

Flexible Inter-enterprise Workflow Management using E-Services

Jie Meng, Raja Krithivasan, Stanley Y. W. Su and Sumi Helal

Database R&D Center
Department of Computer and Information Science and Engineering
University of Florida, Gainesville, FL 32611-6125, USA

{jmeng, rkrithiv, su, helal}@cise.ufl.edu

Abstract

This paper presents a solution to achieve dynamic Inter-enterprise workflow management using the e-services provided by collaborative e-business enterprises. E-services are distributed services that can be accessed programmatically on the Internet, using SOAP messages and the HTTP protocol. We categorize e-services according to their business types and manage them in an UDDI-enabled Broker Server. By E-service requests are specified in the activities of a process model according to some standardized e-service templates and are bound to the proper service providers at run-time by using a constraint-based, dynamic service binding mechanism. The workflow management system is dynamic in the sense that the actual business organizations that take part in a business process are not determined until run-time.

1. Introduction

In the highly competitive and rapidly changing global economy environment, business organizations need to collaborate to achieve common business goals in a more flexible and effective way than ever before. Recently, the use of workflow technology to manage e-businesses has drawn much attention in the academic community [Alo99, Laz01, Gre99, Str00, She99]; however, a good solution to support dynamic workflow management for e-business is still missing.

Business organizations across the Internet have different resources and provide manual and automated services for the manipulation and access of these resources. To conduct a joint business venture, an inter-enterprise workflow management system is needed to integrate data resources, workflow processes, and services provided by collaborative business organizations. Since business organizations

can enter and leave the Internet world freely, their memberships in a virtual enterprise and their services may change from time to time. The dynamic nature of services and their providers requires that service requests specified in a process model be dynamically bound to services at the time when an instance of the process model is in execution.

We have designed and are implementing a dynamic workflow management system to support e-businesses [Men01, Hel01b]. The system is active, adaptive, and flexible. By integrating the system with an Event-trigger-rule Server and an Event Server [Lam98, Lee00, Su00], the enactment of a workflow process may post events to trigger the processing of business rules to enforce security and integrity constraints, business policies, and regulation, etc. These rules may in turn activate other workflow processes, thus making the workflow management system an active system. The triggered rules may also modify process models at run-time to adapt the models to the changing business situations. This paper focuses on the flexible aspect of the workflow management system. It presents a mechanism for dynamically binding e-service requests, which are specified in the activities of a process model, to the proper e-services and e-service providers.

We define an e-service as “any service or functionality that can be accessed by a business or a consumer programmatically over the Internet by using a standard service specification and a standard communication protocol”. *E-service templates*, which are used to standardize the specifications of e-services, are stored and maintained by a Broker Server. Business organizations register the e-services they provide with the Broker Server according to the e-service templates, and the service specifications are also managed by the Broker Server. The activity definitions in a process model contain one or more *e-service requests*, which are specified according to the *e-service templates* and are bound to the proper service providers at run-time by a *dynamic service binding* mechanism. Changes in the membership of a virtual enterprise (i.e., its service providers) will not affect the process models. Thus, the workflow system has the flexibility of bounding service requests to the available and suitable services and service providers.

In this work, we allow e-service requests to be specified in workflow activity specifications; an extension to the traditional workflow process specification. We also introduce constraint definitions in both e-service specifications and e-service request specifications; an extension to the Web Service Description Language [Chr01]. These constraints are used by the Broker Server to do constraint-based matching of e-service requests with e-service specifications. Another new concept introduced in this paper is the specification of restrictions on the selection of providers (or performers) by taking into consideration the interdependencies among activities in a process model.

The organization of this paper is as follows. In Section 2, the architecture of the dynamic workflow management system is introduced. The e-services and the modeling of workflow processes based on e-services are introduced in Section 3. In Section 4, we introduce the constraint-based dynamic service binding mechanism. Section 5 describes the implementation. Section 6 summarizes our research.

2. System Architecture

The architecture of the dynamic workflow management system is shown in Figure 1.

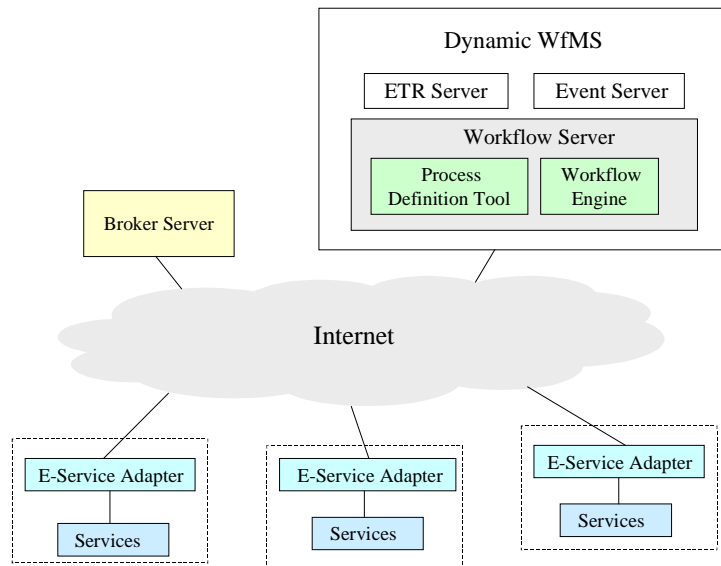


Figure 1: Architecture of the dynamic workflow management system.

Business organizations across the Internet can perform and contribute different manual or automated tasks, which are useful for the operation of a joint business. An E-Service Adapter needs to be

installed at each organization's site to wrap the underlying services, which can be implemented in different ways, as e-services. Thus, these services can be made accessible on the Internet using SOAP [Box00] and HTTP protocol. A Broker Server works as a central repository for e-services and is typically used by the service requesters to find required e-services [Hel01a]. The Workflow Server is composed of two sub-components: namely, the Process Definition Tool and the Workflow Engine. The Process Definition Tool is responsible for modeling business processes, which integrate the e-services across the Internet. The Workflow Engine schedules the enactment of business processes according to the defined process models.

The Event Server and the Event-Trigger-Rule (ETR) Server, which give the workflow management system its active and adaptive properties [Men01], are also shown in Figure 1.

3. E-Services and Process Modeling

3.1 E-Service Template

In order to introduce a standard way to define e-services, it is useful to categorize e-services and their providers by the types of business in which these providers are involved. For example, some business organizations may function as the *Distributor* of a supply chain. For each business type, a set of useful e-services can be defined. Business organizations that are of the same business type may provide all or some of these e-services. The categorization of service providers and the specification of e-services are consistent with the Universal Description, Discovery and Integration (UDDI) specification [Ari00].

To standardize the specification of an e-service, *an e-service template* can be jointly defined by those business organizations of the same business type. An e-service template consists of one or more operations offered by the e-service. For each operation, there are three types of attributes:

- Input attributes, which specify the data needed as input to invoke an operation.
- Output attributes, which specify the returned data of an operation.
- Service attributes, which specify other properties of an operation, such as the length of time the operation takes, the side-effect of the operation, etc.

An e-service template can also contain service attributes for the e-service. These attributes specify the properties of an e-service, such as the cost for using the e-service, the quality of the e-service, etc., which may be useful for negotiation, contracting, or service selection.

A simple example of an e-service template of an e-service *OrderProcessing* provided by the business type *Distributor* is shown in Table 1. It contains an operation *Process Order*.

Table 1. E-Service Template of e-service *OrderProcessing* of *Distributor*

Service	Operations	Description		
Operations	Process Order		Name	Type
		Input Attributes	productDesc quantity userInfo	ProdDesc Int UserInfo
		Output Attributes	satus	Status
		Service Attributes	duration cost	Time Float

3.2 E-Service Constraint

A service provider registers its e-services with the Broker Server by first browsing and selecting the proper e-service templates that are managed by the Broker Server. During the registration, the service provider first provides the Broker Server with its general information, such as its name, URL, telephone, email, etc. It then specifies the e-services it provides. For each e-service, the service provider needs to specify *the e-service binding description*, which contains the location of the service implementation and details on the protocol and the port to be used to access the server that hosts the e-service.

The service provider can also specify the constraints on the service attributes of the e-service, and the constraints on the input attributes and service attributes of the operations. By allowing attribute and inter-attribute constraints associated with e-service and its operations as well as e-service requests to be explicitly specified, we can extend the Web Service Description Language (WSDL) to increase its expressive power. These constraints restrict the selection of proper e-service providers for e-service requests.

For constraint specifications, we adopt the syntax and semantics of the Constraint-Based Requirement Specification Language used in [Su01]. We shall call these constraints *e-service*

constraints. For example, a distributor named *Worldwide* who provides the e-service named *OrderProcessing* may specify the following constraint on the operation *Process Order* as shown in Figure 2. This constraint specification states that the operation *Process Order* of e-service *OrderProcessing* can only process the order of computer product with the quantity less than 1000. If the quantity of the order is larger than 500, this e-service needs to take more than 10 time units. **Iac1** is the name of the inter-attribute constraint.

ATTRIBUTE_CONSTRAINT:			
productName	String	ENUMERATION ["Computer"]	priority[1]
modelName	String	ANY	priority[2]
quantity	Integer	RANGE [1-1000]	priority[3]
INTER_ATTRIBUTE_CONSTRAINT:			
Iac1	quantity > 500	implies	duration>10

Figure 2: Constraint specification for operation Process Order.

For each e-service, the e-service template, the e-service binding description, and the e-service constraints defined by the service provider together form the *e-service specification*. After registration, the general information of the service provider and the *e-service specifications* of the e-services it provides are stored in a persistent store and managed by the Broker Server.

3.3 E-Service Request and its Constraint

In our dynamic workflow management system, e-service requests are the main task items that can be specified in an activity definition. They are defined according to their corresponding e-service templates. The bindings of e-service requests to specific service providers occur at run-time when the available providers are known to the workflow management system through the Broker Server. During process modeling, the model designer first browses the e-service template information provided by the Broker Server. He/she then defines the e-service requests in the activities of a process model according to the corresponding e-service templates. In an e-service request, in addition to the values of the input attributes of the requested operation, the constraints on the service attributes of the operation and the e-service can also be specified. We shall call the constraints in an e-service request *e-service request constraints*. An example of an e-service request constraint for the operation *Process Order* of the e-

service *OrderProcessing* is shown in Figure 4. It states that the requester expects that the Process Order operation should not take more than 10 time units, the cost of the e-service should not be more than \$1,000, and if it takes more than 4 time units, then the cost must be less than \$800.

ATTRIBUTE_CONSTRAINT:			
duration	int	RANGE [0 .. 10]	priority[1]
cost	float	RANGE [0 .. 1000]	priority[2]
INTER_ATTRIBUTE_CONSTRAINT			
iac1:	duration >4	implies	cost < 800

Figure 3: A sample specification of an e-service request constraint.

In summary, at build-time, the service providers register their e-services with the Broker Server according to these e-services' corresponding templates. The e-service specifications are maintained at the Broker Server site. The process model designers define e-service requests according to the same corresponding e-service templates. The build-time relationship among the Broker Server, the service providers, and the process definition tool is shown in Figure 4.

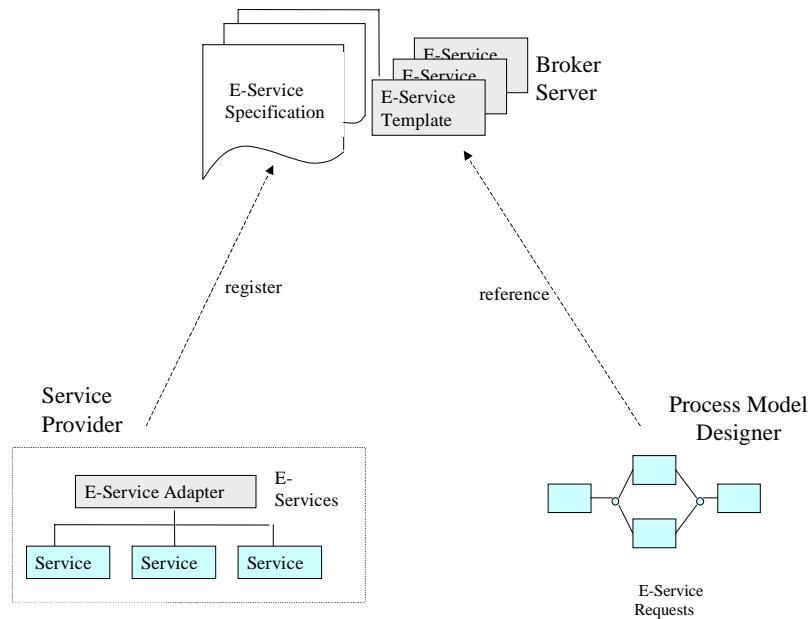


Figure 4: Build-time relationship among the Service Provider, the Broker Server, and the Process Definition Tool.

3.4 E-Service Invocation

The E-Service Adapter at each organization's site wraps services, which have been implemented in different ways, so that these services can be invoked using SOAP messages that are sent through the HTTP protocol, according to the format of the corresponding e-service templates. The E-Service Adapter thus hides the heterogeneity of service implementations and presents a uniform view of these services as *e-services*. During the e-service invocation, the E-Service Adapter parses the SOAP message from the e-service requester (in our case, the Workflow Engine) and determines the operation of the e-service to be invoked. The E-Service Adapter then determines the corresponding method of different service implementations, builds the parameter list as required by the method, and invokes the method. After the invocation is complete, the E-Service Adapter encapsulates the return data into a SOAP message and returns it to the e-service requester. The SOAP message that is used to invoke the operation *ProcessOrder* of the e-service *OderProcessing* is shown in Figure 5.

3.5 Process Modeling using E-Services

A process model consists of activities that are connected by conditional transitions, which specify the control flows. Additionally, we add data flows among activities. We shall describe only the activity definition below because activity is the main modeling construct in a process model and contains e-service requests. The syntax of the activity specification is shown below.

```
ACTIVITY <activity id>
  [DESCRIPTION <description>]
  [PERFORMER <business type name> (<performer selection constraint>)]
  IN_PARAMETER <input parameters>
  OUT_PARAMETER <output parameters>
  [WF_EVENTS] <workflow event list>
  [ACTIVITY_VAR <variable list>]
  IMPLEMENTATION
    .
    ESERVICE <e-service name>.<operation name>
      INPUT <in_attributes mapping>
      [OUTPUT <out_attributes mapping>]
      CONSTRAINT <constraint definition>
    END_ESERVICE
    .
  END_IMPLEMENTATION
END_ACTIVITY
```

```

<SOAP-ENV:Envelope
  xmlns="www.eservices.ufl.edu/services/OrderProcessing"
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">

  <SOAP-ENV:Header>
    <serviceuri>
      www.eservices.ufl.edu/services/Distributor/OrderProcessing
    </serviceuri>
  </SOAP-ENV:Header>

  <SOAP-ENV:Body>
    <!-- element holding remote call param info -->
    <ProcessOrder>
      <productDesc>
        <prodName> Omnibook </productname>
        <modelName> 6000 </model>
      </productDesc>
      <quantity> 5 </quantity>
      <userInfo>
        <username> DB Center </username>
        <maillingAddr> CSE470, University of Florida, Gainesville, FL
        </maillingAddr>
      </userInfo>
    </ProcessOrder>
  </SOAP-ENV:Body>

</SOAP-ENV:Envelope>

```

(A)

```

<SOAP-ENV:Envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/"
  SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">

  <SOAP-ENV:Header>
    <serviceuri>
      www.eservices.ufl.edu/services/Distributor/OrderProcessing
    </serviceuri>
  </SOAP-ENV:Header>

  <SOAP-ENV:Body>
    <!-- element holding operation result -->
    <ProcessPO_result>
      <OrderStatus> order shipped </OrderStatus>
    </ProcessPO_result>
  </SOAP-ENV:Body>

</SOAP-ENV:Envelope>

```

(B)

Figure 5: SOAP messages for invocation of operation *ProcessOrder* in e-service *OrderProcessing*. (A) SOAP request message. (B) SOAP response message.

The *activity id* is a unique identifier for the activity in the extent of a process model. The DESCRIPTION clause contains the description of the activity. The input/output parameters of the activity, which are defined in the IN_PARAMETER/OUT_PARAMETER clauses, specify the input/output data of the activity. The ACTIVITY_VAR clause defines the variables that can be used inside the activity body. The WF_EVENT clause specifies the events that this activity posts. The explanation of workflow events is beyond the scope of this paper.

The PERFORMER clause specifies the *business type* of the business organization whose e-services are requested in the activity. The *performer selection constraint* in the PERFORMER clause is defined to further restrict the selection of an organization as the service provider. There are four types of performer selection constraints as shown below. The last two types take into consideration the interdependencies between activities in the selection of providers for their e-service requests.

- (1) The performer is a particular organization specified by the name of the organization. An example definition is shown below. Here, *Distributor* represents the business type of organizations that provide services as distributors, and *worldwide* is the designated distributor.

PERFORMER *Distributor* (CONSTANT *worldwide*)

- (2) The performer can be any suitable organization of the specified business type. In this case, the e-service requests in the corresponding activity definition will be dynamically bound to a proper organization at run-time in a *dynamic service binding* process. An example of the PERFORMER clause is shown below.

PERFORMER *Distributor* (ANY)

- (3) The performer of the activity should be the same as that of another specified activity. An example definition is shown below. Here, *Activity1* is the name of another activity.

PERFORMER *Distributor* (SAME_AS *Activity1*)

- (4) The performer of the activity is specified by the output parameter of a specified activity. An example definition is shown below. The performer of the current activity is computed by *Activity2* and represented as the output parameter *bestDistributor*.

PERFORMER *Distributor* (VARIABLE *Activity2.bestDistributor*)

The IMPLEMENTATION clause specifies the activity body, which contains a set of task items of this activity. The e-service requests are the main task items in an activity body. An activity may contain several e-service requests, while all these e-services are to be provided by the same business organization.

The e-service request definition is shown inside the activity definition. *E-service name* is the name of the e-service to be requested, and *operation name* is the name of the operation to be invoked. The *in_attributes mapping* in the INPUT clause represents the mapping between the activity data (namely, input parameters of the activity and activity variables) and the input attributes of the e-service request. The mapping between the activity data (namely, output parameters of the activity and activity variables) and the output attributes of the e-service request is represented by the *out_attributes mapping* in the OUTPUT clause. The CONSTRAINT clause specifies e-service request constraints.

In addition to e-service requests, there can be two more kinds of task items in an activity definition: in-line code and event posting. Programming code can be included in the activity definition to do some computation. An activity can also explicitly post events. Remote systems that have subscribed to the events will receive event notifications, which report the execution milestones of a process model.

A sample activity definition, which is used to make an order from a distributor, is given in Figure 6. The only task item in this activity is an e-service request to the operation *ProcessOrder* of the e-service *OrderProcessing*. Since the performer selection constraint of this activity is ANY, the e-service request in this activity will be dynamically bound to a proper distributor during the execution of a workflow instance initiated from the process model, which includes this activity.

4. Constraint-based Dynamic Service Binding

4.1 Broker Server and Constraint-based Brokering Service

An important function of the Broker Server is to do constraint-based brokering and service provider selection. In the dynamic workflow management system called DynaFlow, this function is used by the Workflow Engine to do the dynamic service binding. To achieve this, the Broker Server would

match an e-service request with the e-service specifications given by service providers to identify the proper service provider(s) for the request. The data provided for the input attributes of the requested e-service operation and the constraints specified in the request will have to be compatible with (i.e., not conflict with) the attribute constraints and inter-attribute constraints specified by a service provider. The constraint matching is accomplished by using a Constraint Satisfaction Processor (CSP) used in the system reported in [Su01].

```

ACTIVITY Process_Order
  DESCRIPTION "Process an order from a retailer".
  IN_PARAMETERS String prod_name, String model_name, Integer quantity,
                UserInfo user_info
  OUT_PARAMETERS Boolean order_status
  PERFORMER Distributor (ANY)
  IMPLEMENTATION

    E-SERVICE OrderProcessing.ProcessOrder
      INPUT prod_name, model_name, quantity, user_info
      OUTPUT order_status
      CONSTRAINT
        ATTRIBUTE_CONSTRAINT:
          duration Integer RANGE [0 .. 10] priority[1]
          cost Float RANGE [0..1000] priority[2]

        INTER_ATTRIBUTE_CONSTRAINT:
          Iac1: duration > 4 implies cost < 800
        END_CONSTRAINT
      END_SERVICE
    END_IMPLEMENTATION
  END_ACTIVITY

```

Figure 6: A sample activity definition.

In a matching operation, there are three possible results. First, the Broker Server cannot find a service provider that can provide the e-service that satisfies the constraints and input data given in the e-service request. In this case, the matching operation has failed. Second, there is a single service provider, which provides the e-service that satisfies all the requirements of the e-service request or matches the requirements within an acceptable threshold. In this case, the matching operation succeeds and the e-service of the provider is used to service the request. In the third case, multiple service providers can

satisfy the request. A Cost-Benefit Evaluation Server [Su01] is then used to evaluate and rank the e-services provided by these service providers and the best provider is selected.

4.2. Dynamic Service Binding

The Workflow Engine accomplishes the dynamic service binding with the help of the Broker Server, which performs the constraint-based service provider selection. Before the Workflow Engine calls the Broker Server, some preprocessing work needs to be done to determine which e-services requested in the process model need to be provided by the same service provider. For example, the provider that processes a purchase order should also handle the shipping of the product and the issuing of an invoice.

In addition to the attribute constraints and/or inter-attribute constraints that are defined on the service attributes of an e-service operation during the process modeling, an e-service request constraint should also contain the input data of the requested e-service operation as attribute constraints. These values are obtained at the run-time before the e-service operation is to be invoked, and are added to the original e-service request constraint to generate a new one. A new e-service request constraint of the operation *ProcessOrder* of the e-service *OrderProcessing* is shown in Figure 7.

ATTRIBUTE_CONSTRAINT:			
duration	Integer	RANGE [0 .. 10]	priority [1]
cost	Float	RANGE [0 .. 1000]	priority [2]
productDesc.prodName	String	EQUAL "Computer"	priority [0]
productDesc.modelName	String	EQUAL "Pentium 800"	priority [0]
quantity	Integer	EQUAL 500	priority [0]
userInfo.userName	String	EQUAL "DB Center"	priority [0]
userInfo.address	String	EQUAL "University of Florida, Gainesville, Florida"	priority [0]
userInfo.zipCode	Integer	EQUAL "32611"	priority [0]
INTER_ATTRIBUTE_CONSTRAINT			
iacl: duration >4 implies cost < 800			

Figure 7: A new e-service constraint by adding the input data of the e-service operation.

The new e-service request constraint, along with the information about e-services that need to be provided by the same service provider, is given to the Broker Server to select the proper service provider. The Broker Server first gets the service providers that provide all the e-services requested from the same

in a constraint-based dynamic service binding process. The implementation of the constraint-based Broker Server is being carried out by extending the Constraint Satisfaction Processor and the Cost-benefit Evaluation Server used in an automated negotiation system [Su00a, Su01]. To invoke an e-service, the Workflow Engine generates a SOAP message, which contains the e-service request information and send the message to the E-service Adapter of the selected service provider. The interactions among the Process Definition Tool, the Workflow Engine, the Broker Server, and E-Service Adapter are shown in Figure 9.

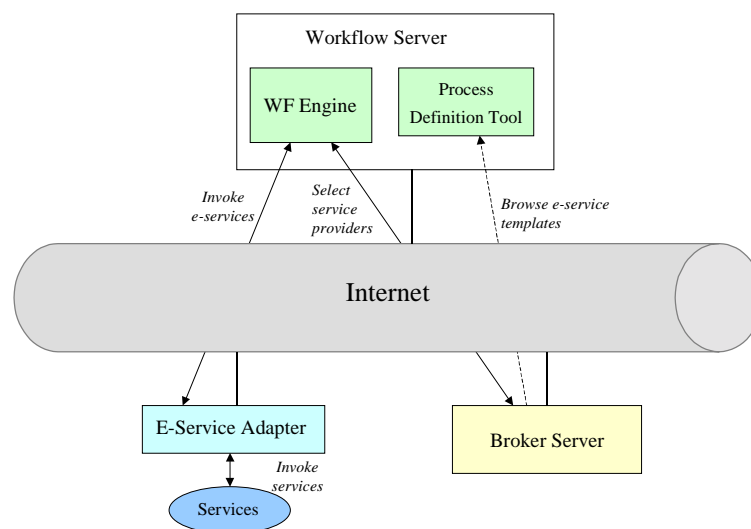


Figure 9: Interactions among Workflow Server, Broker Server and E-Service Adapters

6. Conclusion

This paper presents a solution to achieve flexible inter-enterprise workflow management in an e-service infrastructure. In this infrastructure, providers of e-services register their services with a Broker Server based on standard templates pre-defined for different business types. The service requests specified in the activities of a workflow process model can be dynamically bound to these distributed e-services and their providers with the help of the Broker Server. E-service requests posed in XML SOAP messages can be sent to the identified provider sites through the HTTP protocol. We have extended the traditional workflow process modeling by including e-service requests in activity specifications and

extended e-service specification proposed by WSDL by including constraint specifications so that the selection of e-service providers can be more accurately performed. The constraint-based dynamic e-service binding mechanism presented in this paper allows inter-enterprise workflow process models to be processed in the Internet environment, in which business organizations and their services are changing constantly. The interdependencies among activities of a process model with respect to provider (or performer) selection are also considered.

References

- [Alo99] Alonso, G., Fiedler, U., Hagen, C., Lazcano, A., Schuldt, H., and Weiler, N. "WISE – Business to Business E-Commerce," Proceedings of the 9th International Workshop on Research Issues on Data Engineering: Information Technology for Virtual Enterprises, Sydney, Australia, March 1999.
- [Ari00] Ariba Corporation, IBM Corporation, and Microsoft Corporation, "UDDI Technical White Paper," September 2000, <http://www.uddi.org>.
- [Box00] Box, D., Ehnebuske, D., Kakivaya, G., Layman, A., Mendelsohn, Nielsen, H.F., Thatte, S., and Winer, D., "Simple Object Access Protocol (SOAP) 1.1," May 2000, <http://www.w3.org/TR/SOAP>.
- [Chr01] Christensen, E., Curbera, F., Meredith, G., and Weerawarana, S., "Web Services Description Language (WSDL) 1.1" W3C Note, March 2001, <http://www.w3.org/TR/WSDL.html>.
- [Gre99] Grefen, P. and Hoffner, Y., "CrossFlow - Cross-Organizational Workflow Support for Virtual Organizations," Proceedings of 9th International Workshop on Research Issues on Data Engineering: Information Technology for Virtual Enterprises (RIDE-VE'99), Sydney, Australia, March 1999.
- [Hel02] Helal, A., Su, S.Y.W., Meng, J., Krithivasan, R., and Jagatheesam, A., "The Internet Enterprise," to appear in the Proceedings of the 2nd IEEE/JPSJ Symposium on Applications and the Internet (SAINT02), Nara, Japan, January 2002.
- [Hel01] Helal, A., Wang, M., Jagatheesan, A. and Krithivasan, R., "Brokering Based SelfOrganizing E-Service Communities," Proceedings of the Fifth International Symposium on Autonomous Decentralized Systems (ISADS) With an Emphasis on Electronic Commerce, Dallas, Texas, USA, March 2001.
- [Lam98] Lam, H. and Su, S.Y.W., "Component Interoperability in a Virtual Enterprise Using Events/Triggers/Rules," Proc. of OOPSLA '98 Workshop on Objects, Components, and Virtual Enterprise, Vancouver, BC, Canada, Oct. 18-22, 1998, pp. 47-53.
- [Laz01] Lazcano, A., Schuldt, H., Alonso, G., and Schek, H., "WISE: Process based E-Commerce," IEEE Data Engineering Bulletin, Special Issue on Infrastructure for Advanced E-Services, Vol. 24, No. 1, March 2001, pp. 46-51.
- [Lee01] Lee, M., Su, S.Y.W., and Lam, H., "A Web-based Knowledge Network for Supporting Emerging Internet Applications," WWW Journal, Vol. 4, No. 1/2, 2001, pp. 121-140.
- [Men02] Meng, J., Su, S.Y.W., Lam, H., and Helal, A., "Achieving Dynamic Inter-Organizational Workflow Management by Integrating Business Processes, Events, and Rules," to appear in the Proceedings of the 35th Hawaii International Conference on System Sciences (HICSS35), Hawaii, USA, January 2002.

- [She99] Sheth, A., Aalst, and W., Arpinar, I., "Process Driving the Networked Economy," *IEEE Concurrency*, Vol. 7, No. 3, July-September 1999, pp. 18-31.
- [Str00] Stricker, C., Riboni, S., Kradolfer, M., and Taylor, J., "Market-Based Workflow Management for Supply Chains of Services," *Proceedings of the 33rd Hawaii International Conference on System Sciences*, Hawaii, USA, 2000.
- [Su00a] Su, S.Y.W., Huang, C., and Hammer, J., "A Replicable Web-based Negotiation Server for E-commerce," *Proceedings of the Hawaii International Conference on System Sciences (HICSS-33)*, Minitrack on "Evolution of Business-to-Business Electronic Commerce," Maui, Hawaii, January 4-7, 2000, p. 219, abstract only; full 9-page paper is included in the conference CD.
- [Su00b] Su, S.Y.W. and Lam, H., "IKnet: Scalable Infrastructure for Achieving Internet-based Knowledge Network," invited paper, *Proceedings of the International Conference on Advances in Infrastructure for Electronic Business, Science, and Education on the Internet*, l'Aquila, Rome, Italy, July 31-Aug.6, 2000.
- [Su01] Su, S.Y.W., Huang, C., Hammer, J., Huang, Y., Li, H., Wang, L., Liu, Y., Pluempitiwiriyawej, C., Lee, M., and Lam, H., "An Internet-based Negotiation Server for E-Commerce," *VLDB Journal*, Vol. 10, No. 1, August 2001, pp. 72-90.
- [WfM99] WfMC, "Interface1: Process Definition Interchange, V 1.1 Final (WfMC-TC-1016-P)," 1999, <http://www.wfmc.org>.
- [Xia01] Xian, J., "Graphical User Interface for Dynamic Workflow Management," Masters Thesis, Department of Electrical and Computer Engineering, University of Florida, 2001.