# Web Service Based Universal Management of Workflow Resources

**Jinyoung Jang, Yongsun Choi**

Dept. of Systems Management & Engineering, Inje University, Kimhae, 621-749, Korea
pongsor2@hotmail.com, yschoi@inje.ac.kr

**ABSTRACT**

Implementing business process solutions in the way of Web service is being positioned in the center of workflow manag ement. However, there is no robust standard to expose and access workflow resources by Web service interfaces. In this paper, we propose a web service based workflow resource management framework named Universal Resource Manage ment Framework (URMF) with declarations of web service interfaces and interaction protocols among them. We also in troduce a substitutive workflow interface model employing Web services and URMF. Finally, a prototype implementati on model of URMF with J2EE platform is also introduced.

***Keywords***: Web services, workflow, resource management, organizational resource, resource resolution

## 1. INTRODUCTION

Workflow management is getting more attention as a way of fostering efficiency potentials by eliminating transport and wait times between activities and providing a detailed level of control over the assignment of work to process participants ([25], [26]). Workflow implementation inside the boundaries of single organization is steadily increasing, however the use of workflow management for the coordination of inter-enterprise processes is still at a very early stage ([5], [31]). Recently, Web services technology, that carries interactions through the exchange of XML messages, provides an alternative approach to implementing interoperable business process solutions in an efficient and standard way ([8], [19]). Application integration through language-neutral and environment-neutral Web services model yields flexible, loosely coupled, and highly extensible business systems.

Current web service composition languages focus mainly on process control perspective but not much on organizational perspective. They lack seriously in managing the organizational elements, such as organization chart, roles, and worklists for human or non-human agents [1], which play important roles in executing business processes. For the successful implementation of workflows in organizations, it is required to enhance web services in supporting organizational resources management.

In this paper, we present a web service based workflow resource management framework named *Universal Resource Management Framework (URMF)* that declares Web service interfaces and the interaction protocols among them. We also introduce a substitutive workflow interface model employing Web services and URMF. Finally, a prototype implementation model of URMF with J2EE platform is also introduced. This paper is structured as follows. Section 2 introduces the issues of organizational resource management in workflow and the related works. Section 3 describes the advantageous features Web services as a resource management facility. Section 4 describes our Web service based framework for the universal management of workflow resources, especially of organizational resources. An example implementation model of our framework with J2EE platform is described in section 5. Section 6 concludes the paper.

## 2. ORGANIZATIONAL RESOURCE MANAGEMENT IN WORKFLOW

Workflow management technology supports the execution of business processes through the automated coordination of tasks, data, application logic, and workflow participants (resources). Ideally, the workflow enactment service should be neutral to the organization and resource aspects. Building organization specific services independently from other perspectives, like of process control, can support the needs of wide variety of organizations and resources [33].

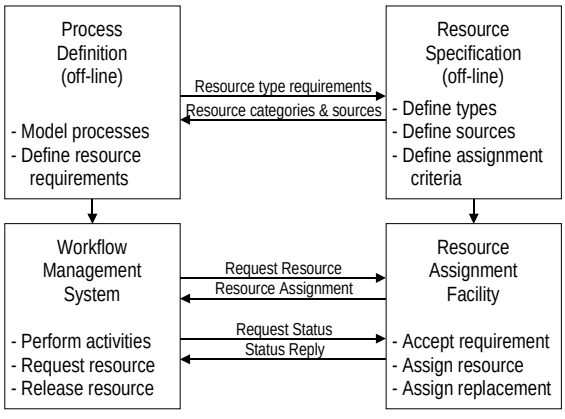


**Fig. 1.** Separation of resource management perspectives in workflow [21]

Fig. 1 illustrates the separation of resource management perspectives in workflow. Division between a process model on one side and a resource model on the other side fosters the separate evolution of both models. It gives robustness to workflow models independent of changes in the organizational structure. Separate

resource model shared by several workflow engines reduces administrative overhead and prevents possible redundancies [33]. Existing workflow management systems have difficulty in distributed worklist management resulted from the architecture of integrated worklists with the enactment service ([21], [22], [30]).

Resource management framework in workflow can be classified into three categories ([33], [34]). Fig. 2 shows the relationship among these categories and related studies. Main focus of each approach can be summarized as follows:

- *Organizational meta model.* Studies under this category suggest the meta models of organizational structure to be used for the mapping of an existing organization. These meta models is moving to be more generic and comprehensive to cover organizational objects such as organizational positions, position types, tasks, or units. Exposing too rich models tend to cause too tightly coupled and highly dependent architecture of resulting systems ([32], [33]).
- *Resource Resolution.* Upon the instantiation of a workflow activity, the workflow enactment service places work items on the work lists of qualified performers [30]. Resource resolution handles on the strategy and the mechanism for work allocation among competing process participants, utilizing role hierarchy, delegation rules, and various organizational constraints like "binding of duties" and "separation of duties". Formal resource resolution languages like Resource Query Language (RQL) [16] are employed to provide a 'view' to access the inner organization model, in finding out eligible resource, without the direct exposure of the entire organizational model.
- *Resource Assignment / Coordination.* This layer provides a mechanism of how to reserve, cancel, or queue workitems into the worklists of selected workflow resources. Services in this layer may be exposed by APIs such as CORBA IDL.
-

| **Resource Assignment / Coordination** |
|---|
| • *Work Activity Coordination System* [14]<br>• *Enterprise Workflow Management Framework* [12]<br>• *Resource Assignment Interface* [21]<br>• *Workflow Resource Management Facility* [11] |
| **Resource Resolution** |
| • *Resource Policy Model* ([6], [7])<br>• *Resource Query Language* [16] |
| **Organizational Meta Model** |
| • *OMM Organization and Role Model* [9]<br>• *Organization and Resource Model (ORM)* ([23], [24])<br>• *Generic Organization Model* ([6], [7])<br>• *A Generic Resource Meta Model* [33] |

**Fig. 2.** Issues and related studies of organizational resource management in workflow

However, these works on organizational resource management in workflow are limited in handling the issues of interoperability and heterogeneity, which can be overcome by the web service technologies. In next section, we will discuss how resource management framework can be modeled and implemented with the web services.

## 3. WEB SERVICES AND RESOURCE MANAGEMENT

Existing workflow resource management approaches are much dependent on the underlying models or technologies, resulting tightly coupled architectures. They have difficulty in allowing the flexible and coordinated interaction patterns, which can be provided by the web service architecture. Although there exist many distributed software platforms that could be useful to implement workflow resource facility, Web services technologies are getting more industry momentum with many advanced features, like language and platform independence, interoperability through wire level standards, and the ability to expose interfaces but hiding implementation [15].

Today, the most prevalent use of Web services is to carry out business processes and transactions, inside or across enterprises, by applying web service composition languages like BPEL4WS. However, there exist few studies of employing web services for the management of organizational resources in workflow. Recently, Aalst et al. [1] introduced their work on how to expose an organizational model in an XML DTD for the exchange of information in cross-organizational processes. But, their approach still bears tightly coupled architecture.

Managing organizational resources by Web services can be a sound alternative by utilizing tools and standards of Web services. Two of the existing works that can be employed to adapt workflow resources to Web services are introduced.

***Intermediary server***

An intermediary server is a component that lies between the service client (subscriber) and the service provider (publisher). It basically intercepts the request from the service client, provides the service of designated function, and forwards the request to the service provider. Similarly, it intercepts the response from the service provider and forwards it to the service client. Intermediary servers can provide value-added services, like authentication, quality of service, management service, or aggregation services. In our approach, we employ intermediary servers as to provide a uniform service of combining different types of organizational resources into a single unified interface.

***Web Service Management Framework***

Web Services Management Framework (WSMF) is proposed for the management of various IT resources, including Web services themselves, through Web services [15]. To encounter various management issues of more complex IT resources, an extensible interface

has been defined utilizing Web services standards and description language. However, WSMF focuses on the managerial aspects of IT resources, such as network management, system management, or database management, which are far from the management issues of organization resources, such as work reservation, work coordination, or organization charts.

## 4. UNIVERSAL RESOURCE MANAGEMENT FRAMEWORK

Now, we introduce a web service based workflow resource management framework named *Universal Resource Management Framework (URMF)*. URMF provides several web service interfaces that support in browsing organization chart and belonging resources, interacting with human and non-human resources, and the dynamic selection of proper resources based on constraints.

### 4.1. Design principles of URMF

The design principles of URMF, differentiated from the traditional distributed computing, are based on the concept of the service-oriented architecture (SOA) [20] to make the workflow resource management facility conform to the Web service environment:

- ***Standardized.*** The URMF architecture conforms to the web service standards (SOAP, WSDL, and UDDI) including the web service based process definition languages such as BPEL4WS. Resources declared and managed in URMF are universally discovered, bound and executed in web service environment.
- ***Loosely coupled by coarse-grained interfaces.*** Distributed object based systems are moving toward coarser-grained interfaces, which act as distributed façades, for less communication with each other across the network and loosely coupled implementation [20]. URMF provides coarse-grained web service interfaces, by employing generic organization model and resolving resources without the direct exposure of the entire organizational model. These features allow the flexible enrichment of organizational meta models, the employment of diverse underlying technologies, e.g. LDAP (Light-weight Directory Access Protocol) or SQL, and the flexible coordination of the services.
- ***Asynchronous work coordination.*** Tasks in business processes, especially those performed by human resources, need to be queued waiting to be processed. URMF supports the message calls for work order asynchronously as well as synchronously by the event-driven mechanism, to be explained more detail in sections 4.3, 5.2 and 5.3.
- ***Dynamic selection of resources.*** URMF provides web service interfaces, for the dynamic selection of resources, which supports the various organizational constraints represented in Resource Query Language (RQL) as well as browsing organizational structure, to be explained more detail in Sections 4.3 and 5.5.
- ***Discoverable.*** All Web services, inside or outside of an organization, are discoverable as a benefit of SOA, either at design time or at run time. Organizational resources conforming to URMF can be discovered, at any time required regardless of the location, to be explained more detail in Sections 4.3 and 5.6.

### 4.2. Entities of URMF and their interactions

Fig. 3 shows the four main entities of URMF, i.e., *WSWE* (*Web Services-based Workflow Engine*), *URB* (*Universal Resource Binder*), *OWQS* (*Organization Work Queue Service*), and *AWQS* (*Agent Work Queue Service*).

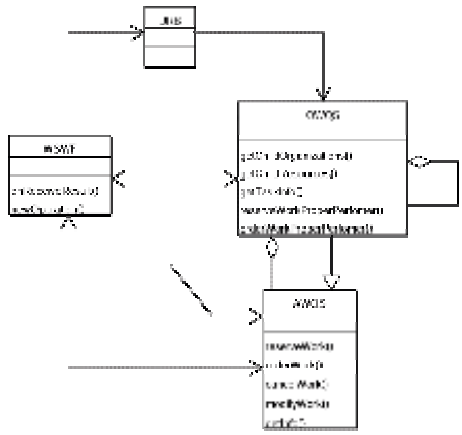

**Fig. 3.** Entities of URMF and their interactions

In URMF, every organizational workflow resource is represented as a web service, called *Work Queue Service* (*WQS*), and acts as a service provider. Each *AWQS* is for a human or non-human resource [30] and takes the work requests of service requestors. Each *OWQS* is for an organization unit and acts as a hub or router for AWQS. On the other hand, WSWE or URB request the services provided by WQS. *WSWE* refers to the workflow engine capable of interacting with web services. *URB* refers to a GUI tool, intermediating WSWE and WQS, and supports for browsing organizational resources during process design or instantiation.

*AWQS* represents each workflow resource, such as a human agent, software, a machine, or a place. *OWQS* inherits the interfaces of AWQS, capable of acting as AWQS, and references one or more child AWQS or OWQS. This self-contained structure allows OWQS to cover any set of resources like companies, branches, departments, or teams. This composite pattern [13] allows each service unit to be declared for any number of WQS's and lets URMF be applied to the organizations of any scale either in size or in complexity of policies. This advanced feature additionally allows OWQS to provide the functionality as an intermediary server. For example, OWQS can delegate or distribute work items to the appropriate child resources under its policy of delegation rules, access control, or security, etc. Furthermore, OWQS provides the RQL interfaces so that the service requestors can get the endpoint of the desired resource without browsing the whole organization charts inside. In this way, should-be secure

objects, like level or delegation, are encapsulated. These structural features of URMF entities support the design criteria of loosely coupled and coarse-grained interface.

### 4.3. Services of URMF

With the design principles and the entity model, we have declared five categories of services for URMF. Specific Web service descriptions of URMF entities, classified into these categories, are provided in appendix A. In addition, an example process definition using these services in BPEL4WS is provided in appendix B.

- ***Organization Chart Service*** provides several subordinate services to access organizational structure and resources. This service is mostly invoked at the design time or when a process is instantiated, by a process designer or an administration tool with the functionality of URB, for the actual binding of endpoints of target resources.
- ***Resolution Service*** enables service requestors to select resources dynamically with the Resource Query Language (RQL). This service returns the endpoint of proper AWQS according to the input query, which can be dynamically built according to the flow of process instance. XML-based string manipulation expressions, e.g. 'Xpath' [10], can used to build changeable RQL statements automatically.
- ***Work Coordination Service*** enables service requestors to reserve, order, cancel, and modify work items to the target resources. It also provides the runtime-state information of started tasks and reservations. The service requestor may call the AWQS direct through the endpoints of each workflow resource. But this direct access to resources may cause security problems and is restrictively to be used only inside an enterprise. On the other hand, the service requestor may call the OWQS and the corresponding OWQS delegates the work order to the final AWQS. This delegated work coordination provides better security management in the form of an intermediary server.
- ***Event Service*** enables service requestors to subscribe and listen to the events of interested resources, tasks, or changes in the organization. This service allows the service requestors to browse the list of events of the objects and subscribe to the event with the event key. Task related events, like a work is reserved or queued, are notified to the corresponding service requestor without subscription.
- ***Universal Discovery Support*** URMF requires each WQS instance to have its own address of web service endpoint URI. With this universal addressing, each resource in URMF can be registered to and discovered by the UDDI directory. By utilizing some taxonomy of role keys at WQS registration, resource searching can be further enhanced.

### 4.4. Web service based workflow interface model with URMF

Fig. 4 illustrates the substitutive workflow interface model, modified from [30], employing URMF and underlying web services architecture. Each interface 1 through 5 is mapped with corresponding Web services standards. Each component in the interface model takes advantage of the Web services infrastructure such as universal discovery, transaction management, or security management. In this way, the URMF architecture offers workflow resource management capabilities in a standard way according to the evolving Web services specifications.

Web service composition languages (WSCL), like BPEL4WS [3], conform to the web service architecture and can be extended to describe resources by URMF. Interface 1 in our model (*IF1* in Fig. 4) focuses on the import and export of process definitions in WSCLs. In our interface model, web service interface AWQS substitute worklists of each end-user (*IF2*) or the non-human resources like software agents (*IF3*). Sharing URMF-conformed resources, with other workflow engines, provides a way of enhancing interoperability (*IF4*).

Finally, the Universal Role Binder (URB), a GUI tool of URMF, interacts with process designer, administration/monitoring tool, and work queue services. URB supports end-users in searching and binding the resources in the way of web service as shown in Fig. 4.

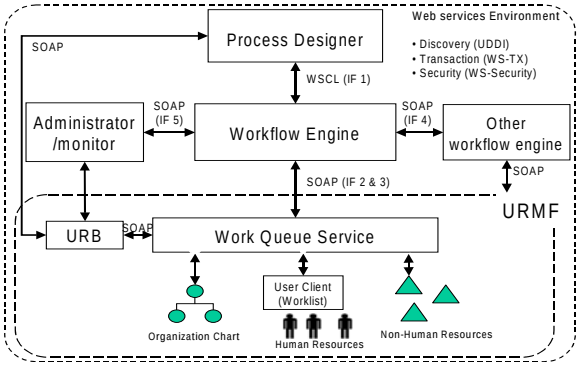


**Fig. 4.** Web Service based workflow interface model with URMF

As illustrated in Fig. 4, URMF complements the resource management facility of Web services environment. Since the interfaces and the tool declared in URMF extract and provide the resource management functionalities, the efforts of design and implementation on this part are relieved when to expose workflow resources as web services. In next section, a prototype implementation model of exposing workflow resources as web services with URMF is introduced.

## 5. PROTOTYPE IMPLEMENTATION MODEL

In this section, we show an example implementation model of URMF and how the web service requests are processed by the implemented system. Two suggestions for the realization of universal discovery support in

URMF are also presented.

Fig. 5 illustrates the system architecture of the prototype implementation model. For better management of reliability, security, and scalability, we adopted one of the most concurrent technologies, i.e. J2EE platform [28]. The interfaces and components to support URMF are managed by the application server, and intermediate the service requestors and the workflow resources of service providers. Service requests in the form of web service invocations are received at the two interfaces (AWQS and OWQS), processed at the corresponding components, and provided with low-level access to the adequate resources. More detailed descriptions of the major components in the implemented model are provided.

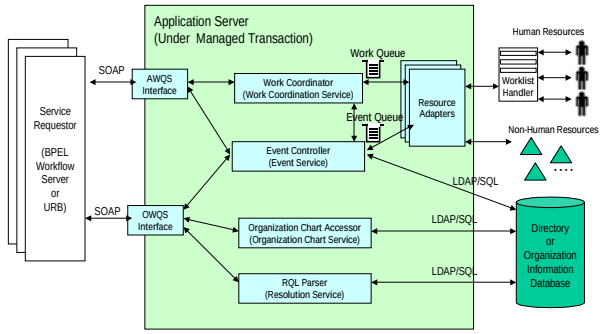


**Fig. 5.** System architecture of the prototype URMF Implementation

- ***Work Coordinator***

*Work Coordinator* processes the requests of reserving, queuing, canceling and modifying work items from AWQS interfaces. For the reliable management of work requests, MQM (message queuing middleware) and managed-transaction of J2EE platform are adopted ([27], [28]). Each work order or work reservation is queued into the message queue named '*WorkQueue*' and is ensured to be properly processed. The *resource adapters* let the work requests be uniformly processed at the heterogeneous system resources. *Worklist handler* provides a user interface for end users to handle workitems with adequate tools.

- ***Event Controller***

*Event Controller* manages the subscription and notification of events. Event Controller listens to the events of interested resources, tasks, or organization changes. Each event fired from other components is queued into the message queue named '*EventQueue*'. Event Controller distributes them to the subscribed service requestors (URB or WSWE). Task related events, e.g., 'onResult' or 'onReservationResult' of appendix A, are notified to the corresponding service requestor even without subscription.

- ***Organization Chart Accessor***

*Organization Chart Accessor* processes the requests for accessing organizational structure or belonging resources on behalf of the *organization chart services* of WQS interfaces. Each request is processed by accessing the directory services through LDAP [18] or the organization information database through SQL.

- ***RQL Parser***

*RQL Parser* executes the requests for the endpoint of eligible resource (WQS) on behalf of *resource resolution services* of OWQS. The inquiry RQL statements are parsed and dispatched through LDAP or SQL.

- ***Universal discovery support***

Web services are discoverable by their own nature. To be discovered by UDDI, WQS of each resource needs unique address and categories. Two suggestions to handle these issues are described.

***WQS endpoint (URI) addressing:*** WQS of each resource in an organization needs a unique address, i.e. endpoint URI. Fig. 6 shows the suggested endpoint addressing policy for WQS. It is similar with e-mail addressing but contains more organizational information, i.e. names of department, branch or company as well as and user id.

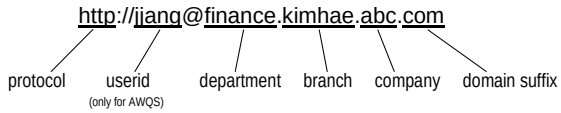


**Fig. 6.** Suggested WQS endpoint (URI) addressing

***Role-based Taxonoy:*** Mutually aware taxonomies are needed to search for adequate web services through UDDI. We suggest employing role-based taxonomy to be utilized in searching for adequate WQS of a resource. Utilizing role-based taxonomy, WQS of adequate human resources with required role or skill could be searched from UDDI servers.

Fig. 7 illustrates the suggested mechanism of universal discovery support, which allows each organization or person to advertise itself, as a resource, over the Web service world.

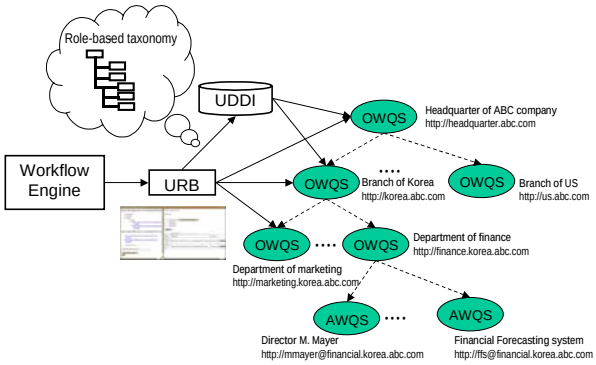


**Fig. 7.** Suggested mechanism of universal discovery support

## 6. CONCLUDING REMARKS

Organizational elements, such as organization chart, roles, and worklists for human or non-human agents, play important roles in executing business processes [1]. For the successful realization of web service based workflow management, a sound management of these elements should be provided. However, there is no

robust standard to expose and access workflow resources by Web service interfaces. In this paper, we proposed a web service based workflow resource management framework named *Universal Resource Management Framework* (URMF) with declarations of web service interfaces and interaction protocols among them. We also introduced a substitutive workflow interface model employing Web services and URMF, where each component of the model can take advantage of Web services infrastructure. To validate the viability of the suggested framework, a prototype implementation model of URMF with J2EE platform is also introduced. In this way, management of business processes composed of Web services is empowered great with the support of URMF in managing organizational resource.

Our framework is fully based on the service-oriented architecture (SOA) [20]. The resources, especially human resources, conforming to URMF are universally interoperable and discoverable as web services, and can be flexibly coordinated in the way of Web services standards. Loosely coupled and coarse-grained URMF interfaces, by the generic organization model and resolving resources with the encapsulation of the entire organizational model, features minimal technical requirements for the implementation.

Some of important issues related with the Web services environment are not dealt in this paper and left as further studies. First, Non-authorized work request or access to the organization should be prohibited. Since URMF is based on Web services, Web services security technologies may leverage the security of our framework. The SSL certificates with HTTPS or WS-Security [17] are examples of those technologies. Some access control mechanism may be applied to the URMF at the interface or operation level. For instance, OWQS can act as an intermediary server for security and provide an access control mechanism based on the service requestor or information requested. Second, an adequate business transaction mechanism to handle long-lived activities should be supported with URMF. To minimize latency of access, for other potential users, to the resources used by such activities, the results of interim operations need to be released before they are completed. URMF needs to be enhanced by applying such frameworks for transaction management, like WS-Transaction [4] or WS-CAF [2], which coordinates the execution of distributed operations in a Web services environment.

## ACKNOWLEDGEMENT

This work was supported by the 2003 Inje University research grant.

## REFERENCES

[1] Aalst, W.M.P. van der, A. Kumar, and H.M.W. Verbeek, "Organizational Modeling in UML and XML in the context of Workflow Systems", *18th Annual ACM Symposium on Applied Computing*, ACM Press, 2003, pp. 603-608.

[2] Arjuna Technologies Ltd., Fujitsu Limited, IONA Technologies Ltd., Oracle Corporation, and Sun Microsystems, Inc., "Web Services Composite Application Framework (WS-CAF). Version 1.0", July 2003.

[3] BEA Systems, IBM Corporation, and Microsoft Corporation, Inc., "Business Process Execution Language for Web Services, Version 1.1", http://www.ibm.com/developerworks/library/ws-bpel/, May 2003.

[4] BEA Systems, IBM Corporation, and Microsoft Corporation, Inc., "Web Services Transaction (WS-Transaction)", http://www.ibm.com/developerworks/library/ws-transpec/, August 2002.

[5] Bussler, C., "Enterprise-wide workflow management", *IEEE Concurrency*, Vol. 7, Issue 3, 1999, pp. 32-43.

[6] Bussler, C. and S. Jablonski, "Policy resolution for workflow management systems", *28th Hawaii Int. Conf. on System Sciences*, Vol. 4, 1995, pp. 831–840.

[7] Bussler, C., "Analysis of the Organization Modeling Capability of Workflow-Management-Systems", *the PRIISM '96 Conf.*, January 1996.

[8] Chappell, D. A. and T. Jewell, *Java Web Services*, O'Reilly & Associates, Inc., March 2002.

[9] Cheng, E. C., "An Object-Oriented Organizational Model to Support Dynamic Role based Access Control in Electronic Commerce Applications", *the 32nd Annual Hawaii Int. Conf. on System Sciences*, 1999.

[10] Clark, J. and S. DeRose (ed.) "XML Path Language (XPath) Version 1.0", W3C, November 1999, available at http://www.w3.org/TR/xpath

[11] Cummins, F. A., "Workflow Resource Management Facility Request for Proposal (Draft)", *OMG Document bom/99-04-01*, Framingham 1999.

[12] Du, Weimin, J. Davis, Y.-N. Huang, M.-C. Shan, "Enterprise Workflow Resource Management", *HP Tech. Report*, HPL-1999-8, 1999.

[13] Gamma, E., R. Helm, R. Johnson, and J. Vlissides, *Design Patterns—Elements of Reusable Object-Oriented Software*, Addison-Wesley, 1995.

[14] Halliday, J.J., S.K. Shrivastava, and S.M. Wheater, "Implementing support for work activity coordination within a distributed workflow system", *3rd Int. Conf. on Enterprise Distributed Object Computing Conference*, 1999, pp. 116-123.

[15] Hewlett-Packard Development Company, "Web Services Management Framework – Overview Version 2.0", July 2003.

[16] Huang, Y.-N. and M.-C. Shan, "Policies in a Resource Manager of Workflow Systems: Modeling, Enforcement and Management", *HP Tech. Report*, HPL-98-156, 1998.

[17] IBM Corporation, Microsoft Corporation, and VeriSign, Inc., "Web Services Security (WS-Security) Version 1.0 05", http://www.ibm.com/developerworks/library/ws-secure/, 2002.

[18] LDAP, *Understanding LDAP*, 1998, http://www.redbooks.ibm.com/redbooks/pdfs/sg244986.pdf

[19] Leymann, F., D. Roller, and M.-T. Schmidt, "Web services and business process management", *IBM Systems Journal*, Vol. 41, No. 2, 2002, pp. 198-211.

[20] McGovern, J., S. Tyagi, M. Stevens, and S. Mathew, *Java Web Services Architecture*, Morgan Kaufmann, 2003.

[21] Object Management Group (OMG), "Workflow Resource Assignment Interfaces (RAI) Request For Proposal", *OMG Document* bom/2000-01-03

[22] Paul, S., E. Park and J. Chaar, "RainMan: A Workflow System for the Internet", *Proc. of USENIX Symp. on Internet Technologies and Systems*, November 1997.

[23] Rupietta, W., "Organization and Role Models for Workflow Processes", *In Lawrence, P.(Ed.) The Workflow Management Coalition Handbook*, Winchester 1997, pp. 165-172.
[24] Rupietta, W., "Organizational Models for Cooperative Office Applications", *In D. Karagiannis (Ed.) Database and Expert Systems Applications. the 5th Int. Conf. DEXA '94*, Springer Publishers, September 1994.
[25] Sheth, A. P., W. van der Aalst, and I.B. Arpinar, "Processes driving the networked economy", *IEEE Concurrency*, Vol. 7, Issue 3, 1999, pp. 18-31.
[26] Stohr, E. A. and J. L. Zhao, "Workflow automation: Overview and research issues", *Information Systems Frontiers*, 3(3):281-296 (2001).
[27] Sun Microsystems, Inc., "Java Message Service, v1.0.2b", August 2001, http://java.sun.com
[28] Sun Microsystems, Inc., "JavaTM 2 Platform Enterprise Edition Specification, v1.4", April 2003, http://java.sun.com
[29] Swenson, K. D., "Trouble Ticket Workflow Scenario (Version 2)", *OMG Document* bom/98-02-09.
[30] Workflow Management Coalition, *The Workflow Reference Model: Document Number TC00-1003*, 1995, available at http://www.wfmc.org.
[31] zur Muehlen, M., "A Framework for XML-based Workflow Interoperability - The AFRICA Project", *Americas Conference on Information Systems* (AMCIS 2000), pp. 13-18.
[32] zur Muehlen, M., "Evaluation of Workflow Management Systems Using Meta Models", In the 32nd Annual Hawaii Int. Conf. on System Sciences, 1999.
[33] zur Muehlen, M., "Resource modeling in workflow applications", *Proceedings of the 1999 Workflow Management Conference*, 1999, pp. 137-153.
[34] zur Muehlen, M., "Organizational Management in Workflow Applications", *Information Technology and Management Journal*, Kluwer Academic Publishers, Vol. 5, No. 3, 2004.

## APPENDIX A. SERVICE DESCRIPTIONS OF URMF ENTITIES

**Table A1.** Service description of AWQS

| Service Type | Service Name | Input & Output |
|---|---|---|
| Work Coordination Service | reserveWork | I: Role Key, Reply Endpoint, Expected Order Date, Due Date; O: Task (Reservation) Key |
| | orderReservedWork | I: Task (Reservation) Key, Parameter (any type); O: Task Key |
| | orderWork | I: Role Key, Reply Endpoint, Due Date, Parameter (any type); O: Task Key |
| | cancelWork | I: Task Key, Reason; O: None |
| | modifyWork | I: Task Key, Due Date, Parameter (any type); O: None |
| | modifyReservedWork | I: Task Key, Expected Order Date, Due Date; O: None |
| | getTaskInfo | I: Task Key; O: Task Information |
| Event Service | subscribeEvent | I: Event Type, Reply Endpoint, Duration; O: Event Key |
| Organization Chart Service | getInfo | I: None; O: Resource information |

**Table A2.** Service description of OWQS*

| Service Type | Service Name | Input & Output |
|---|---|---|
| Organization Chart Service | getChildOrganizations | I: None; O: Endpoints of child OWQS's |
| | getParentOrganization | I: None; O: Endpoint of the parent OWQS |
| | getAgents | I: None; O: Endpoints of child AWQS's |
| | getRoleKeys | I: None; O: List of all the role keys the organization resources can play |
| Resolution Service | getProperPerformer | I: RQL Statement; O: Endpoint of the resolved WQS |
| Resolution Service + Work Coordination Service | reserveWorkProperPerformer | I: RQL Statement, Reply Endpoint, Expected Order Date, Due Date; O: Task (Reservation) Key |
| | orderWorkProperPerformer | I: RQL Statement, Allocation Policy, Reply Endpoint, Due Date, Parameter (any type); O: Task Key |

* Basically, OWQS inherits all the operations of AWQS as described in 4.2.

**Table A3.** Service description of WSWE

| Service Type | Service Name | Input & Output |
|---|---|---|
| Event Service | onResult | I: Task Key, Result of work (any type); O: None |
| | onReservationResult | I: Task (Reservation) Key, Result of reservation; O: None |
| | onEvent | I: Event Type, Event Context; O: None |

## APPENDIX B. AN EXAMPLE BPEL4WS CODE EMPLOYING URMF

The following BPEL4WS code illustrates how the human resources can be dynamically selected and bound with URMF. The code describes parts of the trouble ticket process[1], where each problem instance is recorded with human

[1] The trouble ticket scenario by OMG [29] is provided for the purpose of evaluating workflow facilities in the perspective of human intervention.

intervention. The process is initiated by the former "receive" activity, which receives the endpoint of the OWQS of the organization in charge of the given problem instance. Based on the request, the "assign" activity builds a RQL statement of inquiring eligible resource and set parameters for the invocation of corresponding WQS. The "invoke activity" calls the interface ('orderWorkProperPerformer' in this example) of the WQS for the resolution and assignment of this workitem. Actual selection and binding of the proper human resource (i.e. queuing the workitem in the corresponding worklist) is executed by the *resource resolution service* and the *work coordination service* of WQS. The latter "receive" activity waits for the result performed by the AWQS of actually assigned human resource.

```
<!-- declaration of message types, variables, and correlation sets will be used -->

<message name="TroubleTicketRequestType">
    <part name="originatorOWQSServiceReference" type="wsa:EndpointReference"/>
</message>

<containers>
    <container name="troubleTicketRequest" messageType="tns:troubleTicketRequest"/>
    <container name="OWQSRequest" messageType="owqs:WorkRequest"/>
    <container name="taskKey" messageType="wqs:TaskKey"/>
    <container name="problem" messageType="wswe:WorkRequest"/>
</containers>

<correlationSets>
    <correlationSet name="TaskIdentification" properties="tns:TaskKey"/>
</correlationSets>

<!-- the process will be initiated with the originating organization's service reference (OWQS endpoint) -->
<receive partner="originator" portType="tns:TroubleTicketPT"
    operation="initiate" container="troubleTicketRequest"
    createInstance="yes" name="TroubleTicketReceive"/>

<!-- build parameters for invoking AWQS -->
<assign>
    <!-- provide a RQL for resolving the proper performer for this activity -->
    <copy>
        <from expression="'require humanworker for TroubleTicket:RecordingProblem where branch.location=KR'"/>
        <to container="OWQSRequest" part="rqlStatement"/>
    </copy>

    <!-- provide the reply endpoint for the result of the activity -->
    <copy>
        <from expression="'http://korea.abc.com/bp/troubleticket'"/>
        <to container="OWQSRequest" part="replyEndpoint"/>
    </copy>

    <!-- copy the service(endpoint) reference of originator's AWQS --> <copy>
        <from container="originatorOWQSServiceReference"/>
        <to partner="recorder"/>
    </copy>
</assign>

<!-- queuing a workitem into the performer's worklist -->
<invoke name="RecordingProblem"
    partner="recorder"
    portType="owqs:OrganizationWQS"
    operation="orderWorkProperPerformer"
    inputContainer="OWQSRequest"
    outputContainer="taskKey">
    <!-- correlation with the Task key that was returned as result of invocation -->
    <correlations>
        <correlation set="TaskIdentification" initiation="yes"/>
    </correlations>
</invoke>

<!-- receive the result of the requested workitem -->
<receive name="ReceiveResultFromRecordingProblem"
    partner="recorder" portType="wswe:WSWE" operation="onResult" container="problem">
    <correlations>
        <correlation set="TaskIdentification" initiation="yes"/>
    </correlations>
</receive>
```