# Mapping Enterprise Roles to CORBA Objects Using Trader

Alistair Barros, Keith Duddy, Michael Lawley, Zoran Milosevic,
Kerry Raymond, and Andrew Wood

CRC for Enterprise Distributed Systems Technology (DSTC)
University of Queensland, Brisbane, Queensland 4072, Australia
{abarros, dud, lawley, zoran, kerry, woody}@dstc.edu.au

**Abstract.** The ODP Enterprise Language concept of Role provides a useful abstraction for behaviour in a context that is independent of how the behaviour is enacted in a run time system. In CORBA implementations of ODP systems a client object reference variable is analogous to a Role - it is a placeholder for an object whose behaviour is specified by an IDL type. The DSTC UML Profile for Enterprise Distributed Object Computing expresses the Role concept as a UML Action, which is a placeholder for behaviour in UML, and has an attribute representing constraints on the objects that may perform the behaviour (fill the Role). CORBA Object reference variables are assigned to object references using some "bootstrapping mechanism", implemented by a programmer, perhaps using a Trader or Naming Service to locate suitable objects. For the first time in UML, the DSTC EDOC Profile allows designers to specify Roles independent of the class of objects that may perform the Roles. Designers also specify which objects are appropriate for filling which Roles. Furthermore the mapping of this Profile to CORBA technology allows automatic generation of Trader query code to bootstrap the object references of a distributed application according to the high-level design, not the whims of the programmer.

## 1 Overview

As distributed object technologies mature and become more widely deployed, there is an increasing need for rich modelling languages to be able to describe the kinds of enterprise-wide applications that these technologies facilitate. In particular, it is no longer sufficient to provide just an information or computational specification of a system. Rather, for such enterprise systems, there is a recognised need to be able to describe such enterprise-level aspects of the system as the business processes, entities, roles, and events that are involved.

In addition, it is not sufficient to simply be able to describe such aspects of a system. It is important that there be a clear mapping from such enterprise models to distributed object technologies that will be used in the implementation of systems.

C. Linnhoff-Popien and H.-G. Hegering (Eds.): USM 2000, LNCS 1890, pp. 54–66, 2000.

Our approach to providing this support is based on the introduction of modelling concepts that represent dynamic, structural and policy aspects of enterprises. The goals we have in producing such a modelling language are to provide: a small but powerful set of enterprise modelling concepts; an expressive graphical notation; and a basis for automatic generation of component-based enterprise systems.

This paper focuses on how some of our enterprise modelling concepts can be used to specify aspects of a system implementation involving CORBA and CORBA services. In particular we discuss how the CORBA Trader can be used to bind implementations of business roles and business entities.

This paper begins by introducing the Object Management Group (OMG) activity to standardise a UML Profile for Enterprise Distributed Object Computing (EDOC)[5], and DSTC's submission to this process. Section 2 describes our notions of business process, business roles and business events. It outlines what is seen as the key relationship between the use of business processes and business roles in describing aspects of an enterprise system. Section 3 then provides details of the Role and Entity Model within the DSTC's UML Profile for EDOC submission. Section 4 explores the mapping from these design-level models into CORBA implementations using Trader. Section 6 gives an example of a role-based specification of an enterprise system and how that system would be implemented making use of the OMG Trader. Finally in section 7 we discuss the benefits of this novel approach to raising the bootstrapping of distributed applications into the design of applications.

## 2 Key EDOC Modelling Concepts

This section introduces the key models we use to describe various aspects of Enterprise Distributed Object Computing (EDOC) systems. These models provide direct support for describing business processes, business roles, business entities, and business events. As such, they are well suited for forming the basis of extensions to the UML so as to meet the EDOC requirements specified in the OMG UML Profile for EDOC RFP [5].

In order to provide a set of self-contained concepts suitable for practical enterprise modelling, we have integrated ideas from areas such as workflow systems, requirements engineering, and the ODP Enterprise Language Standard [1]. Our submission to the OMG and the work described in [**?**] focuses heavily on the expression of the modelling constructs in UML. This paper is a companion to [**?**] as it focuses on how Enterprise models can be mapped to technologies to support implementations of systems.

While this section introduces and positions our definitions of business processes and business events, the main focus of this section, and indeed this paper, is the specification of business roles and business entities and the relationship between them.

### 2.1 Process Modelling for Enterprises

In our approach, a business process is represented as a dependency graph of business tasks linked in a specific way to achieve some particular objective. A business process can be control-driven or data-driven, or both, and our model provides a rich semantics for expressions of these task dependencies. Our model also supports the composition of business tasks in a way that is suitable for implementation as off-the-shelf components. We also make provision for an association of business tasks with business roles to execute them.

Although our business process model uses concepts found in many workflow systems, nonetheless we view workflow as an IT solution to automating and managing business processes, mostly focusing on the execution semantics. Instead, in our approach, we have attempted to create a succinct business process model that encompasses different workflow execution semantics. We also consider business processes in the context of other business determinants, such as business roles, business entities and business events resulting in an emphasis on business semantics over computational semantics. Our submission to the OMG [6] describes a number of ways of implementing these business process concepts using CORBA interfaces, only one of which includes the OMG's Workflow Management Facility specification [7].

### 2.2 Business Processes and Business Roles Are Dual Concepts

We believe that business process modelling is only one (though frequent) approach to modelling some aspect of a business. There are other possible ways of modelling business systems. In particular, we argue that business role modelling represents an alternative or possibly complementary way of modelling the enterprise. We provide a separation of process-based and role-based modelling concepts as a way of offering different modelling choices. We also separate the notion of business role and business entity, as this separation provides a powerful mechanism for distinguishing between required behaviour and the business entities that can satisfy this behaviour.

### 2.3 Business Roles and Their Support

We believe that Business Roles should be described as fragments of behaviour of the enterprise - those that can then be fulfilled by specific business entities. The separation of the concepts of business entities and business roles enables the specification of the enterprise in terms of behaviour and not in terms of business entities. This modelling approach provides flexibility in assigning business entities to business roles; one business entity can fill more than one role and one role can be filled by different entities, as long as the behaviour of such an entity is compatible with the behaviour of that business role. This allows flexibility in changing the assignment of business entities to business roles as new policy or resource requirements may demand. This is possible because of the way we partition the behaviour of business roles onto business entities.

Such treatment of business roles also provides a basis for flexible assignment of the performers of actions in a dependency graph of business tasks forming a business process. In fact, a business role can be regarded as a collection of actions that are involved in performing one or more business task. The grouping of these actions corresponds to the definition of business roles. This business task versus business roles separation gives an additional power of expression to the business roles versus business entities separation already described. In this respect, our notion of Role is similar to the OORAM concept of Role [13].

### 2.4 Business Events Are Related to Business Processes and Business Roles

In both, process-based and role-based approaches, it is important to expose business events of significance to the enterprise. These events are associated with the modelling elements that can be their sources or sinks and our approach allows for flexible mapping of business event parameters onto the business process elements as well as business roles.

Although we find the use of business events in the description of an enterprise system to be of great interest, they are not of major significance to the intent of this particular paper and little more about them is discussed.

### 2.5 Example

A very brief illustration of the kind of enterprise model that we are describing here is shown in Figure 1. This is a fragment of a model describing the business processes, business roles and business events in a system for managing the technical support for an enterprises that releases software. The fragment shown describes how requests for support are received and serviced.

Figure 1 shows the business process for receiving and processing requests for support realised as a compound task labelled Receive/Process Support Request. This compound task is composed of three simple tasks - Process Support Request, Service Request, and Recursive Invocation.

The Process Support Request task can begin when the enclosing compound task has started and when it has received an event: sup_req of type support_event. This event delivers as payload the data_input that this task requires to begin, indicated by the open circle in the figure. When this task has completed, it enables the Service Request task, and the Recursive Invocation task to proceed in parallel.

The Service Request task has an associated business role - Software Support which will be the role performing the task. The rest of this paper is dedicated to business roles and their implementation and further examples will illustrate our modelling of roles in more detail.

The Recursive Invocation task invokes a new instance of this task at run-time allowing another sup_req event to be received and processed. This kind of recursion is how iteration is modelled in our system.

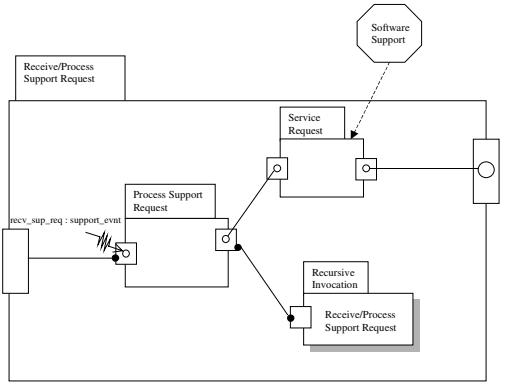


**Fig. 1.** Example fragment of an Enterprise Model

Much detail in this example has been left undescribed due space limitations. However more information and illustrative examples on our approach can be found in [6], [10], and [11].

## 3 Business Roles and Business Entities

In this section we describe how business entities relate to other entities in the business entity model, and to constructs in the business process model. The business entity model is concerned with the descriptions of the behaviour of roles that will, as a collection, describe the behaviour of the enterprise system.

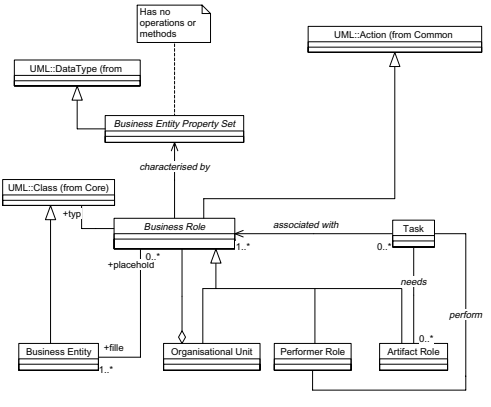


**Fig. 2.** Our Business Entity Model in UML

Central to our Business Entity Model are the abstraction of Business Roles, as illustrated in Figure 2. A Business Role represents a characterisation of some part of the behaviour of the system being described. Performer and Artifact Roles are specialisations of Business Roles. Performer Roles describe active behaviour while Artifact Roles characterise those things that are needed for the actions of Performer Roles (i.e. Artifact Roles do not initiate behaviour). A Business Entity can be said to fill a Business Role if it is capable of enacting the behaviour described by the role being filled. Organisational Units are a composition of Business Roles, enabling the collective behaviour of a set of roles to be another (larger) role. This gives us the ability to describe behaviours at different levels of abstraction.

### 3.1 Business Role

Business Role defines a placeholder for behaviour in a context. This context is an organisational unit (established with some objective in mind) and the behaviour of the role becomes part of the behaviour of the organisational unit as a whole. A business role is defined by its behaviour, its structure and a context in which it exists. For example, a Programme Committee Chair is a role in the context of a Programme Committee (an Organisational Unit).

Business Role inherits from the UML concept of Action, which enables a generic description of behaviour. The behaviour can be expressed using different languages, varying from a program code to English statements. Each Business Role is associated with a UML Class to provide it with a structural description. Finally, each Business Role is defined within the context of an Organisational Unit, which is itself a specialisation of a Business Role that is composed of other Business Roles. Thus Organisational Units (as Business Roles) can be composed into larger Organisational Units, and so on, until the enterprise has been modelled.

Business Role has two subtypes: Performer Role and Artifact Role. Performer Role describes behaviour for carrying out tasks in the enterprise - those that will be assigned to the Business Entities fulfilling the Performer Role. These entities will be responsible for the execution of some aspects of the tasks specified in the business process model described in section 2.1. Artifact Roles have behaviour, however the behaviour described is in some sense passive in that Artifact Roles do not initiate the execution of any action. Artifact Roles are used to represent inanimate things in the system such as resources. For example, a Programme Committee member (Performer Role) performs the review task using a paper (Artifact Role).

In a process-based description, the behaviour of a Business Process is specified in terms of causally-ordered Tasks. There is a correspondence between actions of Tasks and behaviour described by Business Roles. The behaviour of a Task can be composed from (some subset of) the behaviour of one or more Business Roles. Thus a Task is associated with one or more Business Roles (i.e. performed by the Performer Roles and using the Artifact Roles). Each Business Role can be associated with zero or more tasks.

### 3.2 Business Entity

A Business Entity describes an actual object that can carry out (some part of) a Business Role. A Business Role may be filled by one Business Entity or by a collection of them. Similarly, a Business Entity can fill more than one Business Role. For example, Prof. Smith (business entity) can fill the Performer Role of Programme Committee Chair and the paper "O-O for Fun and Profit" can fill the Artifact Role of Submitted Paper.

A Business Entity Property Set is used for specifying non-functional requirements of the behaviour of the Business Role. For a Business Entity to fill a Business Role, the Business Entity object must match the properties specified in the Business Entity Property Set.

Instantiation of a Business Role is achieved by binding to a Business Entity that is able to fulfill the behaviour specified by Business Role. This binding is possible when the Business Entity type is compatible with the Business Role type. However, binding of a Business Entity will commonly be based on more than just type compatibility. Some non-functional characteristics of a Business Role (e.g. QoS) may be specified as a Business Entity Property Set. Hence, Business Entities to be bound to a Business Role can also meet some additional criteria defined by the Business Entity Properties Set. Bindings between roles and objects can be statically defined in the Business Entity model or by Yellow Pages services. In particular, the OMG Trader service [8] can be used to automate the selection of object instances to fill roles, allowing run-time binding. Section 4 details just how Trader can be used for automating this binding.

## 4 Mapping of EDOC Roles to CORBA and Services

While our work describing how it is possible to model Business Entities and Business Roles is useful, this work is of only limited value if there is no clear mapping between the model and some concrete implementation of a system implementing the model. The modelling constructs presented are implementation independent and so can be mapped to a range of suitable technologies including Microsoft COM/DCOM, Java RMI and EJB, and OMG technologies. This paper presents a possible mapping of Roles and Entities to the OMG's CORBA and CORBA Services.

A Business Role is mapped in CORBA as a set of object reference variables in use in some context. This is a novel modelling concept in the Object Management Architecture (OMA) [2], as specifications of clients of CORBA objects, and the binding process by which client code comes to refer to the "right" objects, has been impossible until now.

In order to model a Role we must choose functional characteristics of Objects that will implement the behaviour we are modelling. This is done using a UML Class which is associated with the Role using its "type" metaassociation. The non-functional characteristics of the Role can be specified using a Business Entity Property Set, which is associated with the Role using its "characterised_by" metaassociation. The Business Entity Property Set is also a UML

Class, but one that contains only Attributes, which have names and data types, and may even have specific values given. The Role draws these functional and non-functional descriptions together using its metaattribute called "target" of type ObjectSetExpression, which it inherits from UML Action. This target specification describes (using any appropriate concrete syntax) which Objects it is appropriate for the Role to bind to at runtime.

At the highest level of abstraction, the target expression will probably be a natural language statement of requirements. As the model is refined, technology choices will be made and the target expression can be refined so as to use appropriate concrete syntax.

The type of a Role is mapped from UML Class to IDL Interface.

The mapping for filling a Role is as follows.

As we refine the model by choosing appropriate CORBA Services to implement the binding between object reference variables and CORBA Objects, the "target" expression in the Role's inherited UML Action may provide

- a key for use with a factory/finder (type manager) in order to locate or create an appropriate object.
- an Interoperable Naming iiopname or iioploc URL which nominates a specific object,
- a Naming Context or hierarchy of Contexts which contain appropriate objects.
- a Trader Service Request containing Service Type and Constraint expression which can be used to match appropriate objects through the Trader service.

The mapping for Business Entity Property set will depend on the technology choice made above.

A Business Entity will of course be a CORBA Object instance.

## 5 Mapping Role Binding to Trader

The CORBA Trader allows objects to advertise themselves by submitting a Service Offer in a category of service, called a Service Type. Clients of objects then make Service Requests of the Trader which specify a Service Type and a Constraint expression, which result in a set of appropriate Service Offers being returned to the Client.

The Service Type specifies the CORBA interface type of the objects being advertised, and gives a set of property names and types which will be given values in Service Offers. The Constraint given in a Service Request is a boolean expression over the properties of the Service Type, which the Trader uses to match appropriate Service Offers.

Figure 3 shows a Role with its associated Business Entity Property Set and Type, querying a Trader for matching Service Offers (SO's).

When mapping an abstract Role specification to a Role that will use Trader to bind its entities, the modeller will first design or choose a Service Type. The Business Entity Property Set will then be refined to include property types and

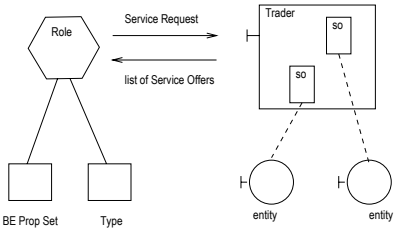


**Fig. 3.** Binding Roles to Entities Using Trader

names that: match the specific properties available in the Service Type; are an element type of a sequence-typed property in the Service Type; and are a structured type representing a high/low range of acceptable values for a property in the Service Type

Naming conventions are established to allow the Business Entity Property Set properties to be related to the Service Type properties.

Finally the Role's natural language target expression must be refined to nominate a Service Type and to formulate a Trader constraint expression that uses the names of the Business Entity Property Set properties as variables to be instantiated when a Service Request is made.

This refinement process is demonstrated in Section 6.

## 6 Example

### 6.1 Scenario

Fnord is a software development organisation that produces a number of software products that are used by clients in many countries around the world. Fnord provides a 24 hour, worldwide support service for clients using Fnord's software products. As part of Fnord's commitment to its clients, Fnord promises to attend to service requests within 24 hours of them being filed. Service requests are made through a web-based online system. For each request, data about the software package concerned, the nature of the problem, the time the request was lodged, as well as the the preferred language of correspondence of the requester is recorded by the service request system.

Based on the data in the service request, the system determines the service representative who is most suitable to handle the request. Because Fnord is a global organisation, this process involves choosing someone who is working in a location in an appropriate time zone, who is able to communicate in the language of choice, and who has the time and expertise to handle the service request.

Figure 4 presents a conceptual overview of the way that Software Support Requests from Clients are assigned by the Software Support Service to Service Representatives. Service representatives may service more than one client at a time, and, as with Client 4, it is possible that Service Representatives cannot be

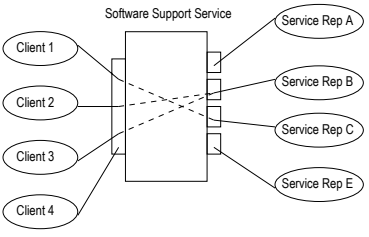


**Fig. 4.** Relating Clients to Service Representatives

assigned to Clients due to some lack of necessary properties such as expertise or availability.

### 6.2 Enterprise Model

Figure 5 is a UML diagram which shows the definition of a Role called `Software Support`. Notationally, we have chosen to represent the Performer Role stereotype with hexagonal boxes. The overlapping box with the two attributes: `lang` and `pkgs` indicates that this Role is a Template Role. This templating is explained in more detail below.

Associated with the `SoftwareSupport` Role are the class `SoftwareSupport Req` which is in the `type` association with the Role. `SoftwareSupportReq` is a class whose operations and attributes are used to specify the functional behaviour for the Role.

The Business Entity Property Set is populated by values from the web-based online `Service Support Request` system. In figure 5 the Role is templated by the required fields from the Business Entity Property Set, namely `<lang>` and `<pkgs>`.

The `SoftwareSupport` Role contains an expression of the required behaviour that the objects filling this Role must exhibit. This behaviour is expressed in terms of a Trader Service Request. The Trader Service Request is composed from the Role target expression parameterised by properties from the Business Entity Property Set.

The Type `SoftwareSupportReq` defines the computational interface that objects filling this role must support. This type information will be included in the Trader Service Request so that only the Service Offers of Business Entity objects with appropriate functionality will be returned as candidates for instantiating the Role. The `SoftwareSupportReq` interface inherits the Traders DynamicPropEval interface so that it may provide dynamic service offer property values to the Trader. In this case, to indicate the time remaining at work for a given support person.

Figure 6 depicts the refinement of the Role. On choosing to use CORBA and Trader, the first stage shows the refinement of the natural language expression of behaviour into the parameterised Trader Service Request. The final

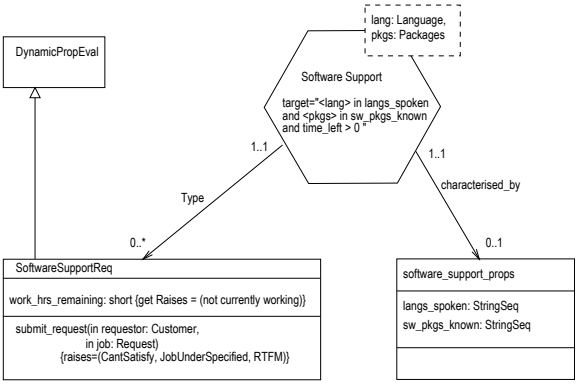


**Fig. 5.** Role, Type and Property Definitions

stage shows the binding that will occur at runtime of the `SoftwareSupport` Role Template with concrete parameters from a Property Set to produce the concrete `SoftwareSupport` Role. At this stage, the Trader Service Request can be completed.

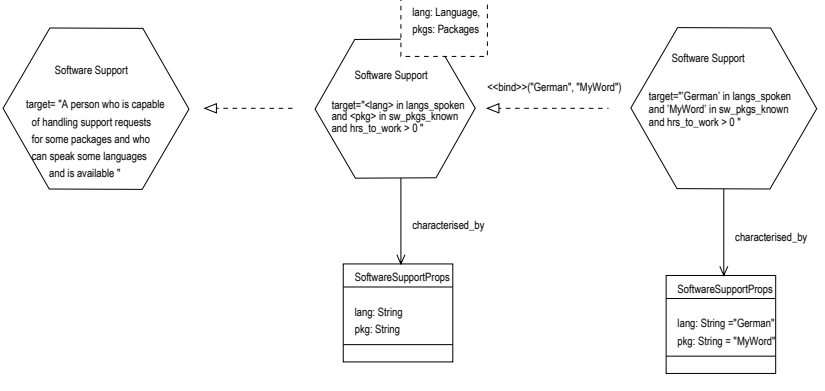


**Fig. 6.** Parameterising a Template Role

Figure 7 depicts a trader with a number of service offers. Corresponding to each service offer is the Business Entity object publishing the service offer. So, the objects `Woody`, `Zoran`, `Kerry` and `Keith` are all objects advertising services in this Trader. In this figure, all these service offers are of the same service type.

The Service Offer for the object Keith has been blown-up to expose the some of the details of the Service Offer. The service offer includes details of the attribute/values of the properties of the object such as `langs_spoken =`

`[‘‘English’’, ‘‘German’’, ‘‘Japanese’’ ]`. It also includes a reference to the object to query to resolve the dynamic property: `time_left` which in this case is the object offering the service.

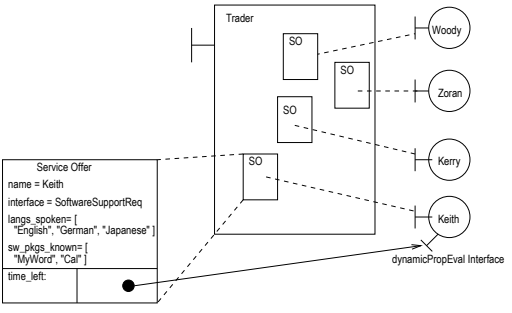


**Fig. 7.** Trader Service Offers

Given the specification of a Software Support Role from figure 6, and the Trader populated with the service offers of objects representing software support personnel who are able to provide support for a range of software products in a range of languages and who are currently working, it is possible to use Trader find appropriate staff to satisfy software support requests in a timely fashion. In this example, the object `Keith` can fill the Software Support Role. That is, given an Enterprise-level description of a Role in the system, it is possible to find objects to fill these roles at runtime using Trader.

## 7 Conclusion

The DSTC UML Profile for Enterprise Distributed Object Computing introduces small but powerful set of enterprise modelling concepts that represent a suitable basis for the automatic generation of component-based infrastructure to support enterprise systems. In this paper we have introduced our notions of Business Process, Role, Entity and Event. The paper has described in some detail our notion of Business Roles and their relationship with Business Entites.

In our model, we expresses the Role concept as a UML Action, which is a placeholder for a behaviour specification in UML. We give this role a computational type by associating it with a class, and non-functional characteristics through an association with a property set.

The mapping to an appropriate binding technology is achieved by refining the behavioural specification into an appropriate concrete syntax. In this paper we show how this is done for CORBA Trader.

The mapping of the DSTC EDOC Profile to CORBA technology allows automatic generation of Trader query code to bootstrap the assignment of object references to objects in a distributed application according to the high-level design, not the whims of the programmer.

## 8 Acknowledgements

Our views on enterprise modelling have been influenced by involvement in the standardisation of the ODP Enterprise Language within ISO [1]. Our Business Process Model has a basis in work [12] done at the Department of Computer Science, University of Newcastle upon Tyne, UK, sponsored in part by Nortel Corporation.

The work reported in this paper has been funded in part by the Co-operative Research Centre for Enterprise Distributed Systems Technology (DSTC) through the Federal Government's AusIndustry CRC Programme (Department of Industry, Science & Resources)

## References

1. ISO, Open Distributed Processing - Enterprise Language, ISO/IEC JTC1/SC7/SC17 N0080, July 1999.
2. Object Management Group, "The Object Management Architecture (OMA)" OMG http://www.omg.org/library/oma1.html,
3. Object Management Group, "Unified Modelling Language v1.3", OMG ad/99-06-08, June 1999.
4. Object Management Group, CORBA Components - Volume 1, OMG orbos/99-07-01, August 1999.
5. Object Management Group, "Request for Proposal: UML Profile for Enterprise Distributed Object Computing" OMG ad/99-03-10, March 1999.
6. DSTC, "UML Profile for Enterprise Distributed Object Computing", OMG ad/99-10-07, October 1999.
7. Object Management Group, Workflow Management Facility, OMG bom/98-06-07, July 1998.
8. Object Management Group, Trading Object Service, OMG formal/97-12-23, 1997.
9. Object Management Group, "Request for Proposal: Action Semantics for UML" OMG ad/98-11-01 Nov 1998
10. A. Barros, K. Duddy, M. Lawley, Z. Milosevic, K. Raymond, A. Woody, "Processes, Roles, and Events: UML Concepts for Enterprise Architecture", submitted to the Third International Conference on the Unified Modeling Language (UML2000), October, 2000.
11. S. Abraham, K. Duddy, M. Lawley, Z. Milosevic, K. Raymond, A. Woody, "Mapping Enterprise Events to the CORBA Notification Service", submitted to the Fourth Enterprise Distributed Object Computing Conference (EDOC 2000), September, 2000.
12. J.J. Halliday, S.K. Shrivastava, S.M. Wheater, Implementing Support for Work Activity Coordination within a Distributed Workflow System, Proc. 3rd International Enterprise Distributed Object Computing Conference, Sept 1999, pp 116-123.
13. T. Reenskaug, P. Wold, and O. A. Lehne, Working with Objects - The OOram Software Engineering Method, Manning Publications, ISBN 1-884777-10-4, 1996