Optimization of the communication among the Workflow Engine and the Client Applications according to the WfMC standard

N. Debnath[1], P. Martellotto[2], M. Daniele[2], D. Riesco[3], G. Montejano[3]
[1]Department of Computer Science – Winona State University – MN 55987 – USA, ndebnath@winona.edu
[2]Universidad Nacional de Río Cuarto – Argentina, {paola, marcela}@dc.exa.unrc.edu.ar
[3]Universidad Nacional de San Luis – Argentina, {driesco, gmonte}@unsl.edu.ar

Abstract

The Workflow Management Systems allows define, create and manage the execution of the business processes on one or more Workflow motors.

The Workflow Management Coalition (WfMC) has developed a Workflow Reference Model with different interfaces defining a generic structure of Workflows, in order to the interoperability of different Workflows. The WfMC has specified a group of WAPIs (Workflow Application Programming Interfaces) for the management of Workflows. These WAPIs defines the functions of the interfaces like calls to APIs in a third-generation language, forcing that when invoking an application it should be known specifically where this application is located.

A Web Service is a service that is offered by means of the Web, and facilitates the reusability of an application in different platforms or programming languages.

Through this work we propose to take advantage of the benefits from the Web Services, using them to specify the functions of the Workflow Client Application Interface, ensuring communications among the Workflow with the applications. This way, the user of the Workflow should not know the exact location of the application to invoke, and the applications can change their location in the net without any change in its invocation.

Keywords: Workflow, Workflow Reference Model, Workflow Client Application Interface, specification, Web Service.

1. Introduction

The Workflow Management Systems are being developed in companies like a new form of organizing and managing the information, helping to automate the business processes. Such systems can predefine a piece of work procedure with the relevant information, the roll of the participants in each work step, and the required software application to process each step. Each new work procedure is assigned automatically to the participants in an actual sequence, the information is given to the person who needs it and the applications are executed when it is necessary.

In order to be able to have a certain level of interoperability among the diverse products of Workflow, it is necessary to define a set of interfaces and formats for the interchange of data between this components.

The Workflow Management Coalition (WfMC) has developed a Workflow Reference Model [7] identifying the interfaces with the generic structures of the applications of Workflow, to allow products communicate at different levels. In particular, to allow the interaction between users with the Workflow Engine a Worklist is used, that is handled by an administrator. The Workflow Client Application Interface is the one in charge to manage the interaction between the Workflow Engine and the administrator of the Worklist. The WfMC has specified a set of APIs (Application Programming Interfaces) for the management of Workflows [4], which are supported by the Workflow products and they are denominated Workflow Aplication Programming Interfaces (WAPIs).

The Workflow Reference Model [7] was developed from generic structures of Workflow applications, identifying the interfaces with these structures, to allow products to communicate at different levels. All Workflow systems contain generic components which interact in a predefined way.

On the other hand, the Web Services are car-contained, car-described applications that can be published, located and invoked through the Web, without the necessity of knowing the exact location of the same ones [1].

The well-known implementations of the Workflow Client Application Interface [8] define the functions of the interfaces and its specification, like calls to APIs in C language, and for the invocation of an application it is required to know its location specifically. In this work a specification of this Interface sets out using Web Services, with the aim to prevent the user of the Workflow from knowing the location of the application that wants to invoke, and to allow any application to change its location in the net without implying any change in its

invocation.

WSDL (Web Service Description Language) [4] [5], based on XML [2], is used for the description of a Web Service. Also, it is necessary to describe the messages among the applications and the web service, and the form in which these will be transported through the Web. SOAP (Simple Object Access Protocol) [3] is the better known protocol based on messages that are used to describe the interaction of the applications with the web services.

On the other hand, the more used transport protocol is HTTP (Hyper Text Transport Protocol). Finally, it is necessary to register and locate the web service, for it defines a directory of distributed Web Services that allows them to be listed, looked for and discovered. Generally this directory is UDDI (Universal Description, Discovery and Integration) [6]. The use of these standard protocols allows achieving the interoperability, with independence of the Operating System, programming language, etc.

2. Background

There are diverse works that present alternatives to implement the communication of the Workflow with the applications of clients. In [9] the WfMC presents a proposal of specification of the Interfaces 2 and 4 based on the use of IDL and bindings with OLE as alternatives to existing "C" and MIME specifications. The early works of the WfMC (on WAPI) concentrate on the definition of the interface functions and their specification as APIs calls in "C" as the key language binding. The interoperability specification was subsequently developed using IDL for the abstract specification and concrete bindings based in MIME, to use via e-mail. The OLE automation binding has been derived from defining an OLE automation class and property for each WAPI data structure and defining a method on the appropriate object class. WAPI query manipulation is replaced by the use of OLE collection type objects and a filter object is defined to replace the WAPI Filter datatype, to be used as a parameter of methods on collection type objects. Collection objects in OLE support a count parameter. The IDL binding of the WfMC standards defines an object model that combines the Interfaces 2, 4 and 5, and it also addresses the area of Invoked Applications, where applications are assumed to the Business Objects. A jFLOW specification is defined using UML.

In [10] the authors present the JointFlow specification. This specification is a submission of 19 companies in response to OMG's Workflow Management Facility RFP. This specification is based on the work of the Workflow Management Coalition and defines the interfaces that support the runtime interaction with workflow components, enable interoperability of workflow components across business domains and allow for monitoring and auditing the workflow processes. The JointFlow specification can be used to do distributed workflow applications.

In [11] the authors present a specification using Grid Computing. In the traditional WFMS (Workflow Management Systems) the processes are designed with process definition tools and executed by workflow engines in the same workflow management system. Tasks are performed by end users or applications. Processes can only be executed by the engines that can be accessed by specific users and specific applications. Therefore, to use WFMS is restrictive by their locations. In this proposal the tasks are performed by grid services (which are based on a standard interfaces set). So the execution of a process is not limited to the workflow engines location and processes can be designed, stored and executed anywhere.

In [12] the authors introduce a workfow composer agent, which is able to compose web services workflows, uses semantic descriptions of web services in finding and matching web services for a workflow. It is shown how a workflow can be composed by using semantic web services ontologies, which refers to define the semantics of a web services, that is to say, its meaning, more than its input and output parameters. With this technology a model is presented to compose web services workflows. The model for the workflow composition describes what happens when a new instance of web service is considered. The instance is published in a directory.

In [13] the authors propose to encapsulate the organization's functionality within an appropriate interface and publish it as Web Services. It is expected that the Web Services to be integrated as a part of web processes. But this integration is difficult given the degree of heterogeneity, autonomy and distribution of the web. A solution is the use of ontology. It is essential for the web service to support all phases of the web process lifecycle. It describes applying semantics to each step in the Semantic Web Process lifecycle can help to the reuse, the integration and scalability.

In [14] the Workflow language presented uses web services as components, architecture for a run-time environment for this language, and takes advantage of the use of this kind of technology. The idea is to develop business processes formed by a compound of services that are available in a

computer net. The Workflow technology is used to coordinate the interactions with the web services. The web services represent the logical steps that compose a workflow. The proposed language AELCWS doesn't support the direct interaction with people during the execution of a process, so the Workflow Client Application Interface is not defined nor is implemented. The Invoked Applications Interface is supported by this language through the invocation of the services belonging to process definition.

3. The Workflow Reference Model of the WfMC

The prevailing tendencies in the world of businesses force the companies to reconsider and to redesign their technological support, with the purpose of: increasing the productivity, improving the quality, improving the service to the client, reducing the costs and adapting to a changing environment.

A Workflow is the automation of the business processes where the documents, information and tasks are applied among the participants of the system according to a set of rules previously established [7].

On the other hand, a WFMS is a system that defines, creates and manages the execution of Workflow through software that is executed on one or more Workflow engines, which interpret the definition of the processes; they interact with the other participants of the Workflow and invoke tools and applications [7].

In order to allow the interoperability among the diverse Systems of Workflow, it is necessary to define a set of interfaces and formats for the interchange of data between these components. The Workflow Reference Model [7] proposed by the WfMC identifies the interfaces with generic structures of applications. All the systems of Workflow contain generic components that interact in a predefined way.

In this model there is a separation between the processes and the control of the logic of the activities. This logic is within the Workflow Enactment Service. This separation allows the integration of the different tools with a particular application.

The Workflow Enactment Service interprets the description of processes and controls the different instances from the processes, sequences the activities, adds items to the Worklist of the users and invokes necessary applications.

The interaction of the Workflow Enactment Service with the external resources occurs through one of the two following interfaces:

- The Workflow Client Application Interface, through which the Workflow engine interacts with the administrator of the Worklist, is responsible for organizing the work through user's resource.
- The Invoked Applications Interface allows the Workflow engine to activate a tool to make a particular activity. This interface could be based on a server, that is to say, the interaction with the user does not exist.

The Worklist allows control the interaction with the users. The engine places in the Worklist the items which have to be executed by each user.

The Worklist manager controls to the interaction among the participants of the Workflow and the Workflow Enactment Service through the Worklist. Also, it supports a wide interaction range with other application clients.

4. Specification of the Workflow Client Application Interface with Web Services

The Workflow Client Application Interface allows the interaction between the applications of clients and the Workflow engine. In order to maintain this interaction a Worklist is used to store the information of the applications to invoke. The APIs associated with this interface provides a basic level of functionality to support the invocation of applications. These APIs can be consulted in [8].

The functions of the Worklist provide information to the participants of the Workflow on the works that they have assigned. Each work piece assigned to a participant of Workflow is denominated Work Item.

In order to specify the Workflow Client Application Interface of the Workflow Reference Model with Web Services this work proposes to specify each of the functions defined with WAPIs in this model. As an example, the specification of the WMOpenWorkList function is presented; it provides information to the participants with the Workflow on the works that they have assigned [8].

The function returns the list of work items assigned to a participant of a particular Workflow or a work group, according to a filter criterion.

For this function, one of the elements to define is the types that the web service will use (Figure 1). These types are declared inside the element <types>. A scheme XML is used to define the types that are more complex.

WSDL allows separate the description of the abstract functionality of a Web Service of the specific details on how and where the functionality is offered. An interface WSDL defines the abstract

```
<types>
  <xs:schema
    ...
    <xs:element name="psession_handle"
                type="Tpsession_handle"/>
    <xs:complexType name="Tpsession_handle">
    <xs:sequence>
      <xs:element name="session_id" type="xs: String"/>
      <xs:element name="pprivate" type="xs: String"/>
    </xs:sequence>
    </xs:complexType>

    <xs:element name="pworklist_filter"
                type="Tpworklist_filter"/>
    <xs:complexType name="Tpworklist_filter">
    <xs:sequence>
      <xs:element name="sqlString" type="xs: String"/>
      <xs:element name="attributeName" type="xs: String"/>
      <xs:element name="comparison" type="xs: int"/>
      <xs:element name="attributeValue" type="xs: boolean"/>
    </xs:sequence>
    </xs:complexType>
    ...
    <xs:element name="WMInvalidSessionHandle"
                type="TWMInvalidSessionHandle"/>
    <xs:complexType name="TWMInvalidSessionHandle">
    <xs:sequence>
      <xs:element name="excep" type="xs: String"/>
      <xs:element name="error" type="xs: String"/>
      <xs:element name="message" type="xs: String"/>
    </xs:sequence>
    </xs:complexType>

    <xs:element name="WMInvalidFilter"
                type="TWMInvalidFilter"/>
    <xs:complexType name="TWMInvalidFilter">
    <xs:sequence>
      <xs:element name="excep" type="xs: String"/>
      <xs:element name="error" type="xs: String"/>
      <xs:element name="message" type="xs: String"/>
    </xs:sequence>
    </xs:complexType>
</xs:schema>
</types>
```

Figure 1. Types

interface of a Web Service like a set of abstract operations (Figure 2).

Another element to specify is binding. It allows detail how the messages can be interchanged (Figure 3). It specifies details on the specific format of messages and the protocol of transmission for an interface and provides such details for any operation and fails in the interface.

Finally, the WMOpenWorkList service is defined (Figure 4). This definition implies to specify where the service can be accessed, by means of the use of the element service. A service WSDL specifies a simple interface that will support the service and a list of location of endpoints where that service can be accessed.

```
<interface name = "WMOpenWorkListInterface" >
  <fault name = "WMOpenWorkList_InvSessionHandle"
       element = "wfms: WMInvSessionHandle"/>
  <fault name = "WMOpenWorkList_InvFilter"
       element = "wfms: WMInvFilter"/>
  <operation name="opWMOpenWorkList"
       pattern=http://www.w3.org/2004/03/wsdl/in-out>
    <input messageLabel="In"
       element="wfms: psession_handle" />
    <input messageLabel="In"
       element="wfms: pworklist_filter" />
    <input messageLabel="In" element="wfms: count_flag" />
    <output messageLabel="Out"
       element="wfms: pquery_handle" />
    <output messageLabel="Out" element="wfms: pcount" />
    <outfault ref="tns: WMOpenWorkList_InvSessionHandle"
       messageLabel="Out"/>
    <outfault ref="tns: WMOpenWorkList_InvFilter"
       messageLabel="Out"/>
  </operation>
</interface>
```

Figure 2. Interface

This way the elements types, interface, binding and service are specified for the WMOpenWorkList operation, of the Workflow Client Application Interface using SOAP and WSDL. This allows define a new class of applications that use the web services distributed by the net, taking advantage of the benefits from the interoperability.

```
<binding name="WMOpenWorkListSOAPBinding"
    interface="tns: WMOpenWorkListInterface"
    type=http://www.w3.org/2006/01/wsdl/soap
    wsoap:protocol=
         "http://www.w3.org/2003/05/soap/bindings/HTTP">

  <operation ref="tns: opWMOpenWorkList"
    wsoap:mep=.../>

    <fault ref="tns: WMOpenWorkList_ InvSessionHandle"
    wsoap:code="soap:Sender"/>

    <fault ref="tns: WMOpenWorkList_InvFilter"
    wsoap:code="soap:Sender"/>
</binding>
```

Figure 3. Binding

```
<service name="WMOpenWorkListService"
    interface="tns: WMOpenWorkListInterface">
    <endpoint name="WMOpenWorkListEndpoint"
        Binding = "tns: WMOpenWorkListSOAPBinding"
        address = .../>
</service>
</description>
```

Figure 4. Service

5. Conclusions

The Workflow Management Systems are developed in companies with the objective of automating the work processes in the offices. Such systems can predefine a work procedure with the relevant information, the role of the participants in each work step, and the software application required to process each step. The Workflow Management Systems make this administration automatically and this way, the work is completed in a more efficient way.

The Workflow Management Coalition has developed a Workflow Reference Model to allow certain interoperability level among the diverse Workflow products. The WAPIs for the management of Workflows developed by the WfMC defines functions that are specific to a language and an architecture. This implementation requires that the invocation of an application implies to know specifically where it is located. The benefits of the Web Services can be taken from improving this implementation, by ensuring communications of the Workflow with the applications. With the use of Services Web it is achieved that the Workflow behaves internally in a distributed way, that is to say, it doesn't need to know where the applications are to invoke them. Simply, it requires services and there are applications that provide these services. Also, it facilitates the re-incorporation services, independently of how they are invoked.

In order to specify the Workflow Client Application Interface, the functions defined with WAPIs are specified, using Web Services. In this work a way to specify one of the functions, WMOpenWorkList, has been shown defining each element that composes a Web Service. In an analogous way, the other functions of this Interface can be specified with Web Services, achieving a specification independent of the programming language, and of the underlying platform.

6. References

[1]. World Wide Web Consortium. Web Service Architecture. http://www.w3.org/TR/ws-arch/, last access May 2007.

[2]. World Wide Web Consortium. Extensible Markup Language (XML) 1.0 (Second Edition). http://www.w3.org/TR/xml, last access May 2007.

[3]. World Wide Web Consortium. SOAP Version 1.2 Part 1: Messaging Framework. http://www.w3.org/TR/soap12-part1/, last access May 2007.

[4]. World Wide Web Consortium. Web Services Description Language (WSDL) Version 2.0 Part 0: Primer. http://www.w3.org/TR/wsdl20-primer, last access May 2007.

[5]. World Wide Web Consortium. Web Services Description Language (WSDL) Version 2.0 Part 1: Core Language. http://www.w3.org/TR/wsdl20, last access May 2007.

[6]. OASIS. UDDI Version 3.0.2. http://uddi.org/pubs/uddi_v3.htm, last access May 2007.

[7]. Workflow Management Coalition. The Workflow Reference Model. WfMC-TC00-1003. http://www.wfmc.org/standards/referencemodel.htm, last access May 2007.

[8]. Workflow Management Coalition. Programming Interface 2&3 Specification. WfMC-TC-1009. V2.0. http://www.wfmc.org/standards/publicdocuments.htm, last access May 2007.

[9]. Workflow Management Coalition. A Common Object Model Discussion Paper. WfMC-TC10-22. http://www.wfmc.org/standards/docs/TC-1022_commom_Object%20Model_Paper.pdf, last access May 2007.

[10]. Schmidt M.T., Building Workflow Business Objects. IBM Software Group OOPSLA'98 Business Object Workshop IV.

[11]. Yang M., Liang H., Xu B., S-WFMS: A service-based WFMS in Grid Environment. Proceedings of the 19th International Conference on Advanced Information Networking and Applications (AINA'05). IEEE. 2005.

[12]. Laukkanen M., Helin H.: Composing Workflows of Semantic Web Services. Workshop on Servicios Web and Agent-based Engineering. AAMAS'2003. Melbourne, Australia. 14/15 de Julio de 2003.

[13]. Cardozo J., Shelt A., Introduction to Semantic Servicios Web and Web Process Composition. Publication of LSDIS. Large Scale Distributed Information Systems. University of Georgia. Computer Science Department.

[14]. Hiane da S. Maciel L. A., Toshiro Yano E.: Uma Linguagem de WF Para Composicao de Web Services. XIX Simpósio Brasileiro de Engenharia de Software. Uberlandia, MG, Brasil. 2005.