An Adaptive Workflow Engine Based on Web Services and Agents

Hongzhen Xu\textsuperscript{a,b,*}, Bin Tang\textsuperscript{a,b}, Ying Gui\textsuperscript{a}

\textsuperscript{a}Department of Computer Science and Technolog, East China Institute of Technology, Fuzhou 344000 Jiangxi Province, China

\textsuperscript{b}Application of Nuclear Technology Engineering Center of Ministry of Education, East China Institute of Technology, Nanchang 330013, Jiangxi Province, China

Abstract

At present workflow management systems are widely used to manage business processes in an organization, while the workflow engine provides a runtime execution environment, manages the overall processing and execution of workflow instances. One of the major limitations of current workflow engine is the lack of flexibility to support adaptive environments. In this paper we propose an adaptive workflow engine based on web services and multiagent technologies to support dynamic interpreting, configuration, monitoring and executing workflows. The main feature of this workflow engine is to integrate event-driven process execution, a unified interaction interface with external environments and exception handling mechanism.

Keywords: Workflow Engine; Web Services; Multiagent; Event-driven

1 Introduction

Due to advances in Information Technology, organizations typically design their information systems to handle their internal business processes and interactions with their partners. Each of these processes is usually made up of many smaller steps, such as filling in forms, getting audit and recording critical information in various places. Workflow technology has emerged as one of those technologies designed to support modeling, designing and executing business processes. Workflow is defined as the movement of tasks through a work process describing how tasks are structured, who performs them, the resources needed and their relative order [1]. Workflow management systems (WFMSs) are widely used to manage business processes due to their known benefits such as automation, coordination and collaboration between entities [2].

In all of these WFMSs, a core component is the workflow engine. The workflow engine provides a runtime execution environment, manages the overall processing and execution of workflow instances. The basic functions of the workflow engine include the interpretation of business process

\textsuperscript{*}Corresponding author.

Email address: xhz\_970163.com (Hongzhen Xu).
specification, creation of new process instances, execution and management of instances, communicat-
ing with workflow applications and clients, and so on.

The business environment today is undergoing rapid and constant changes. The way companies do business, including business processes and their underlying business rules, ought to adapt to these changes rapidly with minimum interruption of the ongoing operations. Traditional workflows were limited to rigid steps that took place entirely within a single organization, and do not consider possible future changes. Accordingly, workflow engines of these WFMSs have not been designed for dynamic environments requiring adaptive response. The existing available workflow engines usually do not offer sufficient support for dynamic managing workflow in an organization [3].

In this paper, we focus on solving some critical challenges that have to be addressed when designing a workflow engine to support dynamic and distributed environments. We propose an adaptive workflow engine based on web services. We integrate web services and multiagent technologies to support dynamic interpreting, configuration, monitoring and executing workflows. We use an event-driven method to handle process execution, use a unified interaction interface with external environments and provide some exception handling mechanism in our workflow engine.

2 State of the Art

2.1 Workflow and Workflow Management

Workflow is a concept closely related to reengineering, automating business and information processes in an organization. Workflow is the automation of a business process, in whole or part, during which documents, information or tasks are passed from one participant to another according to a set of procedural rules [4]. A workflow may describe business process tasks at a conceptual level necessary for understanding, evaluating, and designing the business process. On the other hand, workflows may capture information process tasks at a level that describes the process requirements for information system’s functionality [5]. A workflow is a high-level specification of a set of tasks and the dependencies between them that must be satisfied in order to accomplish a specific goal.

Workflow management is the automated co-ordination and control of such processes. Workflow management is a technology supporting the reengineering of business and information processes. It involves: 1) defining workflows, i.e., describing those aspects of a process that are relevant to controlling and coordinating the execution of its tasks, and 2) providing for fast design and implementation of the processes as business needs and information systems change [6].

2.2 Web Services

Web services represent a new class of interoperable, web-enabled software service, which are language, platform and location independent. It is an open XML-based technology providing a generic data exchange format and has been rapidly adopted by many vendors. Web services can easily be built upon existing applications, no matter what the underlying technology is. Because they are expected to have a growing familiarity and acceptance among many users and offer great technological promises, Web services are an interesting subject for the investigation
of their possible application in many systems [7]. Web Services combines the advantages of the component-oriented methods and web techniques. It can also publish, locate and transfer modularized application in web. Web Services expose to clients their capabilities rather than their implementations. They can be implementation independent and still be compatible with all client applications. Web services represent a kind of implementation of SOA (Service-Oriented Architecture). In addition, the operations of SOA can only process when the components of SOA interact. Therefore some standardized techniques are used in web services, including UDDI, WSDL, SOAP, and XML and so on. Web services become the best choice for developing application of SOA.

2.3 Multiagent Systems

Multiagent systems emerged as a new research area in the early 1990’s. It developed partly from distributed processing and partly from artificial intelligence and its modern incarnation as agents. In recent years, multiagent technology has been rapidly developed in order to fulfill the needs of new conceptual tools for modeling and developing complex software systems. An agent is a computer system that is situated in an environment, which is capable to make autonomous action in the environment in order to achieve its design objectives. The key aspect of agents is their autonomy, where the abilities to perceive reason and act in their surroundings environments, as well as the capabilities to cooperate with other agents to solve complex problems are essential [8].

3 Architecture of the Workflow Engine

We propose an adaptive workflow engine model based on web services and agents, given in Fig. 1, which has five components: the workflow configuration agent, the workflow enactment service agent, the administration and monitoring agent, the adapter agent and the exception handler agent. The workflow engine is designed to provide dynamic management of workflow, support for dynamic interpreting, configuration, monitoring and executing workflows, and use an event-driven method to handle process execution. For all workflow tasks the engine calls appropriate web services using the SOAP protocol. When a web service is called, the corresponding process of the relevant workflow is executed and its results give feedback to the engine, which proceeds to the next task until the business process is completed. In the workflow engine, each component is designed as an agent, which is capable of performing certain tasks. Users or organizations can provide services or solving problems through those agents. These agents act as coordinators whose main tasks are to coordinate the activities of business processes.

3.1 The Workflow Configuration Agent

The agent consists of: 1) an UDDI registrar which can interpret web services of a workflow system; 2) Ontologies which describe the roles, the tasks and the exchange of parameter data between business processes in terms of classes and instances; 3) a workflow configurator by which we can configure workflows of a workflow management system.

Web services are described with WSDL documents in the UDDI registrar, and ontologies are represented according to the RDFS specification. In this context, those web services can be understood as methods having input and output parameters. Ontologies output the association
of the classes and instances corresponding to those web services in an XML formatted document. Thus an efficient mechanism is provided making those web services of a system easy to discover and associate with the business process semantic information contained in ontologies.

In the workflow configuration agent, the first step is to associate the ontological information with the existing web services of a workflow system. Using ontologies, a XML configuration document is created which incorporate web services description according to the semantic representation of business processes. The second step is to configure workflows of business processes. The workflow configurator retrieves information from the XML configuration document. Finally, a BPEL4WS document, the outcome workflow of the workflow configuration agent, is provided as an input to the workflow enactment service agent and stored in a workflow repository.

### 3.2 The Workflow Enactment Service Agent

The agent is the core component of the workflow engine, which provides the run-time environment where one or more workflow processes are executed. It instantiates processes according to the process description and controls correct execution of activities interacting with users via worklists and invoking applications as necessary. The agent maintains control data and workflow relevant data, uses them to evaluate entry and exit criteria for activity steps. It is also responsible for supervisory actions of control and audit.

In our workflow engine, we take advantage of this event notification mechanism to implement process execution behavior. Process execution is determined dynamically as events are raised and corresponding activities are invoked. A key issue for our workflow engine is how to manage the process execution, how to pick the appropriate action to perform in response to an event. We define a process as a set of activity groups. Each activity group consists of an event (a newly arriving message) and a set of related guards and activities, as shown in Fig. 2. Guards are
condition expressions that can refer to both event parameters (message fields) and the current system state, and control whether or not their corresponding activity should execute as part of the response to the event.

![Diagram of process structure]

Each activity represents one possible response to its associated event when its corresponding guard is satisfied. Each activity consists of a set of actions and a set of triggers. The structure of an activity is shown in Fig. 3. Each trigger consists of a condition and a set of zero or more events, i.e. outgoing messages. After the action sequence of an activity has been completed, the trigger condition in a trigger is evaluated to determine what following events need to be raised to further the handling of the previous event.

![Diagram of activity structure]

When an event occurs, its guards are evaluated to determine which of its activities is to be executed. Once an activity is selected, its corresponding actions are executed, and then its triggers may raise further events. After this, if additional events have been raised, each is processed in the same way in turn.

### 3.3 The Administration and Monitoring Agent

The module is the key component to implementing dynamic workflow management. It can capture different aspects of business process modifications, and transfer the information to the workflow enactment service agent. It can translate the (changed) process specifications into workflow
models, and make decisions regarding handling of active workflow instances, create revised process
definitions according to the information from the workflow enactment service agent.

The agent can configure the modification policy, define a set of affected instances, manage
modification logs, and generate compliance graphs which will dictate the revised schedules for
the given set of affected instances. It acts as a mediator between management’s proposals and
strategies, and the propagation of these proposals to the operational level.

3.4 The Adapter Agent

Many present workflow management systems assume specific ways of interacting with resources,
and do not consider that in an enterprise-wide environment, they may be required to interact with
a variety of resources, from human to different systems, applications, or even machines. Each
resource may require a different kind of interaction, from reply/request to iterated attempts of
task acceptance and fulfillment. In our workflow engine, we do not invoke any resource directly.
We use an adapter agent to interact with any resource. In essence, the adapter agent is an
application wrapper that allows the workflow engine to invoke external components according
to an engine-defined interface. It allows the engine to interact with users, databases, external
systems and third-party data formats such as process definition languages. In this way it is
possible to invoke any number or type of resources without having to make any changes to the
way the workflow engine executes processes.

3.5 The Exception Handler Agent

Exceptions are commonly regarded as those abnormal events that happened during the execution
of the workflows. In our workflow engine, we use an exception handler agent to deal with all
exceptions. We classify exceptions into the following categories.

1) Computing and communication errors. These exceptions are caused by computer malfunc-
tions such as disk crash, computing error and communication system problems such as network
down.

2) Workflow operation errors. These exceptions are those which violate the assumptions of
workflow definitions such as there is no appropriate actor for a task.

3) Specification violation. In some cases, the actual outcome may violate the specifications. For
example, a task does not complete within the specified duration.

4) External exceptions. External exceptions are those exceptions raised from the external envi-
environment such as cancellation of a workflow by a user.

4 Case Study

In this section, we show how the main features of our workflow engine model can be effectively
implemented using off-the-shelf system infrastructure technologies such as C# and the .NET
framework. We used an example of the scientific research management system of East China
Institute of Technology [9] to show how to address some common implementation problems of
the workflow engine.
4.1 Event-driven Process Execution

In the system, we define two types of events:

1) Internal events, which are used to control workflow among activity groups within the same business process.
2) External events, which control the communication between interacting business processes.

We use just the information input event of the system to illustrate how to design an event-driven business process execution according to our workflow engine. The input information in the system includes administrator registration information, teacher registration information, paper information, project information, and so on. Fig. 4 shows how the InformationInput activity group could be programmed using our workflow engine.

<table>
<thead>
<tr>
<th>Event</th>
<th>InformationInput</th>
</tr>
</thead>
<tbody>
<tr>
<td>Guard</td>
<td>Administer registration information</td>
</tr>
<tr>
<td>Activity</td>
<td>Administer registration audit</td>
</tr>
<tr>
<td>Action</td>
<td>Construct a science and technology department administrator audit event</td>
</tr>
<tr>
<td>Trigger</td>
<td>Raise the registration-audit-completion event</td>
</tr>
<tr>
<td>Guard</td>
<td>Teacher registration information</td>
</tr>
<tr>
<td>Activity</td>
<td>Teacher registration audit</td>
</tr>
<tr>
<td>Action</td>
<td>Construct a secondary college administrator audit event</td>
</tr>
<tr>
<td>Trigger</td>
<td>Raise the registration-audit-completion event</td>
</tr>
<tr>
<td>Guard</td>
<td>Paper information</td>
</tr>
<tr>
<td>Activity</td>
<td>Paper audit</td>
</tr>
<tr>
<td>Action</td>
<td>Construct a secondary college administrator audit event</td>
</tr>
<tr>
<td></td>
<td>Construct a science and technology department administrator audit event</td>
</tr>
<tr>
<td>Trigger</td>
<td>Raise the paper-audit-completion event</td>
</tr>
</tbody>
</table>

Fig. 4: Information Input Event

As mentioned in section 3.2, events play a vital role in driving the workflow model. While the underlying .NET event publish/subscribe mechanism can be used to implement the basic control flow mechanism. Our current implementation makes use of the .NET Event mechanism which allows code to subscribe to events and be invoked whenever these events are raised. In our engine implementation, internal events are mapped directly to .NET Events, and external events are implemented using technologies such as .NET Remoting.

4.2 The Adapter Agent

Our workflow engine can provide interaction with external resources such as the workflow repository, user, other applications or existing systems through the adapter agent. The adapter agent allows external application services to be invoked during process execution through a standardized interface. We implemented the adapter agent within the standardization efforts of the Workflow Management Coalition. The Object Management Group has proposed a standardized interface for workflow management systems, called the Workflow Management Facility (WMF) [10]. The WMF specifies a set of CORBA interfaces to provide access to the runtime environment of a workflow management system. Any external application service that implements this interface can be invoked during process execution of the workflow engine.
4.3 Exception Handling

We use the exception handler agent to deal with all exceptions of the system. Exception detections and recovery handling are the duties of the exception handler agent in the workflow engine. We define a set of procedures relating to handle different abnormal conditions in advance, which include five different level exceptions. They are: 1) Workflow level; 2) Task level; 3) Agent assignment level; 4) Data level; 5) User level.

The exception handler agent is pro-active, i.e. it will detect and monitor the status of the workflow and system environment continuously. Whenever there are exceptions, the exception handler agent will raise the exceptions, analyze the exceptions, find out the handling methods and activate the recovery procedures. We define the followings actions in recovery from exceptional cases in our system, 1) Cancel the task; 2) Suspend the task; 3) Rollback the task; 4) Prompt the exception message to the appropriate actors or user, and passes the control to them.

5 Related Work

There are many works for describing business processes for Web Services applications. Some of these have been created as part of workflow-management [11-12] or business process offerings [13] by both commercial vendors or open source community. The dominant approach represents a workflow as a graph that controls the flow of control between business steps. These graphical drag and drop tools let users build workflow applications without having to learn underlying complex notations, but their lack of flexibility restricts their ability to meet complex business requirements.

Only a few works focus on the workflow engine. Cao et al. [14] proposed a distributed reconﬁgurable grid workflow engine in the grid environment. Cervantes et al. [15] presented a software product line architecture where applications are assembled by installing a set of plugins on a common software base. The software base embeds a lightweight workflow engine that guides the flow of control and data of the application. Jang et al. [16] proposed an event-driven workflow engine for service-based systems where the Event-Activity notion was used. Ngeow et al. [17] proposed a web-based device workflow management engine that supports seamless control and coordination of hardware devices. Most of these workflow engine lack of flexibility to support adaptive response, and do not provide solutions to the problem of exception handling.

6 Conclusion

In this paper, we propose an adaptive workflow engine based on web services and multiagent technologies to support dynamic interpreting, configuration, monitoring and executing workflows. We explain the functionality of the identified modules of the engine, and discuss the design and implementation consideration through a case study of the scientific research management system of East China Institute of Technology. The main features of this engine are event-driven process execution, a unified interaction interface with external environments and exception handling mechanism.

The next step of our work will be to integrate self-managing behavior and performance evaluation and adapt the system’s configuration accordingly in our engine.
Acknowledgement

The authors wish to thank the support by the National High-Tech Research and Development Plan of China (863 Program) under Grant No. 2007AA06Z111; the Scientific Research Plan Projects of Education Department of Jiangxi province of China under Grant No. GJJ09263.

References