Electronic Commerce Research, 3: 9–24 (2003)

Dynamic Inter-Enterprise Workflow Management in a Constraint-Based E-Service Infrastructure

STANLEY Y.W. SU *, JIE MENG, RAJA KRITHIVASAN, SEEMA DEGWEKAR and SUMI HELAL
{su, jmeng, rkrithiv, spd, helal}@cise.ufl.edu
Database Systems R&D Center, University of Florida, Gainesville, FL 32611-6125, USA

Abstract

This paper presents an infrastructure and a mechanism for achieving dynamic Inter-enterprise workflow management using 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. In this work, we categorize e-services according to their business types and manage them in a UDDI-enabled constraint-based Broker Server. 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, which take part in a business process, are not determined until run-time. We have extended the traditional workflow process modeling by including e-service requests in activity specifications and extended the Web Service Description Language (WSDL) by including constraints in both service specifications and service requests so that the selection of e-service providers can be more accurately performed.

Keywords: e-service, e-service constraint, e-service request constraint, constraint-based brokering, dynamic service binding

0. 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 [Alonso, 1; Grefen, 5; Lazcano, 9; Sheth, 14; Stricker, 15]; 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

* Corresponding author.

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 implemented a dynamic workflow management system to support e-businesses [Helal, 7; Meng, 12]. The system is active, adaptive, and flexible. By integrating the system with an Event-trigger-rule Server and an Event Server [Lam, 8; Lee, 10; Su, 19], 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 Constraint-based Broker Server. Business organizations register the e-services they provide with the Broker Server according to the e-service templates, and the resulting service specifications are maintained 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 binding 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 [Christensen, 4]. These constraints are used by a Constraint-based 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.

1. System architecture

The architecture of the dynamic workflow management system is shown in Figure 1. Business organizations across the Internet can perform and contribute different manual

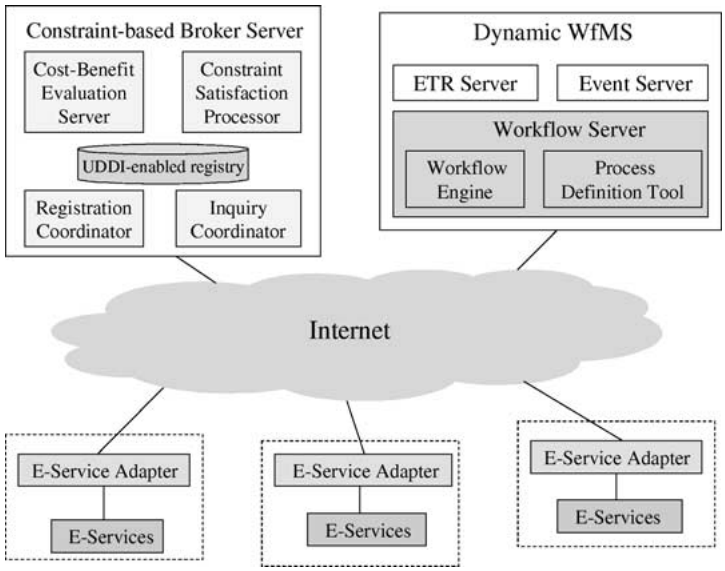

Figure 1. Architecture of the dynamic workflow management system.

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 [Box, 3] and HTTP protocol. A Constraint-based Broker Server works as a central repository for e-services and is used by the service requesters to find required e-services [Helal, 6]. 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 system its active and adaptive properties [Meng, 12], are also shown in Figure 1.

The Constraint-based Broker Server consists of the following components—a Registration Coordinator, an Inquiry Coordinator, a Constraint Satisfaction Processor [Su, 17], a Cost-Benefit Evaluation Server [Su, 17] and a registry. When a service provider registers with the Broker, the Registration Coordinator stores the information in the registry. On receiving an e-service request, the Inquiry Coordinator queries the registry to get a list of suitable service providers. This list is then pruned by invoking a Constraint Satisfaction Processor and a Cost-Benefit Evaluation Server to perform constraint matching and cost-benefit evaluation and selection of service providers, respectively.

2. E-services and process modeling

2.1. E-service template

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 [Ariba Corporation, 2].

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. All e-service templates are managed by the Constraint-based Broker Server. 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 are used to specify the data needed as input to invoke an operation. Output attributes are used to specify the returned data of an operation. Service attributes are used to 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 itself 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 *ProcessOrder*.

2.2. E-service constraint

A service provider registers its e-services with the Constraint-based Broker Server by first browsing and selecting the proper e-service templates that are managed by the Broker. During the registration, the service provider first provides the Broker 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

Table 1. E-Service template of e-service *OrderProcessing* of distributor.

E-Service	Operation	Description		
OrderProcessing	*ProcessOrder*		Name	Type
		Input attributes	*productDesc*	ProdDesc
			quantity	Integer
			userInfo	UserInfo
		Output attributes	*orderStatus*	Status
		Service attributes	*duration*	Time
			cost	Float

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 output attributes of its operations. These constraints restrict the selection of proper e-service providers for e-service requests. We shall call these constraints *e-service constraints*.

For constraint specifications, we adopt the syntax and semantics of the Constraint-Based Requirement Specification Language used in [Su, 17]. In this language, the constraint specification of an e-service is defined in terms of attribute constraints and inter-attribute constraints. Attribute constraints are used to specify the values that input, output and service attributes can have, and inter-attribute constraints are used to specify the interrelationship between the values of these attributes. For example, a distributor named Worldwide who provides the e-service named *OrderProcessing* may specify the following constraint on the operation *ProcessOrder* as shown in Figure 2.

The attribute constraints are listed underneath the key word ATTRIBUTE_CONSTRAINT. Each attribute value is of a particular type (e.g., String or Integer). An attribute value is specified as a set of possible values by a keyword RANGE or ENUMERATION. If an attribute has only one possible value, the keyword EQUAL is used. The attribute value that is marked as ANY means "don't care". The priority values rank the importance of different constraints and are used to decide the order in which constraints are checked. The inter-attribute constraints are listed below the keyword INTER_ATTRIBUTE_CONSTRAINT. The keyword "implies" connects the antecedence with the consequence of an inter-attribute constraint.

The constraint specification shown in Figure 2 states that the operation *ProcessOrder* 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.

By allowing attribute and inter-attribute constraints associated with e-service and its operations to be explicitly specified, we have extended the Web Service Description Language (WSDL) to increase its expressive power.

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

ATTRIBUTE_CONSTRAINT:

productDesc.name	**String**	EQUALS "Computer"	**priority**[1]
productDesc.model	**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 *ProcessOrder*.

specifications of the e-services it provides are stored in a persistent store and managed by the Constraint-based Broker Server.

2.3. E-service request and 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 Constraint-based Broker Server. During process modeling, the model designer first browses the e-service template information provided by the Broker. 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 *ProcessOrder* of the e-service *OrderProcessing* is shown in Figure 3. This constraint specification states that the requester expects that the *ProcessOrder* 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.

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

2.4. 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

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

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

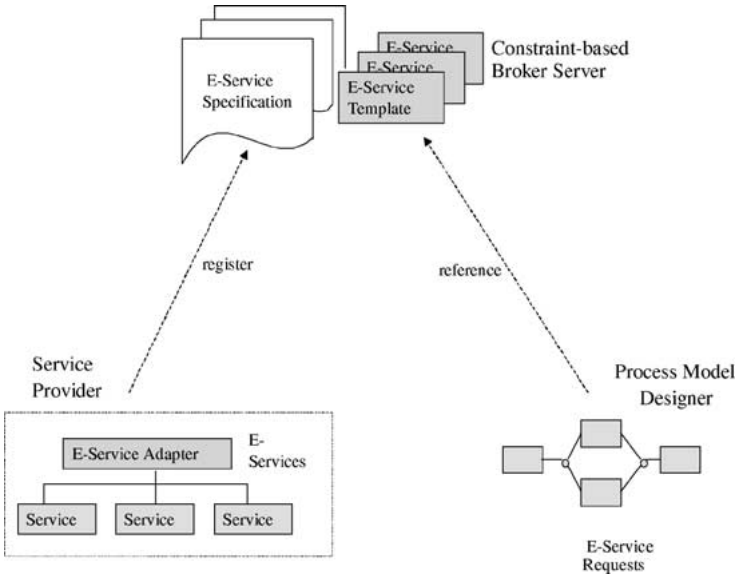

Figure 4. Build-time relationship among the Service Provider, the Constraint-based Broker Server, and the Process Model Designer.

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
        INPUT_CONSTRAINT <constraint definition>
        OUTPUT_CONSTRAINT <constraint definition>
        OPERATION_CONSTRAINT<constraint definition>
        ESERVICE_CONSTRAINT <constraint definition>
      INTER_ATTRIBUTE_CONSTRAINT<definition>
    END_ESERVICE
  . . .
  END_ IMPLEMENTATION
END_ACTIVITY
```

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. It consists of an input constraint definition, an output constraint definition, an operation constraint definition, a service constraint definition each of which contains attribute and inter-attribute constraints as mentioned earlier. But, we can also have some relationship between, say, an operation constraint and a service constraint. This is specified in the clause Inter_Attribute_Constraint.

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 5. The only task item in this activity is an e-service request to the operation

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

        ESERVICE OrderProcessing.ProcessOrder
            INPUT productDesc, 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_ESERVICE

    END_IMPLEMENTATION
```

Figure 5. A sample activity definition.

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.

3. Constraint-based dynamic service binding

3.1. Constraint-based brokering service

An important function of the Broker Server is to do constraint-based brokering and service provider selection. In our dynamic workflow management system, this function is used by the Workflow Engine to do the dynamic service binding. To achieve this, the Constraint-based 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 used in the system reported in [Su, 17].

The Inquiry Coordinator in the Constraint-based Broker Server handles requests for finding a suitable e-service. On receiving an e-service request, the Inquiry Coordinator separates the constraint definition from the rest of the request definition. This request is then given to the UDDI-enabled registry to come up with an initial list of service providers. Then, constraint matching is done between each service provider's and the e-service request's constraints.

In this matching operation, attributes constraints are evaluated first according to their priority orders. The alternatives obtained from the evaluation are then evaluated against the inter-attribute constraints of both e-service request and e-services. There are three possible results in this matching operation. First, the Constraint-based 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 [Su, 17; Liu, 11] is then used to evaluate and rank the e-services provided by these service providers and the best provider is selected. The Cost-benefit Evaluation Server is implemented based on the cost-benefit analysis and decision model (CBADM) reported in [Su, 16]. Different e-services are scored according to the preference analysis of different attributes and the aggregation function used. The e-service provider selection is based on the scores obtained from the evaluation.

```
ATTRIBUTE_CONSTRAINT:
  duration              Integer  RANGE  [0 .. 10]        priority [1]
  cost                   Float    RANGE  [0 .. 1000]     priority [2]
  productDesc.name      String   EQUAL  "Computer"       priority [0]
  productDesc.model     String   EQUAL  "Pentium 800"    priority [0]
  quantity              Integer  EQUAL  500              priority [0]
  userInfo.userName     String   EQUAL  "DB Center"      priority [0]
  userInfo.address      String   EQUAL  "470, CSE        priority [0]
                                        Univ. of Florida"
  userInfo.zipCode      String   EQUAL  "32611"          priority [0]

INTER_ATTRIBUTE_CONSTRAINT
   Iac1:    duration >4   Implies   cost < 800
```

Figure 6. A new e-service constraint by adding the input data of the e-service operation.

3.2. Dynamic service binding

The Workflow Engine accomplishes the dynamic service binding with the help of the Constraint-based 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 6.

The new e-service request constraints, 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 service provider. It then selects the proper one from them using the constraint-based service provider selection mechanism discussed above.

4. E-service invocation

The E-Service Adapter at each organization's site wraps software services that have been implemented in different ways so that they can be invoked by SOAP messages. The E-Service Adapter thus hides the heterogeneity of service implementations and presents a uniform view of these services as e-services. After a proper e-service provider has been se-

lected through a dynamic service binding process, the Workflow Engine will send a SOAP message that contains the input data of the requested e-service operation to the E-Service Adapter at the e-service provider's site to invoke the e-service. During the e-service invocation, the E-Service Adapter parses the SOAP message and determines the operation of the e-service to be invoked. The E-Service Adapter then determines the corresponding method of service implementation, 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 *OrderProcessing* is shown in Figure 7.

5. Implementation

We have implemented the dynamic workflow management system. The Process Definition Tool [Xian, 16] is a user-friendly graphical editor that can be used to specify the diagram of a business process model and the details of e-service requests. When using the Process Definition Tool to define a process model, the process model designer needs to make use of e-service templates, which are accessible from the Constraint-based Broker Server, to specify e-service requests inside activities. To facilitate the process modeling in our implementation, we store these e-service templates in a Metadata Manager implemented for another project [Lee, 10]. The screen layout of the Process Definition Tool is shown in Figure 8.

The Workflow Engine is responsible for scheduling the execution of a workflow instance based on the process model. When an activity is scheduled for execution, if there are e-service requests inside the activity and the performer selection constraint of this activity is ANY, the Workflow Engine would contact the Broker Server to select a proper service provider for the e-service requests inside the activity 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 [Su, 17, 18]. 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 Constraint-based Broker Server, and E-Service Adapter are shown in Figure 9.

The registry in the Constraint-based Broker Server is actually made of two parts. One is a publicly available UDDI-enabled registry such as the IBM test registry v2. This registry is used to store the e-service specification in a form that agrees with the UDDI specification. We are using the Web Services Toolkit—version 3.1 to talk to this registry. The actual packages that do the talking are UDDI4j and WSDL4j that are embedded in this toolkit. However, we cannot store constraint information in such a registry at present. Hence, we need a local registry to store this information. For this purpose, we are using a Persistent Object Manager (POM), which was implemented for an earlier project at

```
<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>
             <name> Computer </productname>
             <model> Pentium 800  </model>
        </productDesc>
        <quantity>   500  </quantity>
        <userInfo>
          <username> DB Center </username>
          <address> 470 CSE, Univ. of Florida   </address>
        </userInfo>
     <ProcessOrder>

    </SOAP-ENV:Body>

</SOAP-ENV:Envelope>
```

(a) SOAP request message.

```
<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 operation result -->
      <ProcessOrder_result>
           <OrderStatus>  order shipped   </OrderStatus>
      </ProcessOrder_result>

   </SOAP-ENV:Body>

</SOAP-ENV:Envelope>
```

(b) SOAP response message.

Figure 7. SOAP messages for invocation of operation *ProcessOrder* in e-service *OrderProcessing*.

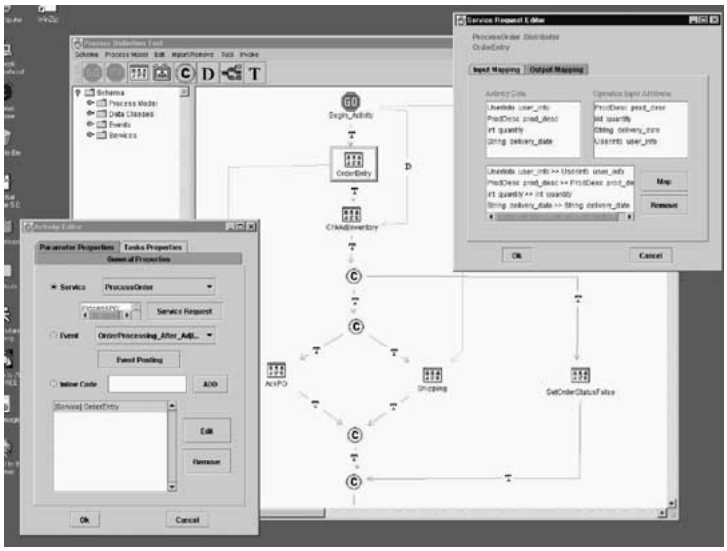

Figure 8. The screen layout of the Process Definition Tool.

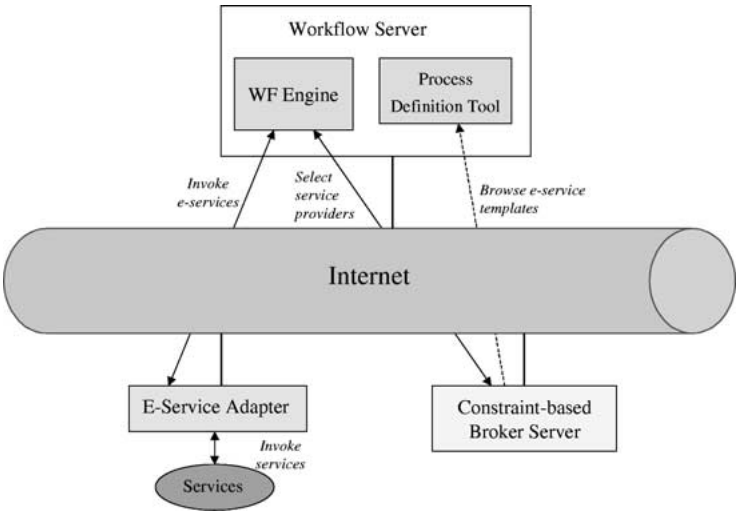

Figure 9. Interactions among Workflow Server, Constraint-based Broker Server and E-Service Adapters.

our research center [Shenoy, 13]. This POM can store the constraint information in the form of Java Objects in a persistent fashion so that they can be used later. Thus, during registration, the Registration Coordinator separates the available information to be stored into two registries. The UDDI-related information goes into the UDDI registry and the constraint information is stored in our local POM. Similarly, during an e-service request,

the Inquiry Coordinator separates the constraint-related information from the request. This UDDI request is submitted to the UDDI registry to get an initial list of satisfactory service providers. Then, constraint information is obtained from the POM to do the constraint matching.

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 Constraint-based 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 Constraint-based 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

[1] Alonso, G. et al. (1999). "WISE—Business to Business E-Commerce." In *Proceedings of the 9th International Workshop on Research Issues on Data Engineering: Information Technology for Virtual Enterprises*, Sydney, Australia, March 1999.

[2] Ariba Corporation, IBM Corporation, and Microsoft Corporation. (2000). "UDDI Technical White Paper." September, http://www.uddi.org.

[3] Box, D. et al. (2000). "Simple Object Access Protocol (SOAP) 1.1." W3C Note, http://www.w3.org/TR/SOAP.

[4] Christensen, E. et al. (2001). "Web Services Description Language 1.1." W3C Note, http://www.w3.org/TR/WSDL.html.

[5] Grefen, P. and Y. Hoffner. (1999). "CrossFlow—Cross-Organizational Workflow Support for Virtual Organizations." In *Proceedings of 9th International Workshop on Research Issues on Data Engineering: Information Technology for Virtual Enterprises (RIDE-VE'99)*, Sydney, Australia, March 1999.

[6] Helal, A. et al. (2001). "Brokering-Based Self-Organizing E-Service Communities." In *Proceedings of the Fifth International Symposium on Autonomous Decentralized Systems (ISADS) with an Emphasis on Electronic Commerce*, Dallas, Texas, USA, March 2001.

[7] Helal, A. et al. (2002). "The Internet Enterprise." In *Proceedings of the 2nd IEEE/JPSJ Symposium on Applications and the Internet*, Nara, Japan, January 2002.

[8] Lam, H. and S. Su. (1998). "Component Interoperability in a Virtual Enterprise Using Events/Triggers/Rules." In *Proceedings of OOPSLA '98 Workshop on Objects, Components, and Virtual Enterprise*, Vancouver, BC, Canada, October 18–22, 1998, pp. 47–53.

[9] Lazcano, A. et al. (2001). "WISE: Process-Based E-Commerce." *IEEE Data Engineering Bulletin*, Special Issue on Infrastructure for Advanced E-Services 24(1) 46–51.

[10] Lee, M., S. Su, and H. Lam. (2001). "A Web-Based Knowledge Network for Supporting Emerging Internet Applications." *WWW Journal* 4(1–2) 121–140.

[11] Liu, Y. et al. (2002). "A Cost-Benefit Evaluation Server for Decision Support in E-Business." To appear in *Journal of Decision Support Systems and Electronic Commerce*.

[12] Meng, J. et al. (2002). "Achieving Dynamic Inter-Organizational Workflow Management by Integrating Business Processes, Events, and Rules." In *Proceedings of the 35th Hawaii International Conference on System Sciences*, Hawaii, USA, January 2002.

[13] Shenoy, A. (2001). "A Persistent Object Manager for Java Applications." M.S. Thesis, Department of Computer and Information Science and Engineering, University of Florida.

[14] Sheth, A., W. Aalst, and I. Arpinar. (1999). "Process Driving the Networked Economy." *IEEE Concurrency* 7(3) 18–31.

[15] Stricker, C. et al. (2000). "Market-Based Workflow Management for Supply Chains of Services." In *Proceedings of the 33rd Hawaii International Conference on System Sciences*, Hawaii, USA, 2000.

[16] Su, S. et al. (1987). "A Cost-Benefit Decision Model: Analysis, Comparison, and Selection of Database Management Systems." *ACM Transactions on Database Systems* 12(3) 472–520.

[17] Su, S. et al. (2001). "An Internet-Based Negotiation Server for E-Commerce." *VLDB Journal* 10(1) 72–90.

[18] Su, S., C. Huang, and J. Hammer. (2000). "A Replicable Web-Based Negotiation Server for E-Commerce." In *Proceedings of the Hawaii International Conference on System Sciences, 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.

[19] Su, S. and H. Lam. (2000). "IKnet: Scalable Infrastructure for Achieving Internet-Based Knowledge Network." In *Proceedings of the International Conference on Advances in Infrastructure for Electronic Business, Science, and Education on the Internet*, l'Aquila, Rome, Italy, July 31–August 6, 2000. Invited paper.

[20] Xian, J. (2001). "Graphical User Interface for Dynamic Workflow Management." Masters Thesis, Department of Electrical and Computer Engineering, University of Florida.