DynaFlow: a dynamic inter-organisational workflow management system

Jie Meng,* Stanley Y.W. Su, Herman Lam, Abdelsalam Helal, Jingqi Xian, Xiaoli Liu and Seokwon Yang

Database Systems R&D Center, CSE 470,
Department of Computer and Information Science and Engineering,
University of Florida,
Gainesville, FL 32611-6125, USA
E-mail: jmeng@google.com E-mail: su@cise.ufl.edu
E-mail: hlam@ufl.edu E-mail: helal@cise.ufl.edu
E-mail: mxian@adobe.com E-mail: xiaoliliu01@yahoo.com
E-mail: seyang@cise.ufl.edu
*Corresponding author

Abstract: As the global marketplace becomes more and more competitive, business organisations often need to team up and operate as a virtual enterprise to utilise the best of their resources for achieving their common business goals. As the business environment of a virtual enterprise is highly dynamic, it is necessary to develop a workflow management technology that is capable of handling dynamic workflows across enterprise boundaries. This paper describes a Dynamic Workflow Model (DWM) and a dynamic workflow management system (DynaFlow) for modelling and controlling the execution of inter-organisational business processes. DWM enables the specification of dynamic properties associated with a business process model. It extends the underlying model of the WfMC's WPDL by adding connectors, events, triggers and rules as its modelling constructs. It also encapsulates activity definitions and allows web service (or e-service) requests to be included as a part of the activity specification. Using DWM as the underlying model, DynaFlow makes use of an Event-Trigger-Rule (ETR) server to trigger rules during the enactment of a workflow process to enforce business rules and policies and/or to modify the process model at run-time. A constraint-based, dynamic service binding mechanism is used to dynamically bind web service requests to web services that satisfy the requirements of the requests.

Keywords: Dynamic Workflow Model (DWM); dynamic workflow management system; web services; e-services; constraint specification; business event; business rule; dynamic service binding.

Reference to this paper should be made as follows: Meng, J., Su, S.Y.W., Lam, H., Helal, A., Xian, J., Liu, X. and Yang, S. (2006) 'DynaFlow: a dynamic inter-organisational workflow management system', *Int. J. Business Process Integration and Management*, Vol. 1, No. 2, pp.101–115.

Biographical notes: Jie Meng received her BA and MS in Computer Science from the Tsinghua University, Beijing, China in 1991 and 1995, respectively. She received her PhD in Computer Engineering at the University of Florida in May 2002. Her research interests include workflow systems, database systems, web services technology and electronic commerce/business systems. Currently, she is working for Google, Inc.

Stanley Y.W. Su received his MS and PhD in Computer Science from the University of Wisconsin, Madison, in 1965 and 1968, respectively. He is Distinguished Professor of the Department of Computer and Information Science and Engineering and the Director of the Database Systems R&D Center, University of Florida. His current research interests include e-business, e-learning, e-government, knowledge base management and collaborative systems. He is an IEEE Fellow.

Herman Lam is an Associate Professor of Electrical and Computer Engineering and is a member of the Database Systems Research and Development Center at the University of Florida. He has over 20 years of research and development experience in the areas of database management, object-oriented systems and knowledge base management systems. Currently, his work is focused on web/grid services, distributed objects and inter-operability of web/grid services and distributed objects through Event-Trigger-Rule (ETR) processing technology. He has authored or co-authored over 70 -refereed journal and conference papers and one textbook.

Abdelsalam Helal received his BSc and MSc in Computer Science and Automatic Control from Alexandria University, Alexandria, Egypt, in 1982 and 1985, respectively and PhD in Computer Sciences from Purdue University, West Lafayette, Indiana, in 1991.

He is a Professor in the Computer and Information Science and Engineering Department at the University of Florida. His broad research interests span the areas of database systems, mobile and wireless computing, collaborative computing and internet applications. He is a senior member of IEEE.

Jingqi Xian joined the University of Florida in August 1999 to pursue a master's degree in Electrical and Computer Engineering and received her Master's degree in Electrical and Computer Engineering in 2001. She worked as a Research Assistant during her Master's degree at the Database Systems Research and Development Center.

Xiaoli Liu joined the Department of Computer and Information Science and Engineering in August 1999 to pursue an MS degree in Computer Science and her Master degree in Computer Science, in fall, 2001. She conducted her thesis research in the Database Systems Research and Development Center.

Seokwon Yang received his bachelor's degree in Computer Science in the Computer Science and Engineering Department at the Hanyang University, Korea, in February 1997 and his MS degree in Computer Science at the University of Florida, Gainesville, in August 1999. He completed his PhD in Computer Engineering at the University of Florida in December 2003. His research interests include active, object-oriented databases, distributed systems, web services technology, e-commerce and internet security.

1 Introduction

The advent of the internet, World Wide Web and distributed computing technologies has enabled business organisations to conduct business electronically. According to a white paper from Morgan Stanley Dean Witter (Phillips and Meeker, 2000), there have been three major phases in the evolution of e-business technology. E-business first appeared in the form of Electronic Data Interchange (EDI) (Adam et al., 1998). The second phase of e-business was Basic E-commerce, where retailers sell their products through their websites. Phase three of e-business, currently in progress, is in the form of eMarketPlace 'portals', which bring trading partners with related interests together into a common community (i.e. an exchange) to match buyers and sellers and to provide other services to serve their interests. The current trend is towards collaborative e-business, in which business organisations form virtual alliances under rapidly changing market conditions and make use of the best of their resources for achieving their common business goals. Business processes and application systems of the participating organisations are important resources that need to be used to conduct a joint business. Workflow technology is an enabling technology for managing, coordinating and controlling the activities of a virtual enterprise. To coordinate the organisations of a virtual enterprise in a highly dynamic business environment, a workflow management system has to be able to handle workflows that are subject to changes and are distributed across organisation boundaries.

To meet the above requirements, we have developed a Dynamic Workflow Model (DWM) to enable the modelling of dynamic properties (i.e. active, customisable, flexible and adaptive properties) in a business process model. DWM is an extension of the underlying model of the Workflow Management Coalition's (WfMC) Workflow Process Definition Language (WPDL) (WfMC, 1999) by adding events, triggers and rules to WPDL's modelling constructs. By doing so, we integrate distributed business events and distributed business rules with distributed business processes. The enactment of a business process can post events to trigger the processing of business rules, which may in turn enact other business processes (i.e. active property). The processing of business rules may also enforce customised business constraints and policies (i.e. customisable property) and/or dynamically changes the process model at run-time (adaptive property). We also separate control information (i.e. Split and Join) from activity definitions and encapsulate activity definitions so that the model can be easily modified. In this work, we treat all the sharable tasks performed by people or automated systems in a virtual enterprise as web services. The web services are categorised according to different business types, for which standardised web service templates are defined. Service providers register their web services with a broker by using the templates. To support inter-organisational business process, web service requests are specified in the activity definitions of a process model based on the web service templates (Meng et al., 2002; Su et al., 2003). Constraint definitions are introduced in both web service specifications and web service requests. Web service requests are bound to the suitable service providers at run-time by using the constraint-based brokering services of a broker to identify the suitable providers (i.e. flexible property). By using the above dynamic binding approach, process models are not bound to specific service providers when they are defined. Changes in the membership of a virtual enterprise (i.e. its service providers and their services) will not affect the specifications and processing of process models because the virtual enterprise has multiple choices in finding suppliers and business partners. The integration of business processes with business events and rules and the dynamic binding of web service to service providers give the workflow management system its dynamic properties.

We have designed and implemented a dynamic workflow management system DynaFlow, which is part of an information infrastructure being developed for supporting an Internet-based Scalable E-business Enterprise (ISEE): the ISEE infrastructure. An early

version of the system was reported in a short conference paper (Meng et al., 2002). This paper presents the implemented version of the system in detail: the implementation of the key component of the system, the Workflow Engine and the technique for run-time modifications of a process model to dynamically alter the processing of its workflow instances.

This paper is organised as follows: In Section 2, research related to dynamic workflow and virtual enterprise is surveyed. In Section 3, the architecture of DynaFlow is introduced. The web service and the DWM are described in Section 4. Section 5 describes the design and the implementation of DynaFlow. Finally, Section 6 summarises the results of this research.

2 Related work

Workflow management has been widely accepted as a core technology to support business processes. Hundreds of commercial workflow systems exist on the market (Sheth et al., 1999). However, most of them are limited to enterprise-wide workflow and do not provide sufficient support for the dynamic properties addressed in this paper.

As a well-accepted standard in workflow area, the WPDL (WfMC, 1999), which was developed by the WfMC in 1993, provides a textual grammar for the specification of process definitions. Its underlying metamodel provides a set of modelling constructs for defining business processes. For example, a business process can be defined in terms of a set of inter-related activities connected by transitions. Each activity specifies the manual and/or automated tasks that involve workflow applications and participants. Transitions that connect these activities determine the control flow of a workflow process. Because WPDL was designed mainly for modelling business processes within an organisation boundary, it lacks the constructs for capturing the dynamic properties of inter-organisational business processes.

A very important extension we made to WPDL is to add events, triggers and rules as constructs for modelling inter-organisational business processes. Events and rules have been used in several workflow projects. In some projects, events and rules are used to define and enact the control flow in a business process model. A good example in this category is the EvE project (Geppert and Tombros, 1998). A distributed Event-Condition-Action (ECA) rule-based enactment architecture was investigated in the EvE. The use of events and rules to handle exceptions and failures during workflow execution falls into another category. In this category, events are produced outside a workflow execution, by either system environment or customers. Corresponding rules will be triggered when these events occur. An example is the WIDE project (Ceri et al., 1997). It uses a distributed architecture for workflow management, based on a database management system with rule processing capabilities. WIDE's model specification allows the definition of ECA rules to support exception handling and implement asynchronous behaviours during workflow enactment. Recent research in AgentWork (Müller, 2002) also introduced event and rule concepts in their work to enable dynamic adaptation of workflow execution to unexpected failures. Different from the ECA rules used in these projects, events, rules and triggers in this work are integral constructs in the model to capture the dynamic properties of business processes. The model designer specifies at design time when (before or after an activity) and what (synchronous or asynchronous) events will be posted at different stages of executing a process model. During the enactment of a business process by the workflow engine, the events are posted at the specified time and manner to trigger business rules to adapt to a changing business environment or to notify interested organisations or their application systems of the processing milestones of an enacted business process and/or exception conditions. Another main difference between the ECA paradigm and this Event-Trigger-Rule (ETR) paradigm is that the latter is used in an inter-organisational, distributed environment, in which events, condition-action rules and triggers can be defined and managed by separate organisations (Lee, 2000).

A dynamic workflow management system needs to be able to modify a process model at run-time to adapt to dynamic business conditions and exception situations. There are few research efforts in this area. The work reported by Reichert and Dadam (1998) presents a formal foundation for supporting dynamic structural changes of running workflow instances. The work reported by Muller and Rahm (1999) describes a rule-based approach for the detection of semantic exceptions and for dynamic workflow modifications, with a focus on medical workflow scenarios. The work in the TAM project (Zhou et al., 1998) presents a dynamic restructuring of distributed transactional activities. These works mainly focus on the structural changes of process models. In this work, DynaFlow supports both structural (e.g. drop an activity or bypass some activities) and semantic (e.g. replace an activity or modify a transitional condition) changes to an inter-organisational business process model. The 'Code Generation' approach, which is used to develop the workflow engine in DynaFlow and will be explained in detail in Section 5, makes it easy to support these changes. The adaptiveness a workflow instance is enhanced when these changes to the process model are performed dynamically by the business rules that are triggered by synchronous events posted by the running business process.

Recently, the use of the workflow technology to manage e-businesses and virtual enterprises has drawn much attention in the academic community. The authors of Dayal et al. (2001) systematically discussed the trends and open issues in the business process coordination area. Emerging standards and technologies, such as XML, SOAP (Box et al., 2000), WSDL (Christensen et al., 2001), UDDI (Ariba Corporation, IBM Corporation and Microsoft Corporation, 2000), WSFL (Laymann, 2001) and BPEL (Business Process Execution Language for Web Services, 2002) further escalated the interest in inter-organisational workflow. Workflow-based Internet

SErvices (WISE) aims to design, build and test a commercially viable infrastructure for developing distributed applications over the internet (Alonso et al., 1999; Lazcano et al., 2000). CrossFlow provides a mechanism for connecting WfMS and other WfMS-like systems in cross-organisational workflows and electronic commerce settings. It defines a service-oriented model for cross-organisational workflows (CrossFlow Consortium, 2000; Grefen and Hoffner, 1999). A Collaborative Process Framework (CPM) has been developed in HP laboratory to support decentralised, peer-to-peer process management for inter-enterprise collaboration at the business process level (Chen and Hsu, 2001). SELF-SERV, from the University of New South Wales in Australia, is a web services composition platform in a peer-to-peer environment (Sheng et al., 2002). In this work, a composite web service is composed of individual web services, which are linked together by a state diagram. ECA rules are used to define the transitions between the composite web services. Preconditions are used to determine when a service should be executed. Post-processings are used to determine what should be done after service execution.

It is clear that a common business process model and a mechanism to publish/discover/use the resources and services that participating organisations can provide are very important for achieving resource sharing and collaboration in a virtual enterprise. This work is unique in that:

1 unlike SELF-SERV, which uses ECA rules to define transitions between composite services, rules are used in this work to provide business process customisation and run-time modifications of business process models

2 we introduce a constraint-based dynamic service binding mechanism, which adds constraint/requirement specifications in both service descriptions and service requests, and enables a constraint-based service broker (or registry) to find services and their providers that satisfy the requirements of service requests.

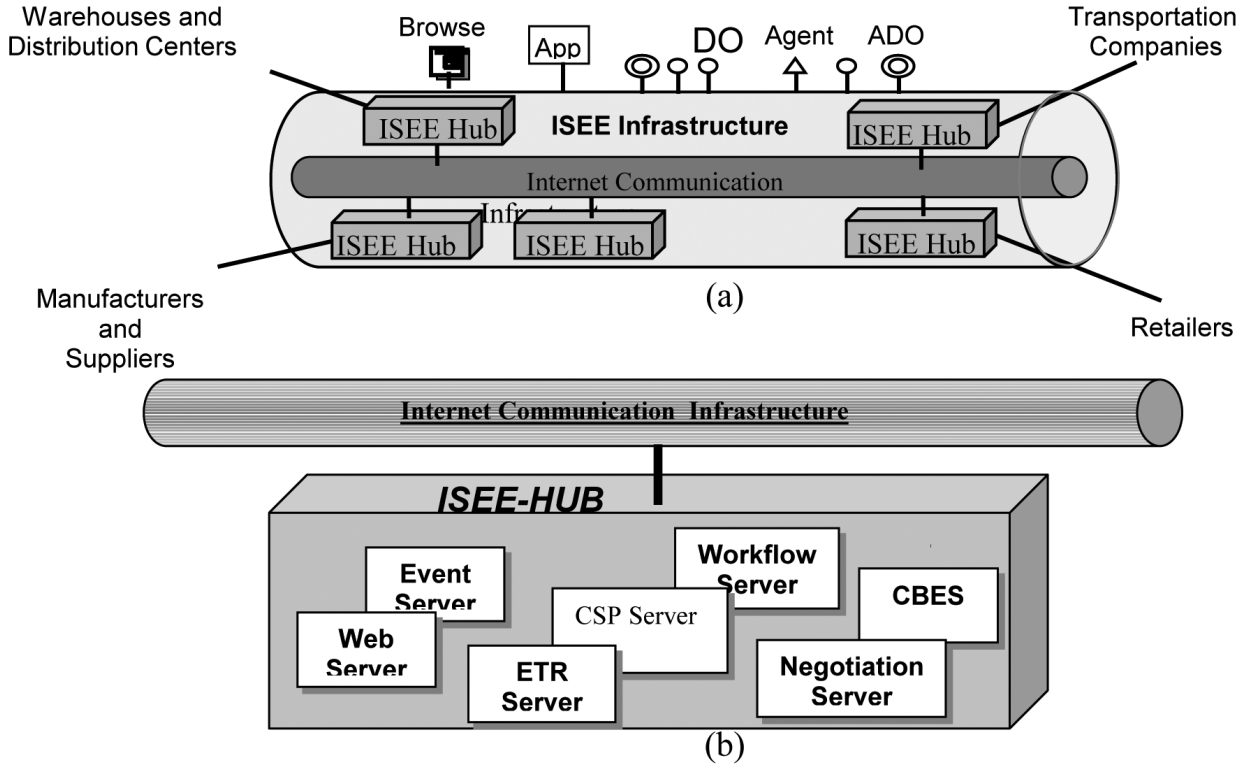
3 Architecture of DynaFlow

3.1 Overview of ISEE infrastructure

A research project, entitled "Research on Advanced Technologies to Support ISEE" and supported by the National Science Foundation, is conducted at the Database Systems Research and Development Center of the University of Florida. It aims to build an advanced information infrastructure to support collaborative e-business and other distributed applications (Su et al., 2001). The ISEE infrastructure is formed by a network of ISEE Hubs, as shown in Figure 1(a), each of which has a number of replicable ISEE servers providing various services to individuals and business companies. Its relationship with the existing application systems, distributed objects, agents, active distributed objects (Lee, 2000; Su et al., 2001) and web browsers, is illustrated in Figure 1(a).

Each ISEE Hub has a Web Server, an Event Server, an ETR Server, a Dynamic Workflow Server, an automated Negotiation Server, a Constraint Satisfaction Processing (CSP) Server and a Cost-benefit Evaluation Server as shown in Figure 1(b). These servers are middleware that provide ISEE services useful for collaborative e-business applications. ISEE Hubs are installed at the sites of ISEE

Figure 1 (a) Overall architecture and (b) ISEE infrastructure



organisations. The users of an ISEE Hub not only have full access to internet and web services, but also to additional services provided by ISEE servers. DynaFlow is built based on the services of some of these servers. For further details concerning the services of an ISEE Hub, please refer to Degwekar et al. (2004), Lee et al. (2004) and Liu et al. (2003).

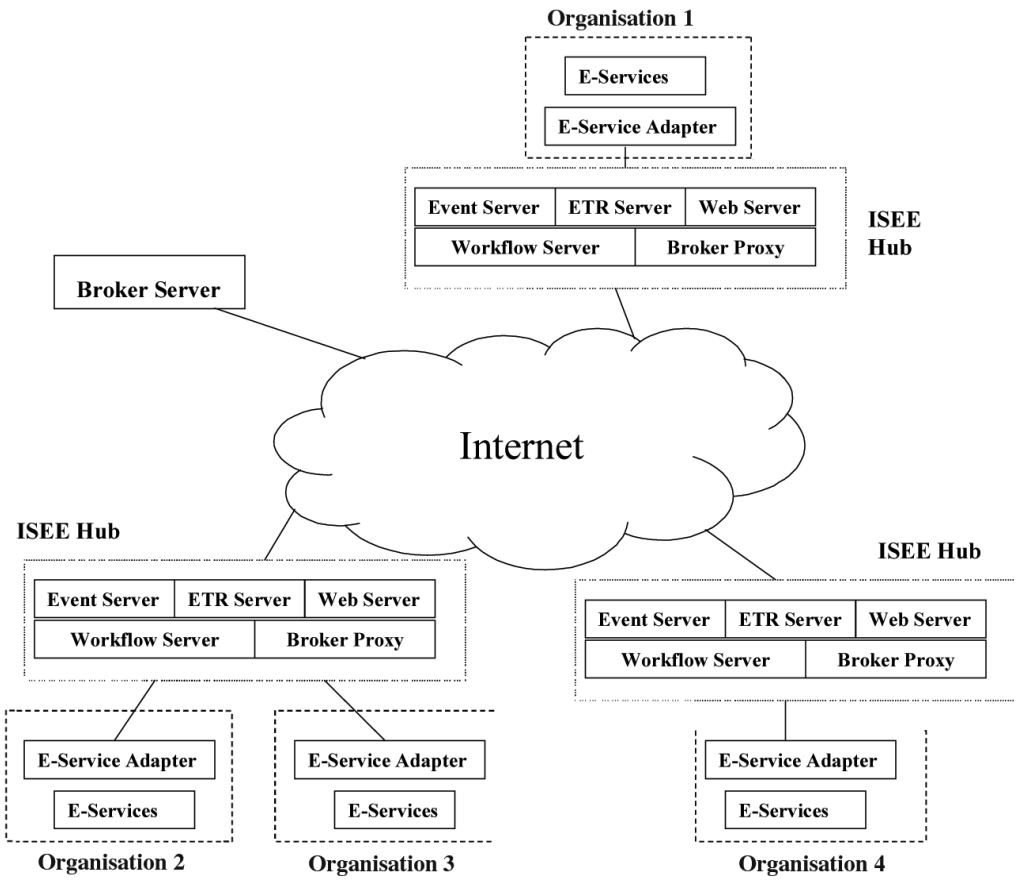
3.2 Architecture of DynaFlow

The architecture of DynaFlow is shown in Figure 2. It is a part of an ISEE hub and consists of a Workflow Server, an Event Server and an ETR Server. A Broker Server is also used in the workflow management system to manage the web service specifications of an e-business domain that have been registered by participating business organisations and to match web service requests against these specifications for selecting the suitable web service providers. A Broker Proxy is installed in each ISEE Hub to communicate with the centralised Broker Server. Multiple Broker Servers can be installed to handle multiple e-business domains.

The Workflow Server is the key component of DynaFlow. It is composed of two subcomponents: namely, the Process Definition Tool and the Workflow Engine. The former is used to design inter-organisational process models, which are modelled by using the DWM. The latter schedules the execution of the workflow instances according to the process model specification. During the execution, the Workflow Engine makes use of the ISEE services provided by the Event Server, the ETR Server and the Broker Server to achieve the dynamic properties (i.e. the active, flexible, customisable and adaptive properties) of the workflow management system. A web service adapter(s) is installed at each organisation's site to wrap the manual or automated tasks that the organisation provides as web services so that they can be invoked in the execution of an inter-organisational business process.

In an ISEE virtual enterprise, inter-organisational process models are designed to capture the business processes of a virtual enterprise. The participating organisations of the virtual enterprise may have their own process models that are published as web services and processed by their own workflow systems. These 'local' process models can be enacted by DynaFlow as a part of an inter-organisational process model. An inter-organisational process model, once designed, is made accessible to all the participating organisations. The model can be stored in a central repository and processed by a centralised workflow management system. However, for scalability reasons, we replicate all inter-organisational process models, as well as the Workflow Server and its supporting servers, at all the ISEE-Hub sites. Authorised users of the virtual enterprise can enact a process model at any site. The workflow instance created by the enactment will then be managed by an instance of the Workflow Server at that site. Thus, concurrent workflow instances initiated at different sites are controlled and managed by multiple instances of the Workflow Servers.

Figure 2 Architecture of the dynamic workflow system



4 Dynamic workflow model

4.1 Web services

Because inter-organisational business process models defined in DWM may invoke manual and automated tasks that can be carried out by the web services of different organisations, we shall first present a uniform way of specifying these tasks as web services. Web services are services offered on the internet that can be accessed programmatically using a standard internet protocol and representation format, such as HTTP and XML, irrespective of how they are implemented (Helal et al., 2002). In the internet environment, providers of web services may change frequently; new providers may be added and old providers may become unavailable. It is therefore important to separate web service requests specified in a process model from their providers. That is, a process model should not statically bind its web service requests to specific providers at the time a process model is defined, except for rare and special cases. Instead, the binding should occur at run-time when the available providers are known to the workflow management system (Meng et al., 2002; Su et al., 2003).

We categorise web services and their providers by the types of business that these providers conduct. For example, a business organisation can be of business type Distributor in a supply chain. For each business type, a set of useful web services can be defined. Business organisations that are of the same business type may provide all or some of these web services. To standardise the specification of a web service, a web service template can be jointly defined by those business organisations of the same business type. Each web service template consists of one or more operations offered by the web service. Each web service can have a number of attributes to characterise it (e.g. attributes related to the cost and quality of the service). Each service operation can also has attributes. There are three general types of attributes associated with a service operation: input attributes, which specify the data needed as input to invoke the operation; output attributes, which specify the returned data of the operation and operation attributes, which specify some properties of the operation, such as the time required to perform the operation, the side effects of the operation and the same.

All web service templates are managed at the Broker Server(s). A service provider registers its web services based on the attributes defined in the corresponding web service templates. For each web service, in addition to specify the web service-binding description, which contains the location of the service implementation and details about the protocol and the port to be used to access the server that hosts the web service, the service provider can also specify some attribute and inter-attribute constraints on the service attributes of the web service, and on the input, output and operation attributes of its operations. Attribute constraints specify the values that the individual attributes can have. Inter-attribute constraints specify the interrelationship between/among the values of these attributes. The input constraints restrict the kind of data that the requestor of a web service can provide when a web service operation is invoked. By allowing constraints associated with web services and service operations to be explicitly specified in a WSDL document, we have extended the WSDL to increase its expressive power. We shall call these constraints web service constraints. For constraint specifications, we adopt the syntax and semantics of the Constraint-Based Requirement Specification Language used by Su et al. (2001).

Suppose for business-type Distributor, there is a web service template named OrderProcessing, which processes orders from customers and contains an operation named Process Order. This operation has three input parameters, namely Produce_Name, Model_Name and Time, and one service parameter, namely Duration. When a Distributor named Worldwide, Inc. registers with the Broker Server as a service provider for OrderProcessing, it may specify constraints on the operation Process Order as shown in Figure 3. The constraints state that the operation Process Order of the web service OrderProcessing can only process an order for computer products having a quantity less than 1000, and if the quantity of the order is larger than 500, this web service operation will take more than ten time units (e.g. hours or days). Iac1 is the name of the inter-attribute constraint.

Figure 3 Constraint specification for *Process Order* operation

```
ATTRIBUTE_CONSTRAINT:
        Product_Name   String   ENUMERATION ["Computer"]
        Model_Name     String   ANY
        Quantity       Integer  RANGE[1-1000]

INTER_ATTRIBUTE_CONSTRAINT:
        iac1  Quantity > 500      implies      Time>10
```

In a process model, web service requests specified in an activity are defined in terms of the attributes given in the corresponding web service template. To define a web service request, in addition to the variable mappings, which map the activity variables to the input and output attributes of an operation, the constraints on the service attributes can also be specified. We shall call these constraints web service request constraints. An example constraint specification for a web service operation request is shown in Figure 4. It states that the requestor expects that the operation should not take more than ten time units, the cost of the web service should not be more than $1000, and if it takes more than four time units, then the cost must be less than $800.

Figure 4 An example constraint specification of a web service request

```
ATTRIBUTE_CONSTRAINT:
        Time           Integer      [0 .. 10]          priority[1]
        Cost           Float_       [0 .. 1000]        priority[2]

INTER_ATTRIBUTE_CONSTRAINT
        iac1   Time >4   implies    Cost < 800
```

During the execution of a business process, web service requests contained in the corresponding process model will be dynamically bound to the proper service providers by

using an important function of the Broker Server: constraint-based brokering and service provider selection. When a workflow instance needs to access a web service, the Workflow Engine will send the web service request and its constraint to the Broker Server. The Broker Server would match the web service request against the web service specifications given by service providers to identify the proper service provider(s). For detail about the constraint-based dynamic service binding mechanism, please refer to Meng et al. (2002).

4.2 Dynamic workflow model

There are some factors that limit WPDL's ability to support the dynamic inter-organisational workflow required by a virtual enterprise. In WPDL, the specifications of Join and Split constructs and their constraints (AND, OR or XOR), which define the structural relationships and constraints among activities, are embedded in the specifications of activities. Thus, any change to the relationships among activities would entail change to activity specifications, even though the activities themselves have not changed. Also in a process model defined in WPDL, all activities can make reference to all the 'workflow reference data' much the same way as global variables in programming languages. The data flows among activities are not explicitly defined. This makes the data relationships among activities unclear. Furthermore, a process model defined in WPDL is static. WPDL does not provide any mechanism to support dynamic properties of a process model, as discussed in this paper. Finally, as WPDL is defined for traditional workflows, it does not support the dynamic binding of web service requests, which is required for an inter-organisation resource sharing.

To adapt to the changing nature of e-business, we made the following extensions and modifications to the underlying workflow model of WPDL. These extensions and modifications not only make it easier to modify a process model both at the build-time and the run-time, but also enable it to support inter-organisational business processes.

1 *Introduction of connector*: we extract the specification of Join and Split constructs and their constraints (i.e. OR, AND and XOR) from the activity definition and introduce a new modelling construct called Connector to specify these aggregation properties. By separating activity specifications from the specifications of control information, any change made in one will not affect the other. The separation also facilitates the run-time modification of the process model itself (see a detailed explanation in Section 5).

2 *Encapsulation of the activity definition*: we modify WPDL's activity definition by adding the specification of input parameters and output parameters. An activity can only reference the data passed to it by the input parameters. It exposes the results of operations (or tasks) specified in the activity only through the output parameters. Inter-activity parameter mappings are used to define the data flows between activities explicitly. By doing this, an activity definition is encapsulated. A modification to the task items inside an activity can be made without affecting other part of the model.

3 *Inclusion of web service requests in activity definitions*: in an inter-organisational process model, the activity specification should include web service request(s) that can be serviced by different business organisations. For this reason, we include web service requests in the activity definition in addition to direct invocations of manual or automated tasks that are performed within an organisation. Web services specified in a process model are outsourced services.

4 *Introduction of event, trigger and rule*: an important extension we make to WPDL is the introduction of events, rules and triggers in a process model specification. The activities inside the process model can post the following types of events:

   a Before-Activity-Event (BE): before an activity is executed, a BE can be posted.

   b After-Activity-Event (AE): after the processing of an activity, an AE can be posted.

   c External Events (EEs): an activity can also explicitly post events that are defined externally to report data conditions or exceptions to the subscribers of the events. Posting an EE is regarded as an operation or a task item in an activity.

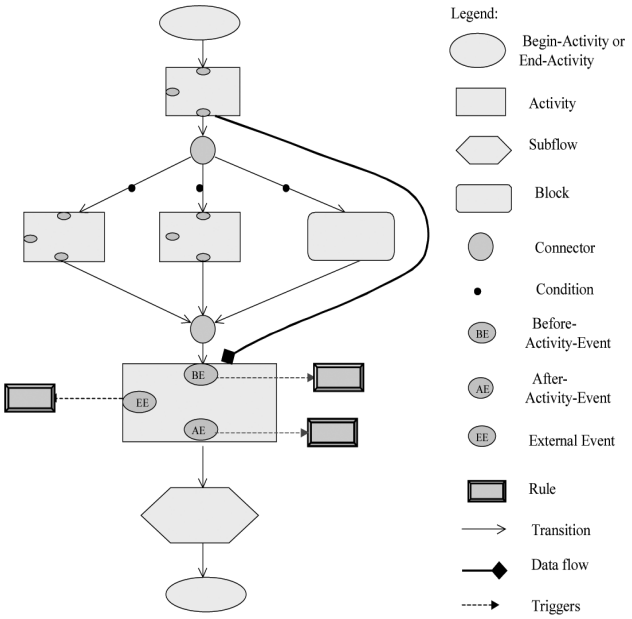
By introducing these events in a process model, the execution of a workflow instance would post events to automatically trigger the processing of some business rules. Events can be defined and managed by different organisations. They are parameterised; the values of the parameters are data relevant to each occurrence of an event (i.e. event data). Rules are condition-action-alternative-action rules, which verify the truth condition of event data and/or some other local data to determine whether to perform some operations or some alternative operations. Rules are also defined and managed by different organisations. Triggers are specifications, which specify the occurrences of which distributed events should trigger the processing of which distributed rules. Similar to events and rules, triggers are defined and managed by different collaborating organisations. In this work, events are defined at the design phase as an integral part of a business process model and are posted and triggered during the enactment of a business process by the workflow engine. When a model is being designed, the model designer specifies when (before or after an activity) and what (synchronous or asynchronous) events will be posted at different stages of executing a process model. Synchronous events are access points of a process model where organisations, which use the process model to conduct business processes, can attach customised business rules to adapt to a changing business environment. Asynchronous events can also be posted within a process model to notify interested organisations

or their application systems of the processing milestones of an enacted business process and/or exception conditions. Events, rules and triggers can be separately defined, stored and activated in a distributed fashion. Triggers and rules can either be defined at the design phase as a part of business process model, at the customisation phase where different triggers and rules can be defined for different organisations, or can be defined during the execution of the business process to handle unexpected situations. As we shall explain inthe remaining part of this paper, the introduction of events, rules and triggers gives both DWM and DynaFlow their active, adaptive and customisable properties.

By extending WPDL in the ways described above, we construct our DWM. In DWM, the modelling constructs used to define a process model include activity, transition, connector, subflow, block, data flow, event, rule and trigger. In addition to regular activities, two special purpose activities are introduced: the Begin-Activity and the End-Activity. They are used to define the entry and exit points of a process model or a block in a process model. A block is a connected network of transitions, activities, subflows and connectors. It is connected to a process model only via its Begin-Activity and the End-Activity. A block can be a loop-block, which defines a set of activities that can be iterated until an exit condition is met.

The graphic representation of a business process model is shown in Figure 5. The nodes in the graph can be activities, connectors or subflows. The connectors and transitions together specify the control flow. The data dependencies among the activities and subflows are captured by data flows. The data flows can be either implicitly defined together with the transitions, or they can be explicitly defined. The three types of events that can be posted by activities are BE, AE and EE. Business rules can be attached to these events by using trigger specifications (represented by dashed lines).

Figure 5 Business process model in DWM



4.3 Activity specification

Activity is the main modelling construct in a process model. The syntax of an activity definition is shown in Figure 6.

Figure 6 Activity definition

```
ACTIVITY   <activity id>
   [DESCRIPTION  <description>]
   [PERFORMER   < business type name> (<performer selection constraint>)]
   IN_PARAMETER   <input parameter list>
   OUT_PARAMETER  <output parameter list>
   [WF_EVENTS   <workflow event list>]
   [ACTIVITY_VAR   <variable list>]
   IMPLEMENTATION   <activity body>
END_ACTIVITY
```

The WF_EVENT clause specifies the workflow events that this activity posts. The PERFORMER clause specifies the business type of the business organisation whose web services are requested in the activity body (e.g. toy retailer as a business type). The process model designer can also specify the performer selection constraint to restrict the selection of a proper performer (i.e. a service provider). The model designer can specify the performer of an activity as a specific organisation (CONSTANT); any suitable organisations of a specified business type (ANY); an organisation that is the same as that of another activity (SAME_AS); or an organisation that is computed by another activity (VARIABLE). The activity body in the IMPLEMENTATION clause specifies a set of task items that need to be done inside the activity. The ACTIVITY_VAR defines the variables that can be used inside the activity body. There are three basic types of task items inside the activity body:

- web service requests are the main task items in activity definitions. The web service request constraints can be defined for these requests
- programming code can be included as inline code in the activity body to perform some simple computation, to make a variable assignment, or to make a call to a local application
- an activity can also explicitly post events inside its activity body.

4.4 Dynamic properties provided by DWM and DynaFlow

DWM and thus DynaFlow, which uses DWM as its underlying meta-model, provide the following dynamic properties:

1 *Active*: the business events and business rules are integrated with business processes. A business process is active in the sense that its enactment may post synchronous and/or asynchronous events to trigger the processing of business rules. Synchronous workflow events may also trigger rules that make changes to the business process by either adding additional tasks to the business process or altering the process model itself at run-time. These rules can be defined by either the process model designer or

the organisation that initiates a workflow instance. Asynchronous workflow events can signal the processing milestones of an enacted business process. Interested organisations can subscribe to these events and define rules to react to these events.

2 *Flexible*: the web service requests specified inside a process model are defined according to standardised web service templates. These web service requests are bound to suitable service providers in a virtual enterprise during the enactment of the business process through a dynamic service binding mechanism. Thus, a process model defined in this way is flexible in that the actual business organisations that take part in the business process are determined at run-time. Changes in the membership of the service providers of a particular service will usually not affect the enactment of a business process. The Workflow Engine accomplishes the dynamic service binding with the help of the Broker Server, which performs the constraint-based service provider selection (Meng et al., 2002).

3 *Adaptive*: during the execution of a business process, the business rules triggered by the synchronous workflow events that are posted by the process can modify the process model itself to adapt to the changing business environment. The process model defined in DWM is thus adaptive. Modifications to activity definitions or to the structural relationships and constraints of these activities can be more easily done because their specifications are separated. These modifications include: delete/add a transition, delete/add a data flow link, replace an activity and/or modify the condition of a transition.

4 *Customisable*: inter-organisational process models are designed for conducting the business of a virtual enterprise. People who work for a participating organisation and have the right of access to a process model can enact the model at its home site. For each enactment (i.e. a workflow instance) at its home site, an organisation may want to customise the model to suit its local business policies, constraints or regulations. This can be achieved by defining its own set of business rules to perform additional work and/or to modify the process model. These rules are stored in a replica of the ETR server at the site of the organisation. During the execution of every workflow instance, the synchronous workflow events are posted to the local ETR server to trigger these rules. We shall call this type of customisation the organisational customisation.

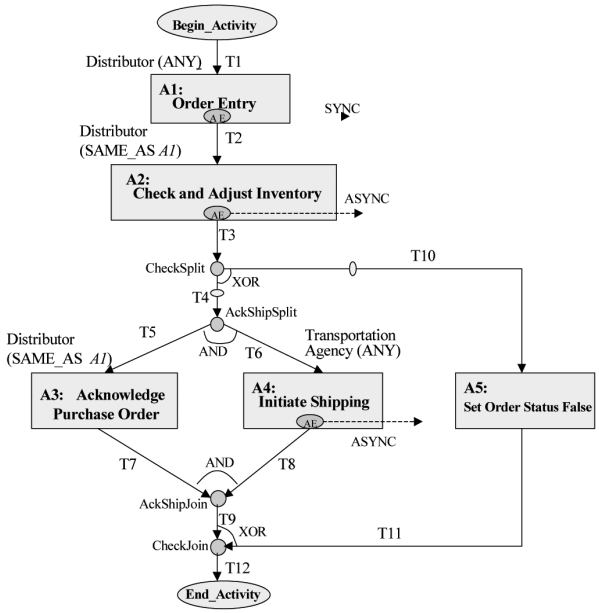
Also, in different enactments of the same process model, an organisation may want different rules to be triggered. This can also be achieved by defining different sets of rules for different enactments. Before a workflow instance is initiated, a unique id is generated for it. This id is incorporated in the rules that should be invoked during the processing of the workflow instance. When a synchronous event is posted, the instance id of the workflow instance that posts this event is passed to the rules and is used to identify the proper rules that are associated with the event. We call this type of customisation the instance customisation.

4.5 Sample scenario: order processing in supply chain community

We use an order processing in a supply chain as a scenario to illustrate how DWM is used to define a business process model and how the dynamic properties described above can be achieved. Suppose several organisations form a supply chain named Supply_Chain_Community. The participating organisations are categorised into four different business types according to the different roles they play: Retailer, Distributor, Manufacturer and Transportation Agency.

A process model, OrderProcessing, is defined in DWM and can be used by distributors in the Supply_Chain_Community to process the orders issued by retailers, as illustrated in Figure 7. The Distributor receives an order from a retailer and checks the inventory to make sure that there are enough products in the inventory to satisfy the order. If the order can be satisfied, the Distributor adjusts the inventory accordingly by reducing the quantity of the product. It then acknowledges the order and asks the Transportation Agency to ship the product to the Retailer. If there are not enough items of the product for this order, the order processing fails.

Figure 7 *OrderProcessing* model for *distributors* in the *Supply_Chain_Community*



In Figure 7, activities are represented by boxes. The business-type information of each activity is indicated above the box. The performer selection constraint of each business type is enclosed within parentheses following the business type. For example, 'ANY' following the business type TransportationAgency means that the web service requests in the activity can be serviced by any

transportation agency that is bound to these requests at run-time through the constraint-based dynamic service-binding process.

The BEs and AEs are specified by the ovals inside the activity boxes. 'SYNC' and 'ASYNC' specify the types of the workflow events to be posted. To avoid cluttering the figure, we do not show the data flow information and the detailed specification of the activities.

This model is replicated at the ISEE Hubs of the collaborative organisations. A qualified organisation/user can enact this process model to process orders from retailers. It can customise the process model by defining business rules that enforce its local business policies, then connecting these rules to the synchronous workflow events of the process model by using trigger specifications.

For example, the distributor Worldwide, Inc. has a local business policy, which says that, after the order from the retailer is received, the credit history of this retailer needs to be checked before further processing. To enforce this policy, Worldwide can define a business rule and attach it to the synchronous event that is posted after activity A1 (Order Entry) has been processed. This rule is triggered to check the credit history of the retailer who submitted the order. If the credit history of the retailer is good, the processing will continue. Otherwise, the rule can modify the process model at run-time so that after activity A1 (Order Entry) is finished, activity A5 (Set Order Status False) is scheduled for processing. To achieve this, transition T2 needs to be deleted and a new transition, which goes directly from A1 to A5, needs to be added. The definitions of the event, trigger and rule are shown in Figure 8.

Business rules attached to workflow events can also in turn enact other business process models. For example, if Worldwide wants to check if replenishment to the inventory is needed after adjusting the inventory, it can define a business rule and attach it to the asynchronous event posted after activity A2 (Check and Adjust Inventory). This business rule first checks if the quantity of the product in the inventory has reached the replenishment threshold. If this is the case, a workflow instance of the InventoryReplenishment process model, which generates an order of this product to a manufacturer, is created and executed.

Figure 8 Sample definitions: (a) workflow event: (b) business rule and (c) trigger

```
IN              workflowSchema
EVENT           OrderProcessing_After_OrderEntry
ATTRIBUTES      WFID wfInstanceID, ProductDesc product, UserInfo
                userInfo, int quantity, Date deliverDate
DESCRIPTION     event posted after activity OrderEntry
TYPE            AFTER_ACTIVITY_SYNC
RETURNS         ProductDesc product, UserInfo userInfo, int quantity,
                Date deliverDate
```

(a)

```
RULE          CheckCreditHistory_Rule(WFID wfInstanceID, ProductDesc product,
              UserInfo userInfo, int quantity, Date deliverDate)
RETURNS       ProductDesc product, UserInfo userInfo, int quantity, Date deliverDate
DESCRIPTION   Check the credit history of the customer. If the credit history is
                bad, dynamically change the model to set the order status to false.
RULEVAR       existing LocalServices localService
              existing WFEngine wfEngine
              boolean creditStatus          // variable used in action
CONDITION     true                          // the action will be enabled only when
                                            // condition is true
ACTION
              creditStatus = localService.checkCreditHistory(userInfo)
              if (!creditStatus) {
                 // if credit history is bad, call API of workflow engine to modify
                 // the process model
                 wfEngine.deleteTransition(wfInstanceID, "T2");
                 wfEngine.addTransition(wfInstanceID, "newTransitionName",
                          "OrderEntry", "SetOrderStatusFalse");
              }
ALTERACTION       None
```

(b)

```
TRIGGER         After_OrderEntry_SYNC_Trigger
DESCRIPTION     Link the event OrderProcessing_After_OrderEntry to rule
                CheckCreditHistory_Rule
TRIGGEREVENT    OrderProcessing_After_OrderEntry(wfInstanceID, product, userInfo,
                quantity, deliverDate)
RULESTRUC       CheckCreditHistory_Rule(WFID wfInstanceID, ProductDesc
                product, UserInfo userInfo, int quantity, Date deliverDate)
```

(c)

5 Design and implementation

5.1 Main components in DynaFlow

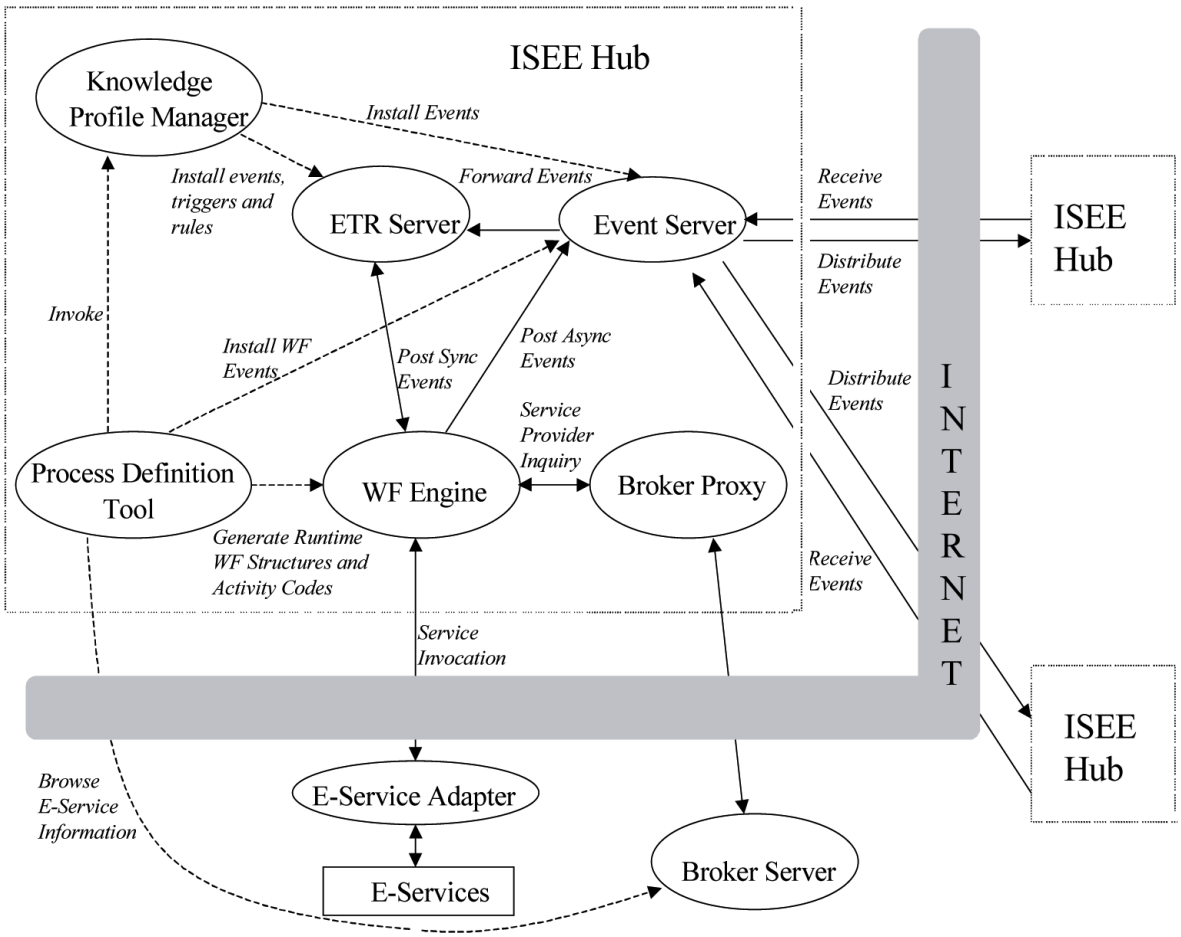
The main servers in DynaFlow and the relationship among them are shown in Figure 9. The dashed lines represent build-time actions, and the solid lines represent run-time actions.

The Process Definition Tool is a major build-time tool for process modelling. It is a user-friendly graphic editor for creating and customising process models using DWM as the underlying model. It invokes another Graphical User Interface (GUI) that has been implemented for a Knowledge Profile Manager (KPM) (Lee et al., 2001; Su et al., 2001) to define business events, rules and triggers that are related to a process model. It also invokes a Constraint Definition GUI (not shown in Figure 9) to define constraints that are associated with the web service requests specified in the process model. The definitions of events are passed on to both the local ETR Server and the local Event Server for installations of the events. The definitions of rules and triggers are given to and managed by the local ETR Server. The Process Definition Tool can also generate the run-time workflow structures and activity code from the meta-information of the process model by invoking a Workflow Code Generator (not shown in Figure 9). The run-time workflow structures and activity code are used by the local Workflow Engine to schedule the execution of workflow instances.

During the execution of a workflow instance, the Workflow Engine posts asynchronous workflow events to the local Event Server. It also posts synchronous events directly to the local ETR Server to trigger the rules that are linked to these events. The Workflow Engine would also contact the Broker Server to get the service provider information for those web services that are specified in a process model. To invoke a web service, the Workflow Engine generates a SOAP message for the web service request and sends it to the web service adapter at the service provider's site to invoke the web service. A SOAP message containing the web service result will be returned to the Workflow Engine after the invocation is done.

The Event Server handles the incoming and outgoing events to and from an ISEE Hub. It also provides an event subscription facility for the participants of the virtual enterprise to register their interest by subscribing to certain events. When an event occurs, the Event Server would notify the remote Event Servers of those ISEE Hubs whose users or application systems have subscribed to the event. The remote Event Servers then forward the event to their local ETR Servers to initiate the processing of triggers and rules that are associated with the event. Each ETR Server processes the triggers and rules installed in an ISEE Hub. It receives events from the Event Server (for asynchronous workflow events) or directly from the Workflow Engine (for synchronous workflow events) and performs the trigger and rule processing. The details of the ETR Server can be found in Lee et al. (2004).

Figure 9 Architectural components of DynaFlow



The Broker Server provides several functions to DynaFlow. It provides facilities for the registration and publication of web services provided by web service providers and manages the registered web service information. It also provides facilities for web service requestors and process model designers to browse and access the registered web services information. The constraint-based selection of service providers is another important function that the Broker Server provides. The Broker Proxy in each ISEE-hub is the local proxy of the remote Broker Server. The Workflow Engine in the ISEE-hub contacts the Broker Server through it.

Web service adapters are installed at participating organisations' sites to wrap their services, which could be implemented in different ways. These services can then be invoked using SOAP messages that are transmitted through the HTTP protocol according to the formats of the corresponding web service templates.

5.2 Code generation approach of workflow engine

To support the run-time modification to a business process model, we use a code generation approach in the implementation of the Workflow Engine. A Workflow Code Generator is used to process a process model's meta-information and to generate run-time workflow structures and activity code. The run-time workflow structures contain the transition information and data flow information of activity interconnections. The Workflow Engine interprets these structures to enforce the inter-activity dependencies of a workflow instance. The specifications of the activity bodies are translated into Java programs and compiled into Java classes, which are called activity code. When the Workflow Engine schedules an activity for execution, the engine simply loads the corresponding activity code from a run-time repository and executes the code directly to perform the specified task items.

This code generation approach results in a lightweight, efficient and adaptive Workflow Engine. It is lightweight because the Workflow Engine does not need logic to interpret the activity specification to execute its task items. Taking advantage of the performance of the compiled classes, the Workflow Engine is efficient. Finally, as the Workflow Engine 'interprets' the run-time workflow structures to enforce the inter-activity dependencies, it is adaptive in that it is easy to modify the course of its execution by modifying the workflow structures at run-time. Dynamically loading activity code at run-time also enables the run-time modification of the task items inside an activity. Any changes made to the task items in an activity body will be reflected in the execution of the activity as long as the regenerated code is placed in the run-time repository before the execution of the activity.

There are basically three types of run-time workflow structures: entity structure control flow structure and data flow structure.

Entity structures provide the Workflow Engine with the necessary information to schedule and execute activities, blocks and subflows in a process model. The contents for the entity structures correspond to the meta-information of these entities.

A control flow structure is generated for each of the following entities of a process model: activity, subflow, block, connector and End-Activity. It captures the control dependency between the entity and its preceding entities in a process model. The three main parts in a control structure are shown below:

- *Entity name*: the name of the entity of the control flow structure.
- *Aggregation property*: if the entity is a JOIN connector, the aggregation property of the control structure is the same as the aggregation property of the connector (i.e. AND, OR or XOR). Otherwise, the aggregation property is SIMPLE.
- *Transition(s)*: if the entity of the control structure is a JOIN connector, there are multiple transitions in this control flow structure. Otherwise, there is only one transition in the control flow structure.

A data flow structure specifies data dependency, that is, the parameter mapping information between two entities in a process model.

In the code generation approach, an activity body, which may contain several task items, is translated into activity code. The generated code consists of three parts:

1 member variable declarations, which specify input/output parameters, activity variables and other variables used to handle web service requests and event postings

2 a constructor method, in which initialisations are performed and

3 an activate method, in which all task items in an activity are performed.

The activate method is the main part of an activity code. The values of the input parameter of this activity need to be passed to the activate method. The task items defined in the activity body are translated into Java statements in the activate method. When there are web service requests defined for the activity, if the activity's performer is of type ANY, performer information and web service request constraints are passed to the activate method to perform the dynamic service binding; otherwise, the service provider's URL is passed to the activate method by the caller (i.e. the Workflow Engine) directly. For each web service request, the activity code generates a SOAP XML message containing the values of input attributes of this web service and sends it to the web service adapter at the service provider's site through the service client to invoke the web service. After the web service invocation, the activity code extracts the output values from the returned SOAP XML message. For each event posting inside an activity body, the activity code generates the corresponding event object and posts it to the ETR Server (synchronous events) or to the Event Server (asynchronous events). Inline codes given in the activity definition are

directly copied to the activity code. At the end of the activate method, the result, which contains the service provider's URL (if dynamic binding is performed in the activate method) and the output data of the activity, are returned.

5.3 Implementation of workflow engine

The Workflow Engine uses a multi-thread architecture to allow the concurrent execution of multiple workflow instances. The components of the Workflow Engine are shown in Figure 10. When a workflow instance is initiated, a Workflow Scheduler thread is created. The Workflow Scheduler interprets run-time workflow structures to enforce inter-activity dependencies during activity scheduling.

The execution of the workflow instance begins from its Begin-Activity. After an entity has been successfully completed, the Workflow Scheduler takes the following steps to schedule the execution of the workflow instance:

1 *Data flow mapping*: if the completed entity is an activity, a subflow or a block, the Workflow Scheduler maps its output data to the input parameters of the entities, which are the target entities of data flow structures with the completed entity as the source entity.

2 *Control flow structure evaluation*: the Workflow Scheduler retrieves the control flow structures, which contain a transition with the completed entity as the source entity, and evaluates them. The target entity can only be scheduled when transition(s) and aggregation expression (if there is any) are evaluated to True.

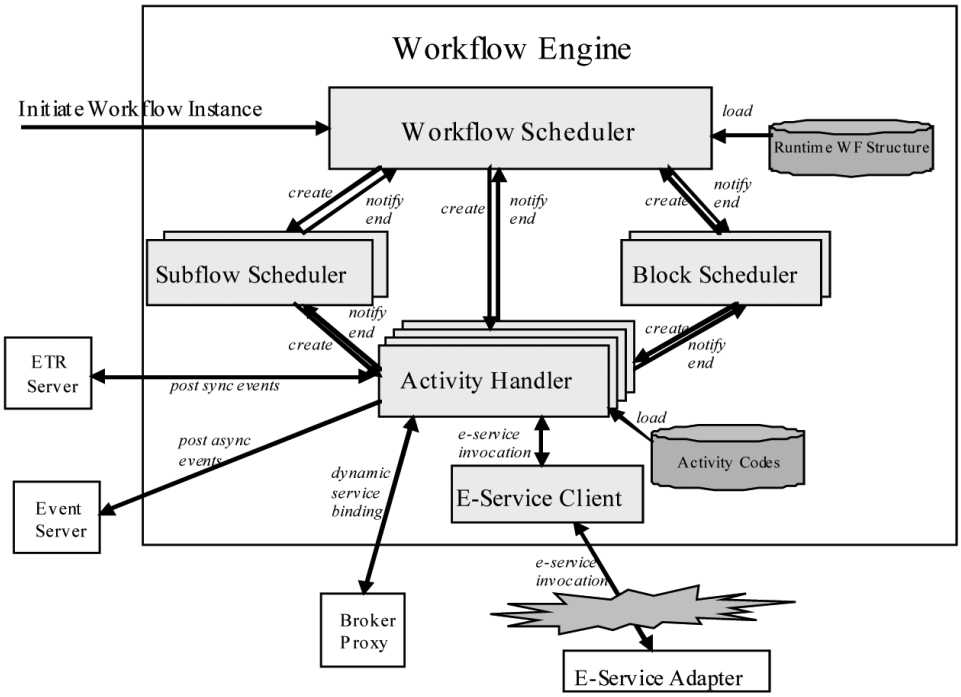
3 *Entity scheduling and execution*: if the scheduled entity is an activity, a subflow or block, the Workflow Scheduler creates an Activity Handler, a Subflow Scheduler or a Block Scheduler, respectively, to handle its execution. If the scheduled entity is a connector, the Workflow Scheduler sets the status of the connector to 'complete' and continues to evaluate the succeeding control flow structures of this connector. If the scheduled entity is the End-Activity of the process model, the workflow instance is completed. The Workflow Scheduler repeats steps 1–3 until the End-Activity is reached.

After an Activity Handler has been created, it first posts the BEs if they are specified for the activity. The synchronous event is posted directly to the local ETR Server, to trigger the associated business rules. Asynchronous workflow event is posted to the Event Server so that the Event Server can distribute them to the event subscribers. The Activity Handler then loads the activity code and executes it. The execution of the activity code is actually the execution of task items specified inside the activity definition. After the execution of the activity code, the result is returned to the Activity Handler. The Activity handler then posts the AEs and passes the output data of the activity to the Workflow Scheduler (or the Block Scheduler if the activity is inside a block or to the Subflow Scheduler if the activity is in a subflow).

5.4 Run-time modifications to business process model

As the Workflow Engine 'interprets' the run-time workflow structures to schedule workflow instances, the run-time modification to the course of executing a process model can be accomplished by modifying the run-time workflow structures.

Figure 10 The components of the workflow engine



Deletion/addition of a transition is accomplished by deleting/adding the transition information from/to the corresponding control flow structure. If the deleted transition is the only transition in the control flow structure, the control flow structure will then be deleted from the run-time workflow structures. Deletion/addition of a data flow is accomplished by deleting/adding the corresponding data flow structure from/to the run-time workflow structures.

Some business policies are embedded inside an activity body. When a change is made to these policies during the execution of a workflow instance, the running workflow instance may need to replace the original activity with the modified one. We achieve this by modifying the name of the original activity in the control structure. Thus, the Workflow Engine will load the new activity code instead of the old one.

In this implementation, the condition of a transition is evaluated in an interpretive fashion. It can be replaced by an alternative one during the execution of a workflow instance.

6 Conclusion

In this paper, we have presented a DWM and a dynamic workflow management system for modelling and controlling the execution of inter-organisational business processes of a virtual enterprise. The design of the DWM is an extension of the underlying model of WfMC's WPDL. The extension includes adding the concepts of event, rule and trigger into WPDL, introducing the connector construct, encapsulating activity definitions and supporting web service requests. The dynamic workflow management system, DynaFlow, uses DWM as its underlying model. Its implementation consists of a Workflow Server, an ETR Server, an Event Server and a Broker Server. The Workflow Engine of the Workflow Server handles the execution of workflow instances. It makes use of the services provided by other servers to archive the following dynamic properties:

1 constraint-based dynamic binding of web service requests to web services enables the flexible use of inter-organisational data/software/application resources, and the accurate selection of service providers

2 the inclusion of events, rules and triggers in business process modelling enables the posting of synchronous and asynchronous events during the enactment of a business process to, not only signal the progress of the business process, but also automatically activate operations that are relevant to the process

3 inter-organisational process models and their enactments (i.e. model instances) can be customised by triggering different rules to enforce different organisations' local policies, constraints and regulations and

4 workflow process models can be modified at run-time to adapt the models to the changing business environment and process requirements.

The code generation approach described in this paper enables a lightweight and efficient implementation of the above dynamic properties.

References

Adam, N., Adiwijaya, I. and Atluri, V. (1998) 'EDI through a distributed information systems approach', *Proceedings of the 31st Hawaii International Conference on System Sciences*, Kohala Coast, HI, January.

Alonso, G., Fiedler, U., Hagen, C., Lazcano, A., Schuldt, H. and Weiler, N. (1999) 'WISE – business to business e-commerce', *Proceedings of the Ninth International Workshop on Research Issues on Data Engineering: Information Technology for Virtual Enterprises*, Sydney, Australia, March.

Ariba Corporation, IBM Corporation, and Microsoft Corporation (2000) 'UDDI technical white paper', September, Available at: http://www.uddi.org.

Box, D., Ehnebuske, D., Kakivaya, G., Layman, A., Mendelsohn, N., Nielsen, H.F., Thatte, S. and Winer, D. (2000) 'Simple object access protocol (SOAP) 1.1', May, Available at: http://www.w3.org/TR/SOAP.

Business Process Execution Language for Web Services (2002) 'Specification Version 1.1', Available at: http://www-106.ibm.com/developerworks/library/ws-bpel/.

Ceri, S., Grefen, P. and Sanchez, G. (1997) 'WIDE – a distributed architecture for workflow management', *Proceedings of the Seventh International Workshop on Research Issues in Data Engineering*, Birmingham, UK, April.

Chen, Q. and Hsu, M. (2001) 'Inter-enterprise collaborative business process management', *Proceedings of the International Conference on Data Engineering*, Germany.

Christensen, E., Curbera, F., Meredith, G. and Weerawarana, S. (2001) 'Web services description language (WSDL) 1.1', *W3C Note*, March, Available at: http://www.w3.org/TR/WSDL.

CrossFlow Consortium (2000) 'Flexible change control', *CrossFlow Deliverable: D8.a*, February, Available at: http://www.crossflow.org/.

Dayal, U., Hsu, M. and Ladin, R. (2001) 'Business process coordination: state of the art, trends, and open issues', *Proceedings of the 27th VLDB Conference*, Roma, Italy.

Degwekar, S., Su, S.Y.W. and Lam, H. (2004) 'Constraint specification and processing in web services publication and discovery', *Proceedings of International Conference on Web Services (ICWS04)*, San Diego, CA, 6–7 July, pp.210–217.

Geppert, A. and Tombros, D. (1998) 'Event-based distributed workflow execution with EVE', *IFIP International Conference on Distributed Systems Platforms and Open Distributed Processing*, The Lake District, England, September.

Grefen, P. and Hoffner, Y. (1999) 'CrossFlow – cross-organizational workflow support for virtual organizations', *Proceedings of the Ninth International Workshop on Research Issues on Data Engineering: Information Technology for Virtual Enterprises*, Sydney, Australia, March.

Helal, A., Su, S.Y.W., Meng, J., Krithivasan, R. and Jagatheesam, A. (2002) 'The internet enterprise', *Proceedings of the Second IEEE/JPSJ Symposium on Applications and the Internet (SAINT02)*, Nara, Japan, January.

Laymann, F. (2001) 'Web services flow language', May, Available at: www-4.ibm.com/software/solutions/ web services/pdf/WSFL.pdf.

Lazcano, A., Alonso, G., Schuldt, H. and Schuler, C. (2000) 'The WISE approach to electronic commerce', *International Journal of Computer Systems Science and Engineering*, *Special Issue on Flexible Workflow Technology Driving the Networked Economy*, Vol. 15, No. 5.

Lee, M. (2000) 'Event and rule services for achieving a web-based knowledge network', *PhD Dissertation*, Department of Computer and Information Science and Engineering, University of Florida, Available at: http://www.cise.ufl. edu/tech-reports/ tech-reports/tr00-abstracts.shtml, TR 002.

Lee, M., Su, S.Y.W. and Lam, H. (2001) 'A web-based knowledge network for supporting emerging internet applications', *WWW Journal*, Vol. 4, Nos. 1/2, pp.121–140.

Lee, M.S., Su, S.Y.W. and Lam, H. (2004) 'Event and rule services for achieving a web-based knowledge network, knowledge-based systems', *Journal of Knowledge Based Systems*, Vol. 17, pp.179–188.

Liu, Y., Yu, F., Su, S.Y.W. and Lam, H. (2003) 'A cost-benefit evaluation server for decision support in e-business', *Journal of Decision Support Systems and Electronic Commerce*', Vol. 36, No. 1, pp.81–97.

Meng, J., Krithivasan, R., Su, S.Y.W. and Helal, S. (2002) 'Flexible inter-enterprise workflow management using e-services', *Proceedings of the Fourth IEEE International Workshop on Advanced Issues of E-Commerce and Web-based Information Systems*, California, June.

Meng, J., Su, S.Y.W., Lam, H. and Helal, S. (2002) 'Achieving dynamic inter-organizational workflow management by integrating business processes, events, and rules', *Proceedings of the 35th Hawaii International Conference on System Sciences*, Hawaii, January.

Müller, R. (2002) 'Event-oriented dynamic adaptation of workflows: model, architecture and implementation', *PhD Dissertation*, University of Leipzig.

Muller, R. and Rahm, E. (1999) 'Rule-based dynamic modification of workflows in a medical domain', *Proceedings of BTW99*, Freiburg, Germany, February.

Phillips, C. and Meeker, M. (2000) 'The B2B internet report: collaborative commerce', *Morgan Stanley Dean Witter*, April, Available at: http://www.morganstanley.com/ techresearch/b2b/b2bp1a.pdf.

Reichert, M. and Dadam, P. (1998) 'Adept_flex-supporting dynamic changes of workflows without losing control', *Journal of Intelligent Information Systems, Special Issue on Workflow and Process Management*, Vol. 10, No. 2, pp.93–129.

Sheng, Q.Z., Benatallah, B., Dumas, M. and Mak, E.Q. (2002) 'SELF-SERV: a platform for rapid composition of web services in a peer-to-peer environment', *Proceedings of the 28th VLDB Conference*, Hong Kong, China.

Sheth, A., Aalst, W. and Arpinar, I. (1999) 'Process driving the networked economy', *IEEE Concurrency*, Vol. 7, No. 3, pp.18–31.

Su, S.Y.W., Huang, C., Hammer, J., Huang, Y., Li, H., Wang, L., Liu, Y., Pluempitiwiriyawej, C., Lee, M. and Lam, H. (2001) 'An internet-based negotiation server for e-commerce', *VLDB Journal*, Vol. 10, No. 1, pp.72–90.

Su, S.Y.W., Lam, H., Lee, M., Bai, S. and Shen, Z. (2001) 'An information infrastructure and e-services for supporting internet-based scalable e-business enterprises', *Proceedings of the Fifth International Enterprise Distributed Object Computing Conference*, Seattle, WA, September.

Su, S.Y.W., Meng, J., Krithivasan, R. and Helal, S. (2003) 'Dynamic inter-enterprise workflow management in a constraint-based e-service infrastructure', *Electronic Commerce Research Journal*, Vol. 3, Nos. 1/2, Kluwer: Academic Publishers.

WfMC (1999) 'Interface1: process definition interchange V 1.1 final (WfMC-TC-1016-P)', October, Available at: http:// www.wfmc.org.

Zhou, T., Pu, C. and Liu, L. (1998) 'Dynamic restructuring of transactional workflow activities: a practical implementation method', *Proceedings of the Seventh International Conference on Information and Knowledge Management (CIKM'98)*, Washington, DC, November.