Extending Enterprise Portal Capabilities With a Workflow Meta-Component Framework

Cláudio R. M. Silva¹ and Manuel de Jesus Mendes²

¹*Technology Center of Rio Grande do Norte Federal University, Brazil
Campinas State University - UNICAMP, Brazil*

crmsilva@dca.fee.unicamp.br

²*Renato Archer Research Center - CenPRA, Brazil
Campinas State University - UNICAMP, Brazil*

mendes@dca.fee.unicamp.br

Abstract: A key function to enterprise portals is the ability to participate in a workflow process. Business processes across enterprises can function in an integrated manner with the help of business integration servers that broker information across multiple loosely-coupled applications. A business integration server maintains workflow process definitions and routes business documents via a messaging infrastructure. In this scenario, the portal works as a workflow client handling messages as they arrive and supporting process managers. This paper describes a portal prototype implementation that is supported by an open service platform and by a framework that extend the portal capabilities with metadata and workflow management.

1. INTRODUCTION

A key function to enterprise portals is the ability to participate in workflow processes. Unfortunately, this is not a trivial and direct interaction. To achieve a certain level of integration in their operation, the business processes need the help of business integration servers that broker information across multiple loosely-coupled applications, maintain workflow process definitions and route business documents or messages via a messaging infrastructure inside/outside intranets/extranets used by organizations and its partners.

A business integration server is defined as a scalable platform for the integration of applications within enterprises that eventually interacts with enterprise portals to enable external access to some internal applications behind the portal.

The original version of this chapter was revised: The copyright line was incorrect. This has been corrected. The Erratum to this chapter is available at DOI: [10.1007/978-0-387-35617-4_48](https://doi.org/10.1007/978-0-387-35617-4_48)

J. L. Monteiro et al. (eds.), *Towards the Knowledge Society*

© IFIP International Federation for Information Processing 2003

In this scenario, a portal, supported by an integration server, functions as a workflow client handling messages as they arrive and interacting, eventually, with a process manager that coordinates interactions from many different sources as: collaboration, mail servers, or other integration servers.

The accurate definition of “Portal” is very difficult, since many different types of portals exist. To some, portals represent communities; to others, they are trading hubs or e-marketplaces; to many, they are integrated desktop environments. As the main objective of portals is to unify data streams, we could say that a portal is a single integrated point of access to information, applications, and people.

Figure 1: Enterprise Information Portal Example

We are especially interested in the Enterprise Information Portals (EIP). They are the intranet and extranet versions of Web portals like Yahoo! or Excite. Where web portals mainly display information with very few dynamic capabilities, EIPs are portals embedding tools for analysing data, launching productivity tools, and conducting e-commerce. They offer one-stop shopping for knowledge workers; the portal is both a gateway to and a destination on the enterprise network that provides transparent, tailored access to distributed digital resources (Wiseth, 1999). In Figure 1, we have an example of an enterprise portal and its external relationships with partners (e.g. resellers and suppliers) and some internet clients. Inside the enterprise, behind the firewall, a business integrator server handles the extranet accesses and a Web Server handles the HTTP access to the

portal. There are also several other servers that contribute to the portal functionality. They are: the information, application, message and database servers.

There is not a complete consensus about all the requirements the next generation of portals should have, but some requirements are common to almost all of the analysis and proposals (Wesker, 2001) (Davies-Carr, 2002). They are: More integration, interactivity and bidirectional portal functionalities; Use of XML as enabler; Increase convergence with applications server market and Advanced personalization/collaboration. Some of these requirements are intimately related with the workflow management systems characteristics and this suggest a certain advantage in applying workflow based solutions to extend portal capabilities.

Workflow is defined as the automation of a business process, in whole or part, during which documents and information are passed from one participant to another for action, according to a set of procedural rules (WfMC, 1999). Workflow systems are used to define, create and manage the execution of flows of work through a business process. Workflow systems use a combination of various software components including workflow engines, relational databases, graphical process mapping tools, document management, middleware and reporting tools, which allow organizations to reengineer and monitor their business processes. Workflow systems are able to interpret business process definitions, interact with workflow participants and, where required, call up specific technology tools and business applications.

The field of workflow management received a lot of attention in the past decade. A number of valuable research papers were published during this period, and several standards for workflow specifications were suggested (Boldt, 2000). In the practical plan, various commercial workflow management systems (WFMS) appeared on the market. The technology and deployment practice have developed and matured so much that it is now re-branded as 'e-process management' and it is beginning to play a key role in managing business processes in e-commerce, content and knowledge management. This potential powerful relationship that workflow technology performs together with portals describes an interesting scenario to extend current portals capabilities and achieve the next generation of the portal evolution process. Of course, workflows are not the only important enabler in this direction and among several other ones, we can identify also W3C XML (Extended Markup Language) and OMG MOF (Meta-Object Facility) as important ones.

This paper describes a portal prototype implementation that is supported by an open service platform and a workflow meta-component framework that extends the portal capabilities with metadata and workflow management. It also describes a possible usage scenario for the workflow framework inside a proto-

type corporate portal and a set of other internal servers. In section 2, the modeling concepts are introduced; In Section 3, the framework developed is described; In Section 4, some implementation details are presented and in Section 5, the pros and cons are discussed.

2. COMPONENT MODELING AS METADATA

2.1 Basic Concepts

Modelling is a key first step in the building of strength enterprise software systems and in this direction the OMG's Unified Modelling Language specification (UML) (Heaton, 2001) performs an important role. UML is a visual language for the development and exchange of well-defined models for software development. Standard UML language components provide much of the functionality that users need to model their applications and the UML mechanisms for extension and specialization allow use in areas in which standard components fall short.

Supplementing the UML, MOF (Boldt, 2002) provides a standard repository for metadata within the CORBA architecture. Metadata are, basically, data about data or as some others prefer to say: it is structured data about data (Iannella, 2002). MOF is defined in terms of a meta-model and set of IDL interfaces. it supports component-based computing from modelling and design through implementation to run time. The common meta-model can support introspection, sharing of components across heterogeneous distributed environments, and through revisions as components' life cycles evolve.

OMG's UML is based on a common meta-model. This meta-model is, basically, a set of diagrams, semantic and syntactic rules that defines the core elements and relationships used in UML. As such, the meta-model derives all other terms from that core set of elements and relationships. Varying application development efforts call for differing diagrams.

The UML meta-model ensures that UML is more systematic and internally consistent than any earlier software modelling technology. In addition to core symbols, the meta-model contains supplementary symbols, called extensions. Extensions are usually represented by graphical adornments added to existing core symbols. Also featured are stereotypes, which are created by those interested in customizing UML. Using stereotypes, an industry group can extend

UML to incorporate some special-purpose notational system. If the group follows the rules, and stereotypes are derived from core UML elements, they can be incorporated into UML diagrams and used by UML tools. An extensive package of stereotypes is referred to as a UML profile.

The OMG UML Profile for EDOC (Enterprise Distributed Object Computing) specification (Heaton, 2002) is an important step in this direction. It provides a modelling framework by defining: A set of profile elements comprising a technology independent profile, an architecture for enterprise collaborations (ECA) allowing the definition of platform independent models, a pattern profile, a set of technology specific models allowing the definition of platform dependent models and a structure for the application of the profile elements in the specification of EDOC systems.

2.2 Workflow Meta-model

This work is influenced by several EDOC concepts, such those related with the process components specification described in the Component Collaboration Architecture. But it is a minimalist approach that tries to apply only elements strongly related with workflow business process.

The main objective of the meta-model is to provide the definition and explicit representation of the entities used to build particular configurations of the framework. This is done in terms of meta-information that describes the structure of such entities, as well as the relationships among them.

The structure of the meta-model is derived from the standard CORBA Interface Repository (Heaton, 1999) and in terms of modularization, the meta-model is divided into five interrelated packages: Interface, Component Structure, Task, Organizational and Document Packages.

The interface package represents the constructs used to define the interfaces of components and other objects in the meta-models. Component structure describes the encapsulation of state and functionality in the meta-model programming model. A component exposes its functionality in terms of one or more well-defined interfaces, which are the only points at which the component can be accessed. See Figure 2 below.

Figure 2: Component Model

Figure 3: Activity (a) and CompoundTask (b) Representation

The document model includes detailed description of documents and forms used in other models and finally, the organization model includes organizational entities and relationships necessary to our case study enterprise organization modelling.

The Task model is the main part of meta definitions. As all the other models, it is represented as a package and describes the business process definitions thru tasks, compound tasks and other elements necessary to connect them and their data according to a graphical representation, initially presented in (Warne, 1998). See Figure 3 and the corresponding model in Figure 4. In order to describe the meta-information necessary to the framework, a formal tool is used to represent the details of all packages in UML Class diagrams, as illustrated to Task package in Figure 4, and then mapped to a formal description language known as MODL (Meta-object Definition Language) (DSTC, 2000).

Figure 4: Task Model UML Class Diagram

In the context of this work, the definition of the meta-model packages inside a MOF repository enables our framework to achieve an important set of capabilities as the possibility to change the metadata definition in run-time.

The MOF standard is intended to support a wide range of usage patterns and applications. To understand the possible usage patterns for the MOF, the specification presents two distinct viewpoints that must be understood. They are: The modelling viewpoint and the Data view point. In the Modelling viewpoint the MOF is used to define an information model for a particular domain of interest. This definition is then used to drive subsequent software design and/or implementation steps for software connected with the information model. In the Data viewpoint, the MOF is used to apply the OMA-based distributed computing paradigm to manage information corresponding to a given information model. In this mode, a CORBA client can obtain the information model descriptions and to use them to support reflection.

The specification also defines a core MOF model that includes a relatively small, though not minimal, set of constructs for object-oriented information modelling. The MOF model can be extended by inheritance and composition to define a richer information model that supports additional constructs. Alternatively, the MOF model can be used as a model for defining information models. This feature allows the designer to define information models that differ from the philosophy of the MOF model. In this context, the MOF Model is referred to

as a meta-meta model because it is being used to define meta-models such as the UML.

3. THE FRAMEWORK PROPOSED

The workflow meta-component framework developed in this work has, initially, been designed with the main objective of aggregating value to accomplish some requirements for the next generation of portals presented in section 1. They are: a) More integration, interactivity and bidirectional portal functionalities; b) Increase convergence with applications server market and c) Advanced personalization/collaboration. The accomplishment of these requirements is a natural consequence of the integration of the workflow and metadata capability inside the service platform we have chosen. This platform was developed to open marketplaces. It has strong profile capabilities that enable personalization and customization in several levels to the users.

To understand this section, it is necessary to present basic concepts related with patterns and frameworks in general. Patterns support reuse of software architecture and design. They capture the static and dynamic structures and collaborations of successful solutions to problems that arise when building applications in a particular domain. Frameworks support reuse of detailed design and code. A framework is an integrated set of components that collaborate to provide a reusable architecture for a family of related applications. Together, design patterns and frameworks help to improve key software quality factors and reusability in general.

The Frameworks are strongly implemented based in design patterns collections of metadata and workflow management. In this way, some well-known design patterns described in (Gamma et al., 2001) were also applied to both frameworks to solve common problems. But besides these common problems, there are some specific problems to solve when one has the task to develop an object-oriented workflow framework. In this context, some authors have identified special patterns named as workflow patterns that have been studied in the latest years with the maturing of workflow management systems implementation and deployment in research or commercial work. (Aalst et al., 2000) and (Manolescu and Johnson, 1998) are good examples in this direction. The documentation for a framework must meet several requirements, but these requirements can all be met by structuring the documentation as a set of patterns.

The framework is a set of components that implements the workflow and metadata capability necessary to the portal, inside the open service platform. The framework has four functional groups of components: Basic Control, Metadata management, Workflow Kernel and Workflow Basic Services. See Figure 5.

Figure 5: Workflow Meta-Component Framework Integration

The Basic Control group is responsible by initialization, termination, repository discovery and update operations necessary to all the other groups inside the workflow meta-component framework.

The Metadata Management group provides functionality that enables any platform component to create, change or delete model meta-objects in a MOF 1.3 repository. It is initially based in the meta-models presented in section 2, but it is flexible enough to be extended to other group of models, since its implementation is strongly based in Dynamic invocation interface. So, it also enables an administrator user to create, change or delete meta-model meta-objects, regenerate the interfaces in run-time and then migrate (when possible) the instances from a meta-model to another. It also interacts directly to a federation of repositories available and is able to manage collections of meta information in all repositories available in the federation. This group of components is our main contribution since we implement several functionalities not available in the MOF facility tools currently available. They are: meta model collection management, repository discovery process and metadata access thru dynamic invocation interface.

The Workflow Kernel is the group responsible by all operations necessary to the enactment of a workflow process definition in real-time. It is a distributed environment, where each component, using their own workflow engine interacts with other workflow engines thru the enactment service available in the kernel. It also implements inside the engines the interface model proposed by OMG and

is strongly based in the patterns described in (Gamma et al., 2001), (Aalst et al., 2000) and (Manolescu and Johnson, 1998); The Workflow basic services supports the workflow components with functionality like adaptability features (callbacks for dynamic changes), SMTP messaging and audit system access. This is a modular and extensible framework, since new groups/services can be integrated later.

4. IMPLEMENTATION

4.1 Overview

This section presents some implementation details of the corporate prototype portal and its relationship with its supporting open service platform. The open service platform is the middleware platform PLATIN (Eckert et al., 2001). It forms a processing environment for an open service market into which distributed applications can be managed, launched, and executed. This environment envisaged by TINA (telecommunications information networking architecture) (Chapman, 1995) comprises heterogeneous and interconnected DPE (distributed process environment) nodes that fulfil mainly two purposes: they hide technical/organizational heterogeneity and they present a target system that can help to ease the application design tasks.

The middleware platform helps to decouple applications and the systems necessary for their support, such that both can be developed and evolve in different time lines. The PLATIN service architecture provides a framework that takes into consideration the requirements of an open service market. It describes how services are developed, offered and provided, and how the mobility of users, output devices and sessions is achieved. The platform has already a framework that supports the service components in several capabilities as: access session management, subscription, accounting, communication environment and others. The developed workflow meta-component framework extends the platform and portal capabilities with workflow and metadata management features. The framework is integrated with the existing PLATIN framework and enables transparent usage to the components that support the portal.

4.2 Meta-Component Repository

Our Repository development process comprises three main steps: formalization of the meta-model to be used; development of a repository manager and development of a set of repository client components inside the workflow framework. The information model used to describe the workflow process definitions and its resources was partially presented in section 2. The second and third steps are implemented with DSTC dMOF tool product. The dMOF is, basically, a set of command-line interface commands that enable the generation of MOF-based repository server code to information models defined in MODL and stored in the MOF format. After these steps, the repository server interacts with its clients throughout IDL interfaces generated by dMOF tool.

When the metadata repository has been implemented, it is time to develop the metadata tools needed by the system. These may include: Metadata input/output tools; Metadata interchange tools or Metadata versioning, comparison and archiving tools. In our case, two set of classes were necessary. One to support a designer component in adding new process definitions to the repository and a set of classes to support the workflow-based components in accessing them in run-time to fetch and execute the workflow processes. The designer component is a simple ad-hoc program without any graphical resources, to guarantee that the definitions would be in the repository.

Considering the automatic generation of the repository manager, to integrate the repository in the platform means basically to put the two set of classes implementations inside the metadata management part of the workflow framework and enable the workflow kernel and other application services to manage metadata previously stored in a database.

4.3 Scenario

A fictitious scenario is implemented to enable the necessary functionality to the prototype portal. The level of web portal functionality is a function of the amount of underlying system integration. We have a simple underlying system. The portal enables to search for products, to build a shop list and to order for the shop list items. The internal workflow components contain most of the standard workflow entities and attributes for modelling a workflow process. The informal description is provided in a form that is often employed by user organization. The fictitious company is an enterprise that sells microcomputers thru internet. Let's suppose its name is: Internet Micro Supplier Company (IMS Co.). The

IMS Co. assembles a complete range of microcomputer equipment. They only sell to IMS resellers and the sales are made by way of Purchase Orders.

The company has grown from a small organization to the point where the responses to their customers are getting longer and longer. They have decided to employ a workflow management system in order to improve their business processes. In the existing environment all orders come thru their portal order forms and they are processed by a component that distributes the orders to the appropriated department component.

The business processes components (as Sales, Finance, Stock, etc) are implemented in java as PLATIN service components (inside the PLATIN Server program) and are accessed , internally, through a client program named Provider Agent. There are several approaches to integrate services realized with EJB technology in PLATIN platform. It is possible to implement pure Java-RMI communication, or one can use CORBA 2.3 IIOP communication. See Figure 6.

Figure 6: Prototype Scenario – Details

From the standpoint of (IKV, 2001), the most elegant solution is an integration using a CORBA 2.3 implementation of the Java service as an adapter to the Platform. As a consequence, then, the communication between the service user interface and the service server occur via CORBA 2.3 IIOP, so that the EJB-based service implementation could be replaced by another technical solution. The scenario component implementation has a mixed form, containing elements of pure Java and elements of a CORBA environment.

When the scenario is setup to support the portal functionality, an administrator uses the configurator component to initialize the workflow framework and the workflow components necessary to the portal operation. This is an internal activity and just available to authorized personnel inside the enterprise. It is important to say that when a workflow component is started, it has to fetch its functionality, non-functional properties and process definition in a MOF metadata repository, in which, the meta-component information had been previously defined in build-time. Inside each component there will be a worklist and a light-weight workflow engine provided by the framework thru one of its callback methods. In Figure 7, the internal activities of the component process definitions are shown. They are connected by “dependences” (dj) between inputs and outputs inside input/output sets. Workflow components have a compound task inside their description, so the activities are also inside a task structure. The finance component is supposed to receive three types of documents (Sales orders, Payments and Invoices). Each one of them is associate with a different input and dependence (d1, d2 and d3).

Figure 7: Process Definition loaded by Finance Component

Inside the workflow basic services there are a set of callbacks designed to implement dynamic changes in run-time. This means that configurable properties and a process definition executed by a component may change in the processes that are already running. This is not a trivial operation, since it depends on the process state and type of change. Sometimes it is not even possible to change a process that is already running, just the new instances to come. Any way, the

framework adds some level of run-time adaptability providing callbacks to the following operations: activity bypass, addition/deletion of activities, addition/deletion of input/output sets, consistency check after changes, check for feasibility for change in running process, functional and non-functional property change and dynamic choice of participant based in non-functional property criteria. See the example in which the administration decide to delete the manager authorization to orders over a specified limit. The framework callback responsible for activities deletions would basically do the following steps: 1. deleting activity from the running instances in which the execution state is prior the activity to be deleted.; 2. replacing dependence 10 and 12 by dependence 16; 3. replacing dependence 9 and 13 by dependence 17.

Its is important to note that there are two types of consistence analysis to be considered before change a workflow process definition: The one that is made over the process metadata and the one that is made over the process instances. Both are important and difficult issues to handle, but the second is harder and is rarely implemented in known solutions. They assume that only new instances to a changing process migrate to a new process design. This work is our first step to build better adaptable workflow-based solutions.

5. RELATED WORK

The work reported in this paper corresponds to the research areas: flexible workflow management systems and metadata management. Several other works have been published in these areas, aggregating value to some specific needed feature. Some examples of related work are: Obligations (Bogia 1995), MOBILE project (Jablonski and Bussler, 1996), OPERA project (Hagen, 1999), MICRO Workflow (Manolescu, 2001), etc. Though, our approach is different in applying full OMG modelling standards and by supply a flexible and adaptative way to manage process definitions metadata. The approach is also different for considering itself as part of an legacy open service platform.

6. CONCLUSIONS

Our work is specially focused on portal requirements that could benefit from workflow and metadata advantages. They are: better integration, interactivity and bidirectional portal functionalities and increased convergence with applications server market. The business process modelling in meta components and the

workflow capabilities aggregated to PLATIN platform contribute strongly to these requirements.

The implementation process has proved to be a complex engineering work and this had an important impact onto the software process deployed. Backend integration is based on PLATIN platform, front-end integration is based on a commonly used user interface which called for careful design.

Most requirements for IMS Prototype portal were fulfilled by integrating standard tools. The productive use of IMS Prototype portal showed that the openness of the architecture is a crucial issue. The flexibility is also a crucial issue. The MOF repository features as run-time meta information management provides the degree of adaptability necessary to flexible WfMS, enabling the redefinition and reconfiguration of meta-components in run-time. Finally, the effort for implementing was lower than initially expected, simply because we were able to benefit from design patterns and specialized tools.

7. ACKNOWLEDGEMENTS

We would like to thank the institute GMD FOKUS, Berlin – Germany, by the opportunity to use their open service platform in this project and by all the support we have received during our work. Other important help we have had came from Distributed Systems Technology Center – DSTC, Brisbane - Australia. Without the use of dMOF tool, we would have had to implement all the MOF specification or to use a non-standard repository. We also thank the Brazilian Research Institutes CAPES and CNPq by the help with scholarship programs and other research expenses.

8. REFERENCES

- Aalst, W.; Hofstede, A., Kiepuszewski, B. , Barros, A. (2000) “*Workflow Patterns*”, URL <http://tmitwww.tm.tue.nl/research/patterns/>
- Boldt, J. (2000) “*Workflow Management Specification*”, ver. 1.2, OMG doc, URL <http://www.omg.org/cgi-bin/doc?formal/00-05-02>.
- Boldt, J. (2002) “*Meta Object Facility Specification*”, ver. 1.4, OMG doc, URL <http://www.omg.org/cgi-bin/doc?formal/02-04-03>.
- Chapman, M. (1995) “*Overall Concepts and Principles of TINA*”, TINA consortium, URL <http://www.tinac.com/specifications/documents/overall.pdf>
- DSTC (2000) “*dMOF tool User's Guide*”, release 1.1, DSTC, Brisbane, Australy, <http://www.dstc.edu.au/Downloads/CORBA/MOF/dMOF1.1.UserGuide.pdf>

- Eckert K.; Körner E, Schoo P. (2001) *"Introduction to Middleware Platform"*, doc. Ver. 1.0, <http://www.enago.de/>
- Gamma E.; Helm R., Johnson R.E., Vlissides J. (2001) *"Design Patterns – Elements of Reusable Object-Oriented Software"*, Addison Wesley, july.
- Hagen, C.J. (1999) *"A Generic Kernel for Reliable Process Support"*. PhD thesis, Swiss Federal Institute of Technology, Zürich, Switzerland, URL: http://www.inf.ethz.ch/departement/IS/iks/publications/files/diss_hagen.pdf
- Heaton, L. (1999) *"CORBA 2.3 - Interface Repository"*, OMG doc, URL <http://www.omg.org/cgi-bin/doc?formal/99-07-14>.
- Heaton, L. (2001) *"Unified Model Language Specification"*, ver. 1.4, OMG doc, URL <http://www.omg.org/cgi-bin/doc?formal/01-09-71/80>.
- Heaton, L. (2002) *"UML Profile for Enterprise Distributed Object Computing"*, URL <http://www.omg.org/cgi-bin/doc?ptc/02-02-05>.
- Iannella, R. (2002) *"Mostly Metadata: A Bit Smarter Technology"*, University of Queensland, URL http://www.dstc.edu.au/RDU/publications/m_vala98/
- IKV (2001) *"Enago, OSP Version 0.9 Introduction"*, version 0.9.02, Draft April, IKV URL <http://www.enago.de/>
- Jablonski, S. and Bussler, C. (1996) *"Workflow Management—Modeling Concepts, Architecture and Implementation"*, International Thomson Computer Preess.
- Manolesco, D. (2001) *"Micro-Workflow: A Workflow Architecture supporting compositional object-oriented software development"*, PhD thesis, University of Illinois, USA.
- Manolescu, D. and Johnson, R. (1998) *"Patterns of Workflow Management Facility"*, <http://citeseer.nj.nec.com/manolescu98patterns.html>
- Warne J. (1998) *"Nortel Revised Submission to the Workflow Management"*, URL <http://www.omg.org/cgi-bin/doc?bom/97-08-04>.
- WfMC. (1999) *"Terminology & Glossary"*, standard number TC 1011, pg 8, URL http://www.wfmc.org/standards/docs/TC-1011_term_glossary_v3.pdf
- Wiseth, K. (1991) *Portal Power*; Oracle Magazine; November, 1999, last visit June 2000 URL: <http://www.oracle.com/oramag/oracle/99-nov/69ep.html>
- Wesker, M. (2001) *"Using a Portal to solve Business Problems"*, KMWorld magazine, july/august, pg.54 and 55.
- Davies-Carr, H. (2002) *"Next Generation Portals"*, BEA Web Logic presentation, URL <http://www.client-information.com/bea/PDF/ukportal.pdf>