for workflow management should control workflows autonomously. It only should offer interfaces to assign tasks to the framework. Meanwhile, OMG started a standardisation initiative, the workflow resource protocol [15] dealing with this problem, too.

These efforts resulted in the conceptual architecture shown in figure 1. On top of CORBA and CORBA services, a business object framework has been discussed by the OMG in the context of CORBA facilities. Actually it is called Business Object Component Architecture (BOCA [12]). Specific application frameworks for application areas such as finance, transport, or medicine are possibly being derived. One specific example is a workflow framework [9] that deals with repository and execution tasks of workflows. The CORBA workflow facility does actually not deal

with runtime aspects of co-operating objects. It is currently restricted to the WfMC's point of view (workflow clients API and audit data) [11, 17].

Common business objects [14] are base types of business behaviour, such as address, partner, currency, task. They can be specialised and supplemented by domain business objects for specific tasks, for example for transportation or financial issues. An interoperability layer [13] allows to use transparently CORBA and CORBA services. All standards are still under construction. They do not consider CORBA Components [16] at the moment. But, it would be very promising because it will be compatible to Enterprise Java Beans.

We consider these concepts a starting point for the development of our generic workflow environment. The major challenge, however, is to detail these rather abstract concepts significantly in order to provide an implementable systems solution.

Other software component technologies are of particular interest for office automation tasks. Today DCOM and Java Beans are promising component technologies, too. They are not discussed in this paper. DCOM is one of the leading technologies for PC-based components. Java Beans is the major Web-oriented technology for components. Different ways of co-operation are coming up for software components:

- mapping of abstractly modelled BOs to different platforms,
- integration of Java Beans in CORBA-middleware, and
- cross-connected software for DCOM and CORBA components.

2.2 Software Architecture and Solution Frame

First of all, an information system that controls business processes uses almost the same resources as other business applications. Therefore, the traditional way is to use integrated, homogeneous solutions. Business objects allow to develop more flexible and extendible solutions. A 3-tier abstract system structure is required. Data is handled as a lower level concern that is used by all kinds of business objects via appropriate CORBA services (externalisation service and persistent object service) or ORB's POA-mechanism (Portable Object Adapter). This avoids problems with multiple copies of business data.

The medium level deals with all business objects: the organisational structure, all processes (workflows, activities) and all resources (employees, servers, and data-objects). Distribution and integration of the middleware platform (CORBA) are realised here. WfMS are inherently distributed. Distribution is an implicit concern of business objects in the 3-tier architecture. That means they are responsible for hiding the technological impact of their distribution. On the other hand, the mapping of abstract descriptions of business objects to middleware platforms is a main concern of the development of WfMS.

The top level contains applications, for example a WfMS with their front-ends. They are composed of visualisation components. All office applications have to use business objects wrapped by CORBA-interfaces for their special purposes. They use specialised views of the business objects. A WfMS has to integrate state characteristics of an enterprise (organisational structure, i.e. working groups,

departments and their resources). Its main task is to model the pool of workflow types and activities.

3 BPAFRAME, A BUSINESS OBJECT-WFMS

In this section, we describe our prototype implementation of a business object-based WfMS, BPAFrame (Framework for Business Process Automation).

3.1 System Architecture and Workflow Specification

Considering the discussion in section 2, we designed a CORBA- and BO-based environment for the generic support of workflow management systems. It is called BPAFrame and has evolved from subsequent predecessors [7]. The system implementation is based on Windows NT 4.0 with Microsoft Visual C++ and the CORBA implementation Orbix (Version 2.3) of IONA [4].

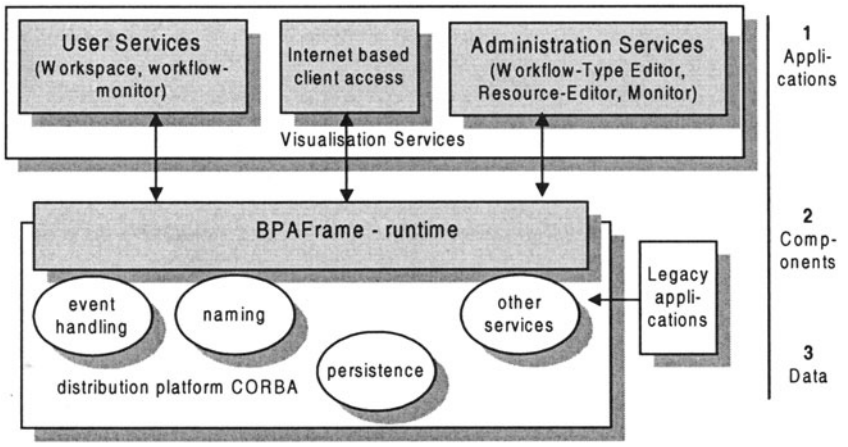

Figure 2. System architecture.

An implementation-oriented view of the system architecture is given in figure 2. The main constituents are a visualisation services framework (administration services, user services, and Internet-client), and the CORBA-based distributed runtime environment. All administration and user tools inherit from our visualisation services framework. Within the user services, a so-called workspace realises the front-end application for authenticated users of BPAFrame. It contains a task list for users involved in the processing of workflows, the list of started workflows and the list of workflow types that may be started by the user. A monitor component is also being developed that enables the inspection of active workflows and of resources as well. User services contain so-called interaction servers, that negotiate data to local applications (for instance a desktop publishing system). A front-end written in Java

exists in a webpage, that allows to start and monitor workflows. Our prototype does not contain special design tools at the moment. Organisational structure, resources and workflow types must be implemented with simple text editors.

Workflows are always associated with predefined workflow types. Their control structure is specified by an execution graph of sequential, parallel or conditional controlled activities. Workflow instances are able to switch between different states with different behaviour during their life cycle (working, suspended, successfully terminated, mobile-working, or aborted). Workflow instances contain an interpreter object that is able to interpret the textual description of the execution graph at runtime. The interpreter accesses adequate resources (modelled by BOs) for activity control. Resources can also be reserved in advance. That allows to guarantee the processing quality of workflows. The runtime environment makes use of ORB communication, naming (Orbix Naming), and own implementations of CORBAservices: event handling, persistence and several other services. Encapsulated legacy applications can also be accessed based on object wrapping. Currently, BPAFrame works in a single administrative domain. Co-operation of domains takes place on the level of the workflow-resource protocol, i.e. by invoking sub-workflows and activities of other domains.

```
// Extract of a workflow example Online-Shopping: Workflow initiation
and 1st activity

extern Any shoppingorder; // input data for the workflow

// declaration of required interpreter functions,
extern getObject(String name, Object obj); // resolve name
extern suspend(); // possibility to suspend the workflow
...
Object CustomerCheck,AbortBehavior, DetailedCustomerCheck, Stock,
SupplierOfFunds,
    Controlling, Shipping;
String rc; // returncode

void main() {
initGlobalObjects();
 try {
 CustomerCheck.checkOrder(in shoppingorder, out rc);  }
 catch(...) {
  printErrorMsg(„The customer check failed...");
  return;
 }
 if(rc == „not clear") { // condition complied branch to „decide by
              responsible"
  try {
   getObject(„manager", DetailedCustomerCheck);
   DetailedCustomerCheck.checkOrder(in shoppingorder, out rc); }
  catch(...) {
   printErrorMsg"The customer check by the manager failed...");
   return;
  } // catch
 } // if(rc == „not clear")
```

Figure 3. Extract of a workflow specification example.

Each workflow type specifies its associated execution graph and all associated resources. Within BPAFrame, we developed a textual description language for distributed applications, called AgentScript2; it is not workflow-specific. It allows to describe processing structures, resources and application-dependent parameters, see figure 3. After the definition of auxiliary data types and external operations, the execution graph is specified within the main body. Each activity specification also comprises an exception handling clause. Parallel execution (for example, of packing and invoice processing) is implemented by subworkflows that are managed independently. Execution is synchronised by using a CORBA event channel. Thus, the successful termination of a subworkflow leads to an appropriate event notification at its event channel. This allows the interpreter of the superworkflow to synchronise the processing. In addition to the constructs shown in the example, repetitive task execution is also possible.

A workflow type description is checked for syntactic and semantic correctness by a parser as part of our architecture. It transforms the definition into an internal representation and passes this to the interpreter component. Standard compiler tools (Lex and Yacc) were used for the parser.

3.2 Class Structure and Runtime Support

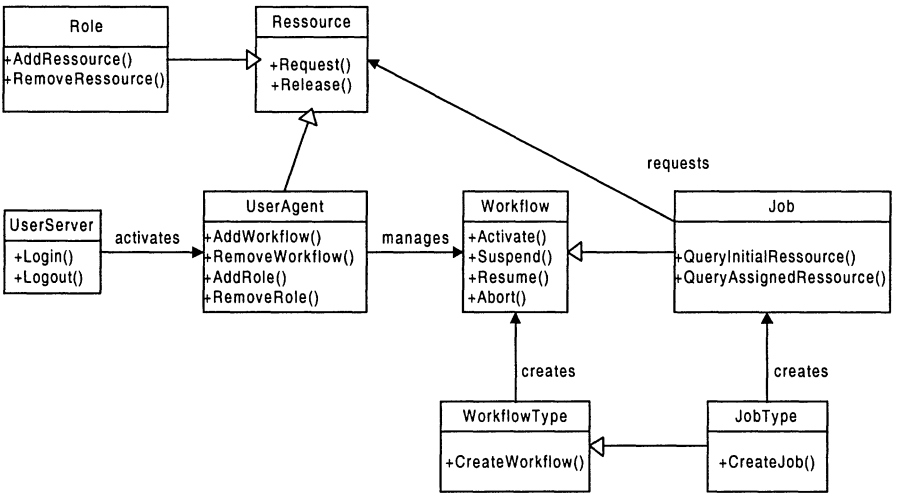

Figure 4. Class structure of the workflow control framework.

Basic components of the runtime-system are workflows and resources. Business process models are described in the workflowtype class. Its instances are results of the application of workflow development tools. During runtime they are only

responsible for the instantiation process of corresponding workflow objects. Workflow objects contain all necessary functions for the processing of workflows. An interpreter controls the processing of the workflow graph and the invocation of resources. Figure 4 shows the class structure of the workflow control framework.

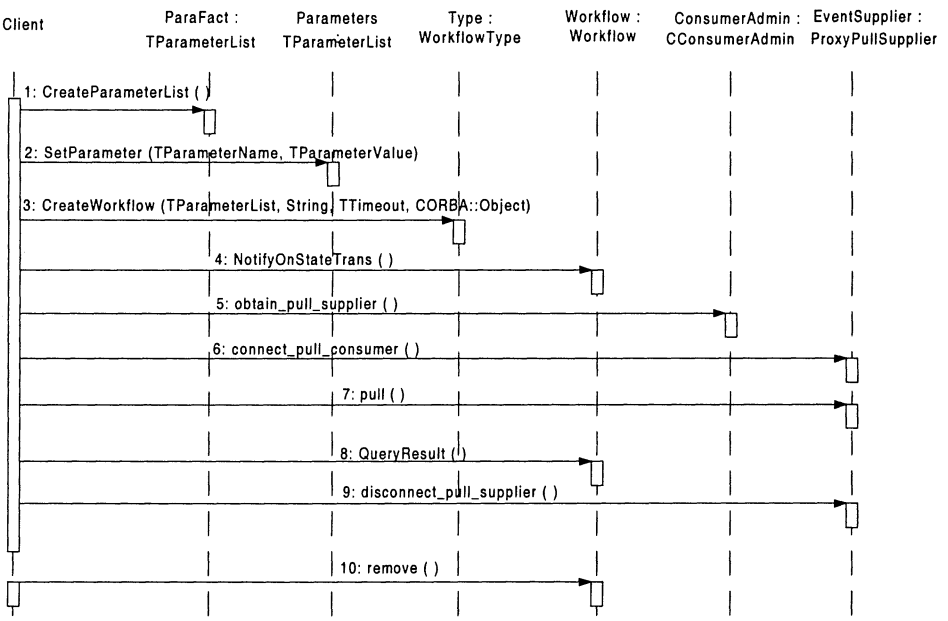

Figure 5. Use of a workflow object from the point of view of a client.

A workflow type factory enables the creation of individual workflow types via tools of the user services. Each workflow type is represented by a business object and offers methods for creating dedicated instances and for querying instantiated workflows. The workflow interface allows its clients to change the workflow's state and to query its parameters. This way, workflows can be inspected interactively via different tools. Sub-workflows are running autonomously. Workflow factories and workflow-type factories are responsible for the distributed life-cycle management of their objects. They are key components in order to control their distribution and to manage system configuration with performance aspects and reliability requirements.

Figure 5 illustrates the actual use of a workflow object by a client, for example by the initiator. After inspecting the offered workflow types, a specific type is selected, and an input parameter list for a workflow is generated and instantiated. Then, the workflow object is created passing these parameters, and an event channel object is established. A monitor object and a recorder object are two of the event consumers listening to the event channel object of the workflow. Legitimated users (owner of the workflow, responsible agent of the organisation) may also listen to interesting events

of the processing of the workflow. A user may invoke methods to control and influence the processing (change time-outs, abort, etc.). Finally, the workflow object switches into the state „finished", and the event channel is released. Additionally the owner of the workflow is informed about the successful termination and about the location of the collected execution results.

As a basis for workflow execution by business objects, the organisational structure of the business environment has to be specified and represented explicitly, too. This structure is represented in a hierarchical way by a logical tree. All organisational units have URL-based addresses as found in WWW. These addresses are managed by the CORBA naming service. The organisational units contain all business objects of their administrative domain, e.g. associated resources (users, roles, software servers) with their appropriate activities and workflows.

Job objects control resources that are used within activities of the workflow. At the moment all kinds of resources in our WfMS are non-exclusive resources. Roles, departments and non-exclusive users maintain lists with currently associated clients. Mobile working users can only work non-exclusively. They are included in BPAMobile, a specialisation of BPAFrame for some tele-working scenarios.

The communication subsystem of BPAFrame defines interfaces of the internal event communication. The supplier delivers information of all kinds of events (i.e. user accepts job, object is created, etc.). There exist four modes of event channel communication: push or pull supplier and push or pull consumer. Supplier and consumer may be interconnected indirectly via event channel objects in different communication modes, in order to fulfil different communication protocols.

Clients can act as event consumers (push or pull) to monitor workflows, to collect all information of their current workflows, or to collect history information in a workflow data recorder. Additionally, the subsystem uses factories as already discussed for the remote creation of objects (object life cycle). Workflow objects contain additional characteristics (i.e. time-out-value(s), owner, priority) and operations necessary for interactions with active workflows (abort, reduce time-out, etc.). The job control subsystem works like the workflow control, but for single tasks. The main difference is the necessity to reserve resources and to interact with them. Roles only organise the notification of resources.

For all remote interactions, the dynamic invocation interface (DII) of CORBA is used. The invocations can be implemented within the basic platform independent from the actual type of the invoked resources. The specific type information is then accessed at runtime, enabling a type-conforming call structure. The distribution platform is able to manage all distribution aspects. That liberates the workflow control from tasks, like invocation, start of objects (if required), or distribution-error detection. Moreover, the runtime system must not be aware of the persistent state of objects. Orbix activation services [4] together with our own implementation of the CORBA persistence service manage the reload of passive objects, if necessary. Abstractions from security details can be provided in a similar way.

3.3 Performance Aspects

The described runtime environment has been implemented completely under Windows NT with Orbix 2.3 and C++, and its functionality has been validated by concrete applications. All components co-operate successfully. The use of CORBA has significantly facilitated the implementation and has contributed to a relatively rapid completion of the system. Basic measurements of throughput and delay of such CORBA invocations were introduced in [19].

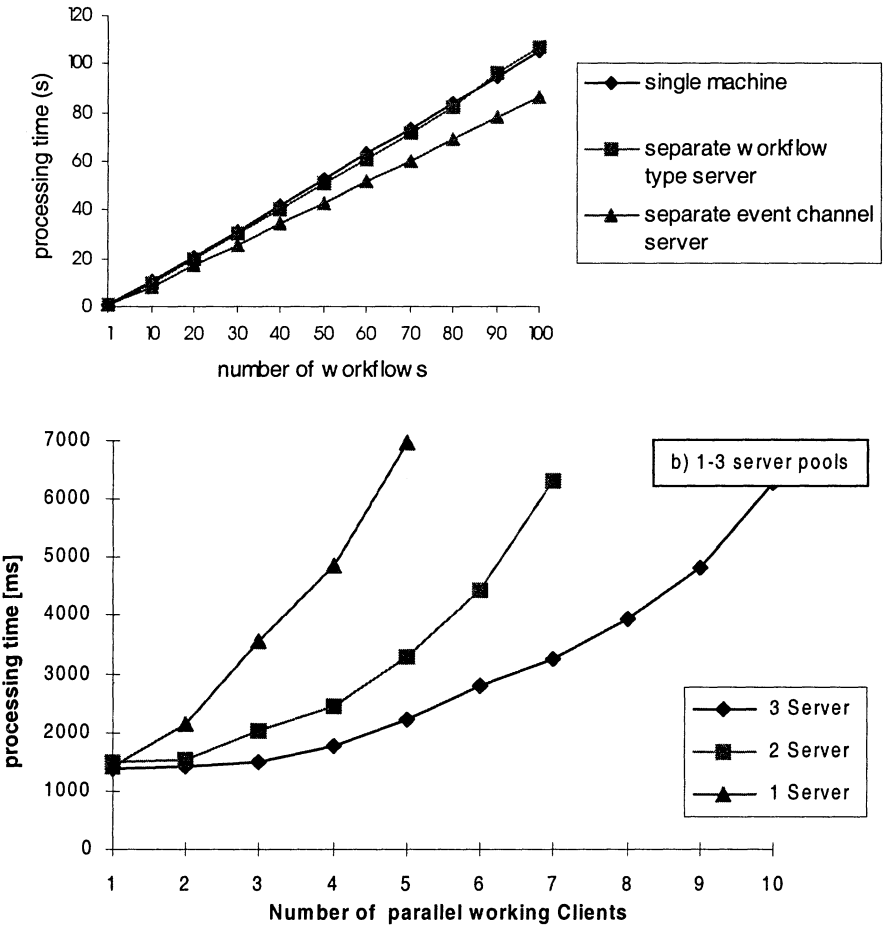

Figure 6. Performance of different server configurations in BPAFrame.

Nevertheless, there are still potential bottlenecks within the current prototype, although these problems are not a design but only an implementation issue. For

example, there is only a single factory for all workflow type objects and just one factory per type for workflow instances. The diagrams of figure 6 give a short overview of the measured performance of different server configurations. The first diagram shows the speed-up of the workflow control framework, if the event channel server runs on a separate machine (empty workflows with constant delay time for activities). One reason to replace it has been the bad performance of most event channel implementations [20].

Figure 6b shows the influence of replication on processing times. If all server processes are available on three machines, the processing overhead decreases more than two times. Additionally a much larger number of workflows can be managed without considerable delay of the workflow processing. The diagram illustrates a main advantage of such decentralised information systems. All PCs can be used to optimise the performance of the entire system without fast servers.

As already mentioned in the previous section, BPAFrame uses the Orbix-naming service in conjunction with environment objects of the workflow control framework. The latter are responsible for the caching of naming information in smaller domains (in organisational units). The naming service takes control of the general names: Start-up servers, the organisational structure, names of related organisations and so on. With other words, naming is responsible for general (and open useable) information, that does not remain under control of domains. Additionally, this way of information sharing reduces the security problem with names of essential parts of an organisation: only freely available names are stored in the naming service.

Other identified improvements of the performance are:

- Some very shortly running subworkflows could be managed more effectively, if such subworkflows could start in threads instead of sub-workflows.
- Role servers could play a more active role for the scalability of resources, if they would actively take part in the workflow-resource protocol. We intend to compare the current protocol with alternative role servers, that are responsible for resource invocation and substitute management.

3.4 Comparison with Other Workflow Approaches

Due to the importance of workflows to most companies, a large number of WfMS with middleware-extensions have been developed in the recent past; examples are FlowMark [6], Exotica [1] and Meteor [21,22] (for more detail see [19]. Moreover, standardisation efforts were made by the WfMC [25, 26, 27, 28] and by the OMG [17] in order to achieve interoperability, portability of workflow applications and integration into existing standards. This is being continued with the above mentioned initiative of OMG for a workflow-resource-protocol [15] and for workflows on the Internet.

Most of the existing approaches, however, are based on a rather centralised view and are implemented on top of a particular database system [1]. This led to a lack of scalability and adaptability in the context of widely distributed environments. In addition, the interfaces of existing systems are mostly proprietary; therefore, it is often difficult to interface between such an overall workflow management system and

arbitrary local resources involved in workflow processing. It is difficult to customise the implementation of workflow management systems for specific application areas.

Most of these systems also fall short in addressing other requirements, especially encapsulation and re-use of resources and workflows, flexible task mapping, and decentralised control. Additionally, they can not quickly adapt to new business tasks of an enterprise. Earlier research approaches addressed these problems – especially distributed systems aspects – to some larger degree; examples are Mentor [30], WASA [29] and Mobile [2]. Some approaches also address specific aspects such as web integration [10, 23], or virtual enterprises [18]. However, they were not using standardised middleware but relied on either proprietary or rather low-level protocols.

4 CONCLUSIONS AND FUTURE WORK

This paper has discussed the paradigm of business objects in open distributed systems and the current standardisation efforts in this area. Based on this background, we presented a CORBA- and business object-based environment for workflow management. We outlined that workflows require generic support with special emphasis on decentralisation, reuse of components, encapsulation of resources, and the use of standards. We also demonstrated that the notion of business objects provides advantages in terms of flexibility and adaptability in the context of workflow management. CORBA and CORBAservices had been proven to present a viable middleware platform for such applications.

Current implementation work focuses on the integration of additional tools for workflow specification and management. Our set of front-ends has been augmented with Internet access facilities based on HTML and CORBA-Java-Applet communication. BPAMobile has been developed, an extension for mobile users in BPAFrame. Future work will address design and implementation of other, possibly application-dependent workflow management schemes. Finally, the integration of distributed multimedia components like video-conferencing is an additional issues. Especially the integration of asynchronous co-operation support by WfMS and synchronous co-operation, for instance in group discussions is of particular interest of our further investigations. The current approach in this direction is, to support such scenarios by ad-hoc workflows. Ad-hoc workflows are already supported by BPAFrame, due to the flexible communication facilities based on DII. They allow changing processing graphs during runtime, that is adaptively.

The mechanism of substitutes has to be enhanced in conjunction with the load of resources. Another future extension deals with the integration of a start-up service based on the corresponding CORBA service to assist configuration management issues. Moreover, selected interactions within a workflow should also be embedded into transactions, making use of the CORBA transaction service. The organisational model also has to be extended in order to reflect the real-world co-operation of organisations and organisational units more appropriately. Here, investigations for virtual enterprises supported by co-operating WfMS on a more abstract level are planned. Each WfMS will itself interact as a business object offering a number of services.

References

[1] ALONSO G., AGRAWAL D., EL ABBADI A., MOHAN C., *Functionalities and Limitations of Current Workflow Management Systems*, 1997, http://www.almaden.ibm.com/cs/exotica/exotica.

[2] BUSSLER C.; JABLONSKI S., Scalability and Extensibility through Modularity: Architecture of the MOBILE Workflow Management System, *Proc. Of the 5th Workshop on Inf. Techn. And Systems*, p.98-107, Amsterdam, Dec. 1995.

[3] COTTER S.; POTEL M., *Inside Taligent Technology. Reading* (Mass.), Addison-Wesley, 1995.

[4] IONA, *Orbix – Distributed Object Technology, Programmer's Guide.* – Dublin, IONA Techn.Ltd.1997.

[5] JACOBSON I., GRISS M., JONSON P., *Software Reuse*, New York: ACM Press, 1997.

[6] LEYMANN F., ROLLER D., Workflow-based applications; *IBM Systems Journal.* Vol 36 (1997), No. 1 – Application Development, 1997.

[7] MITTASCH, IRMSCHER, ZIEGERT, Design and Use of BPAFrame – a Decentralized CORBA-based WfMS, IFIP World Computer Congress, Canberra, Sept. 1996, *Terashima N.; Altman E. (Eds.): Advanced IT Tools*, Chapman & Hall, S. 303-310, 1996.

[8] MOHAN C., Recent Trends in Workflow Management Products, Standards and Research, To appear in *Proc. NATO Advanced Study Institute (ASI) on Workflow Management Systems and Interoperability*, Istanbul, August 1997, Springer Verlag, 1998.

[9] MILLER, J., SHETH, A., KOCHUT, K., WANG, X., CORBA-Based Runtime Architectures for Workflow Management Systems. *Journal of Database Management, Special Issue on Multidatabases*, pp.16-27, vol. 7 (1996), No. 1.

[10] MILLER J., SHETH A., KOCHUT K. AND PALANISWAMI D., The Future of Web-Based Workflows, *Proc. Of the Int. Workshop on Research Directions in Process Technology*, Nancy, France, July 1997.

[11] OMG BUSINESS OBJECT DOMAIN TASK FORCE, *Workflow Management Facility Submission Nortel*, Aug. 97.

[12] OMG, *BODTF Combined Business Object Facility Business Object Component Architecture (BOCA), Proposal, Rev. 1.1*, OMG-Docu. 98/01/07, Framingham: OMG, Jan. 1998.

[13] OMG *BODTF: Combined Business Object Facility Interoperability Spec.*, BODTF-RFP 1 Submission, March 1998.

[14] OMG: *Joint Common Business Objects Revised Submission to BODTF-RFP 1 Submission*, Framingham: OMG, March 1998.

[15] OMG: *Draft RFP Workflow Resource Management Facility (RMF)*, OMG Document: bom/98-12-01, Dec. 98.

[16] OMG: *CORBA Components, Joint Revised Submission*, OMG Document orbos/98-10-18, Nov. 98.

[17] OMG BUSINESS OBJECT DOMAIN TASK FORCE: *Workflow Management Facility Revised Submission jFlow*, April, 98.

[18] OTT M., NASTANSKY L., Modelling Organizational Forms of Virtual Enterprises, *Griese, J.; Sieber, P. (Eds.): VoNet, The Newsletter Institute of Information Systems Department of Information Management University of Berne*, pp. 20-39, Vol. 1, No. 4, September 1, 1997.

[19] SCHILL A., MITTASCH C., A Generic Workflow Environment based on CORBA Business Objects. Middleware'98, The Lake District, England, September 1998, *Davis, N.; Raymond, K.; Seitz, J.: Middleware, IFIP Int. Conf. on Distributed System Platforms and Open Distributed Processing*, pp. 19-34, London: Springer 1998.

[20] SCHMIDT D.C., VINOSKI S., Overcoming Drawbacks With the OMG Events Service, *SIGS*, Vol. 9, No 6. June, 1997.

[21] SHETH A., From Contemporary Workflow Process Automation to Adaptive and Dynamic Work Activity Coordination and Collaboration, *Workshop on Workflows in Scientific and Engineering Applications*, Toulouse, France, September 1997.

[22] SHETH A., The METEOR Workflow Management System and its Use in Prototyping Significant Healthcare Applications, *Proceedings of the Towards An Electronic Patient Record (TEPR '97) Conference*, April-May 1997, Nashville.

[23] SIKKEL K., NEUMANN N., SACHWEH S., Process Support for Cooperative Work on the World Wide Web, *Proceedings 6th EuroMicro Workshop on Parallel and Distributed Processing*, pp. 325-331, Madrid, January 1998, IEEE Computer Society Press.

[24] TAYLOR D., *Business Engineering with Object Technology*, 1995.

[25] WORKFLOW MANAGEMENT COALITION, *The Workflow Reference Model*. TC00-1003, issue 1.1, Nov. 94.

[26] WORKFLOW MANAGEMENT COALITION: Interface 1, Workflow Process Definition Read/Write IF: RfC, *WFMC-WG01-1000*, Feb 1995.

[27] WORKFLOW MANAGEMENT COALITION: WfMC Spec. Terminology and Glossary, *Docu. No WFMC-TC-1011*, Issue 2.0, June 1996.

[28] WORKFLOW MANAGEMENT COALITION, Interface 4 - Interoperability - Abstract Specification, *WFMC-TC-1012*, issue 1.0, Oct. 96.

[29] WESKE, M.; VOSSEN, G.; BAUZER MEDEIROS, C.: Scientific Workflow Management: WASA Architecture and Applications, *Fachbericht Angew. Mathematik und Informatik* 03/96-I, Uni. Münster, 1996.

[30] WODTKE, D.; WEISSENFELS, J.; WEIKUM, G.; KOTZ DITTRICH, A.: The Mentor Project: Steps Towards Enterprise-Wide Workflow Management. - *ICDE'96*, New Orleans 1996.