An Ad-Hoc Workflow System Architecture Based on Mobile Agents and Rule-Based Processing

Jie Meng, Sumi Helal and Stanley Su
Computer and Information Science and Engineering
University of Florida, Gainesville, FL 32611
{jmeng,helal,sul}@cise.ufl.edu

Abstract

As e-commerce permeates the globe, hundreds of thousand of web-based services are being created and provided to end users and organizations. Even though these services are developed independently, they are nevertheless loosely coupled by virtue of relevance. Unfortunately, there is not an easy way to tie these related services together to fulfill an e-business opportunity (an e-business process). In this position paper, we argue that the ad-hoc workflow and mobile agent technologies can enable e-business processes over loosely coupled web-based services. We present our ideas in the context of an order processing application in supply chains. We also present the architecture and implementation of a simple mobile-agent-based ad-hoc workflow system.

***Keywords**: Ad-hoc workflow, Mobile agents, e-services, service brokering.*

1 Introduction

Workflow management (WFM) is an enabling technology for the automation and integration of process-oriented tasks. In a WFM system, a workflow model (workflow type) is predefined for each business process. A workflow model consists of a number of activities and a set of conditional transitions that link these activities together. No assumption is made about the location of activities that could potentially be spread across the network, especially in inter-organizational workflow systems.

In this paper, we present a design and an implementation of an ad-hoc workflow system based on the mobile agent technology. Unlike traditional WFM systems [2], in ad-hoc workflow, there is no predefined workflow model for the business process. The business process is built *ad-hocly* as it is enacted using distributed services (activities). Business rules, which capture invariant inter-service logic, are associated with the services and are used to determine the next activities to be performed on behalf of the business process. The main advantage of ad-hoc workflow systems is that the business process can be built dynamically and flexibly. Another advantage is the ability of ad-hoc workflow to cope with dynamic changes in business rules and service definitions, implementations, and/or locations. The service providers taking part in the ad-hoc workflow are not required to be dedicated services. They could be cooperative services in an organization's Intranet, or could very well be loosely coupled e-commerce services.

With the emergence of business to business e-commerce, a new breed of business processes that relies on multiple independent web-based services is created. We call these processes "e-business processes". We argue that ad-hoc workflow modeling is most appropriate for e-business processes. We further argue that mobile agents [3][7][8] are very well suited for enacting ad-hoc workflow applications across loosely-coupled web-based services. The critical aspect that makes mobile agents suitable for this kind of workflow is their ability to move ad-hocly to local services. Without mobile agents, such web-based services would have to be integrated and designed as distributed cooperative services that are able to communicate with a remote workflow engine. With mobile agents, this requirement is simply waived.

Another important advantage for using mobile agents in ad-hoc workflow systems based on web services is the suitability of asynchronous communication in this environment. Web-based services are usually not real-time tasks and often involve human operators who connect to the Internet only sporadically. Synchronous communication such

as RMI, CORBA, TCP/IP, or even HTTP will not be suitable for enacting this kind of ad-hoc workflow.

An important and synergistic technology to mobile agents is service brokering. Ad-hoc workflow systems can benefit greatly by using a broker to dynamically bind services to proper service providers. Such late binding is key in enabling the flexibility achieved by ad-hoc workflow. It also allows the workflow to be enacted over alternative services in case multiple providers for the same service are discovered to be available. The brokering service is therefore critical to determining the transitions for the he ad-hoc workflow.

In this paper, we describe how ad-hoc workflow, mobile-agents and service brokering participate synergistically in realizing the e-business processes. The rest of this paper is organized as follows. In section 2, we present our ad-hoc workflow system design, followed by an example scenario drawn from an order processing application in Supply Chains. In section 3, taking the order processing application as an example, we illustrate the design of the ad-hoc workflow system. In section 4, we present our implementation, and finally section 5 concludes the paper.

2 Mobile Agents and Ad-hoc Workflow

Mobile agent [1][3][7][8] is an emerging technology attracting interest from the fields of distributed systems, mobile computing, electronic commerce, among others. The mobile agent technology has the potential to provide a convenient, efficient and robust programming paradigm for distributed applications, even when partially connected computers are involved. A mobile agent is a process that can migrate from one computer to another during its execution. Mobile agents can also communicate with each other, clone, merge, and coordinate their computations. Mobile agents are autonomous agents in the sense that they control their relocation behavior in pursuit of the goals with which they are tasked. These properties are additional reasons that make mobile agents a good candidate to be used in a loosely coupled network environment [5][6][9].

To make use of the services distributed across the network, for each ad-hoc workflow instance, a workflow mobile agent is created. This agent goes to different locations where services reside to invoke the services locally. The arrival of the workflow mobile agent will trigger a rule to invoke the local services. Once a service (or a collection of local services) is (are) performed successfully, another business rule is fired to determine the next service to be executed. In this case, the business rule works as a run-time generator of a conditional transition to handle the transition from one activity to another. The workflow mobile agent then contacts a broker to get the provider of the next service and its location, and then moves to that location to execute the next service. The business rule engine at the service side, together with the workflow mobile agent constitutes the workflow engine for an ad-hoc workflow. We name the business rule engine as the Service Side Ad-hoc Workflow (SSAWF), and the workflow mobile agent as the Mobile Agent Side Workflow (MASAWF).

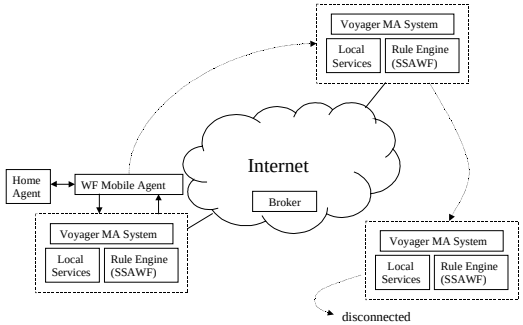


Figure 1. General Architecture of Mobile Agent Workflow Software

To monitor and manage the ad-hoc workflow easily, a static home agent is created for each workflow instance. The workflow mobile agent can also communicate with the static home agent during the execution, which allows the user to interact with the workflow instance. The general system architecture is shown in Figure 1. A workflow mobile agent originates at some node, moves to other services based on local rules and the help of a broker, and finally returns to its originating

node. Each network node that participates in the ad-hoc workflow processing runs in the ad-hoc workflow management environment, which consists of the layers of software shown in Figure 2. These includes the Mobile Agent environment (we use ObjectSpace's Voyager mobile agents), the local services (we use Java classes whose interfaces are advertised to the broker), and a local rule engine (we currently use a simplified rule engine).

Our architecture supports mobile users, where a service provider node is allowed to be disconnected from the network. After the service has been finished and the node reconnected to the network, the workflow mobile agent returns back to the network and moves to the location of the next service.

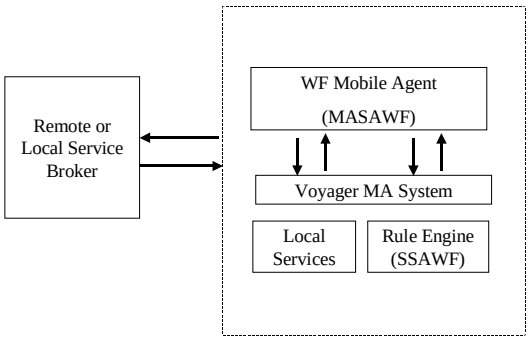


Figure 2. Ad-hoc Workflow Management Environment

3 An Order Processing Application in a Supply Chain

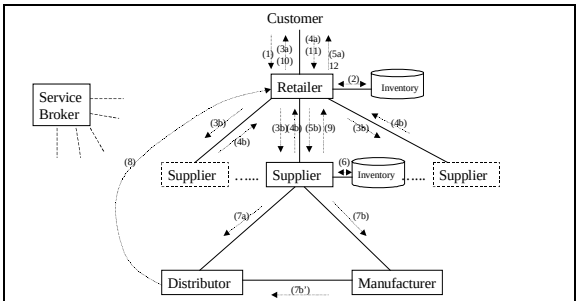
In this paper, we use a supply chain scenario as an example application to demonstrate the mobile-agent-based and rule-based ad-hoc workflow processing. In a retailer company, an order process is initiated by a customer request. When a customer puts in an order, the retailer first checks the local inventory. If there are enough quantity of the product at the local site, the retailer will ship the product and the invoice to the customer. A receipt will be send to the customer after the payment. Otherwise, the retailer will request offers from several suppliers and will choose the best supplier based on the received offers.

This order process can be accomplished using an ad-hoc workflow. Suppose the service providers are distributed across the network, including the retailers, suppliers, distributors, and manufacturers. Each of the service providers has certain business rules that determine when a specific service should be executed, and which service need to be executed next in a business scenario. When an order process begins, a workflow mobile agent is created for this order process. At the first step, the workflow agent invokes the inventory manager at the retailer site to check whether the requested item is available at the local site. Then, a rule will be triggered that defines the service transition and the transition conditions. In this example, if there are enough products locally, the next service is to ship the product and send the invoice to the customer. A receipt will be sent to the custom upon receiving the payment. Otherwise, the workflow agent will issue Request-for-Quote (RFQ) to suppliers who sell the product and receive quotes from them. It will then migrate to one of them (e.g., the one that gave the best quote) with a purchase order and interact with the supplier locally to see if the supplier is qualified (based on some supplier information) and is willing to process the purchase order (a local decision). If not, the workflow mobile agent will go to another candidate supplier.

At the supplier site, after making the joint decision with the mobile agent that the supplier is qualified and can process the purchase order, the supplier will determine if it can provide the quantity of the product ordered. If it can, it responds to the workflow mobile agent positively. A supplier rule is activated to direct the mobile agent to migrate to one of the supplier's distributors and use its service to deliver the product to the retailer. Otherwise, it directs the mobile agent to its manufacturer and uses its services to manufacture the right quantity of the product and deliver the product to the supplier's distributor. The mobile agent returns to the distributor to activate its service to deliver the product to the retailer and then to the supplier and use its service to issue invoice to the retailer. It then returns to the retailer to ship the product and invoice to the customer and receive payment from the customer.

In the above ad-hoc workflow, there is no explicit business process model. Instead, an

implicit model exists in the business rules on the service provider sites. The scenario of the business process is shown in Figure 3, in which the links connecting activities are modeled by business rules. The advantage of using business rules instead of a static process model is that rules can be dynamically changed to account for and reflect the changing nature of business. In our implementation, business rules are translated into Java classes. A class reloading feature is provided to allow new rule classes to be dynamically loaded to replace the old rule classes when business rules are changed.



Note: (1) Retailer gets an order from the customer. (2) Retailer checks the inventory. (3a) Retailer ships the products and the invoice to the customer. (4a)Customer makes payment. (3b) Retailer calls for quota from suppliers. (4b) Suppliers return quote. (5b) Retailer puts an order to a supplier. (6) Suppler checks inventory. (7a) Supplier asks distributor to ship the products to the retailer. (7b) Supplier asks the manufacturer to make the products. (7b') Manufacturer ships the products to distributor. (8) Distributor ship the products to retailer. (9) Supplier sends invoice to the retailer. (10) Retailer ships the products and invoice to the customer. (11) Customer makes payment. (12) Retailer sends out the receipt.

Figure 3. Implicit Workflow Model in the Supply Chain Scenario

4 Ad-hoc Workflow: Services and Rules

We chose Voyager as the mobile agent platform for the ad-hoc workflow system. Voyager is the first platform to seamlessly integrate traditional distributed computing technology such as CORBA, DCOM and RMI, with the agent technology, as it was designed from the ground up to support object mobility. It is the product of ObjectSpace Corp. The ObjectSpace's philosophy is that an agent is simply a special kind of object that has the ability to move itself and continue to execute as it moves; it should otherwise behave exactly like any other distributed object. The agents would move when appropriate in order to avoid low-bandwidth connections. Voyager also provides powerful messaging system, that allows distributed objects and mobile agents to communicate seamlessly regardless of their location. It was a natural choice for us since distributed services are to be provided by distributed objects according to our system design.

In our scenario, there are four distributed organizations taking part in the order processing ad-hoc workflow. The four organizations are represented as four distributed Voyager objects. The services provided by each distributed organization are represented by their APIs. A partial list of services provided by these four organizations is given below:

1. Retailer
   - *Submit Order*: Customer uses this service to submit an order.
   - *Select the Best Supplier*: Select the supplier that offers the best price quote.
   - *Ship Product and Send Invoice:* Ship product to the customer and ask the customer to make the payment
   - *Receive Payment and Send Receipt:* Receive payment from the customer and send receipt as the acknowledgement of the receipt of the payment.
2. Supplier
   - *Send Price Quote*: This service is used to send the price quote to the agent when the supplier gets the RFQ.
   - *Process Order*: This service is called when the workflow mobile agent travels to the supplier.
   - *Send Invoice*: Send invoice to the retailer.
   - *Receive Payment and Send Receipt*: Receive payment from the retailer and send the receipt.
3. Distributor
   - *Receive Shipment Request*: The distributor receives a shipment request from the supplier.
   - *Receive Product*: Receive product from the manufacturer.
4. Manufacturer
   - *Receive Order:* Receive order from the supplier.
   - *Manufacture Product*: The manufacturer produces the products as ordered by the supplier.
   - *Send products:* Send manufactured product to the supplier's distributor.

Although the services defined above at different sites do not fully capture the real world business process, they are sufficient to illustrate the proposed ad-hoc workflow concept and implementation strategy. For each ad-hoc workflow instance, a workflow mobile agent is created. Since mobile agent can move autonomously during execution, this workflow mobile agent travels to the participating service sites and invokes the heterogeneous services locally there. In this sense, the workflow mobile agent bridges the loosely coupled services together to fulfill the desired business process. Since the rule engine at different service sites are used to request local services and to handle the activity transitions in the ad-hoc workflow, we consider the workflow mobile agent and the rule engine together to be the workflow engine.

Business rules are divided between the mobile agent and the local services. The former rules are part of the mobile agent logic and itinerary. For instance, in our scenario, deciding if a supplier is appropriate (e.g., good credit, best prices, etc.) is governed by business rules in the mobile agent. Service rules, on the other hand, are created by the service providers. They are used to trigger the appropriate service(s) upon the arrival of a mobile agent, based on local as well as mobile-agent-related conditions. For example, in the supply chain scenario, receiving an order from a government agency by the supplier would trigger a service that is slightly different from the service usually used with ordinary customers. This could be to fulfill certain governmental procedures and guarantees. Service rules are also critical in determining run-time conditional transitions of the ad-hoc workflow. For instance, upon successfully invoking a *process order* service at a supplier, a service business rule should take the ad-hoc workflow into one of the distributors of that supplier.

We use simple ECAA rules to handle the service requests and the service transitions. For an ECAA rule, there are four parts:

- *Event*: The event that triggers this rule.
- *Condition*: The condition that needs to be checked before the action is performed.
- *Action*: The action to be performed when the condition is true.
- *AltAction*: The action to be performed when the condition is false.

There are several events in the ad-hoc workflow that can trigger the ECAA business rules for the e-business process. Some of these events are described as follows:

- *Event: The arrival of Mobile Agent*: when a mobile agent arrives at a site, a rule will be triggered to invoke the most appropriate service for that agent. Here the simple logic of invoking the most general "entry" service upon the arrival of a new request (the mobile agent) is used. The condition part of the ECAA rule can be used to check additional conditions for executing the service, such as, security or capability conditions. An example of this type of rules at a supplier site is:
  Event: Arrival of mobile agent at supplier site.
  Condition: Credit history of the retailer is good.
  Action: Invoke the service “Process Order”; pass parameters carried by mobile agent.
  AltAction: Reject.

- *Event: Success of executing service*: Upon the successful completion of the service, a transition rule will be triggered to determine the next service to be executed (the ad-hoc workflow transition). An example of this type of rule at a supplier site is:
  Event: The success of executing “ProcessOrder”
  Condition: True.
  Action: Next service is “InitiateShipment”, service provider is “Distributor”.
  AltAction: null.

- *Event: Failure of executing service*: This type of events will trigger the rules to handle exceptions in the execution of services. For example:
  Event: The failure of executing “ProcessOrder”

Condition: If there are other candidate suppliers.
Action: Go to next candidate supplier.
AltAction: Go back to home agent and notify the user.

In this mobile-agent-based workflow system, for each workflow instance, we create a static home agent that sits on the home site to monitor the workflow execution. As long as the workflow mobile agent conducts a service, it will communicate with the home agent so that the home agent knows the status of the workflow. In addition to monitoring, the workflow user can interact with the workflow agent in some situations and get involved in determining the workflow transitions. The user can also change some of the workflow business rules in the mobile agent dynamically.

The broker is used by a workflow instance to choose a proper service provider from the available provider candidates for a specific service. For example, the event "Success of Executing Service" at a supplier site triggers the action that the next service to execute is ”Receive Shipment Request” at a distributor. The broker role here is to find the location of the distributor or to choose one if multiple distributors are found available. Brokering-based processing contributes to scalable workflow processing. Whenever a new service (company) desires to participate and join the workflow, it only needs to register with the broker. We should mention here that some of the business rules could provide complete and exact specifications of the next services to execute (including location, and APIs), in which case, the service broker is not used.

5 Implementation Status

We have built a prototype system based on the design presented in the previous sections. We choose Voyager as the mobile agent platform. For simplicity, we implemented each organization server as a Voyager server. For each organization, we provided a Graphic User Interface so that the services and rules provided by the organization can be seen and browsed.

In the current implementation, instead of using a real ECA rule engine, we simulated the rule engine as another Voyager server. Whenever an event occurs, we call the corresponding method in the Rule Server to simulate the rule execution. A real ECA rule engine will replace the Voyager rule simulator.

One execution snapshot can be seen in Figure 4. In this figure, four GUIs represent the Retailer, Supplier1, Supplier2, and the Distributor, respectively. In the "Service Provided" area, the services provided by the corresponding organization are displayed. In the "Logging" area, the user can see the execution status of the workflow. In the GUI for the Retailer, there is another area called "Sites visited". This area is used by the end-user of this workflow to monitor the workflow execution. We call this area the monitor. It serves as the Home Agent.

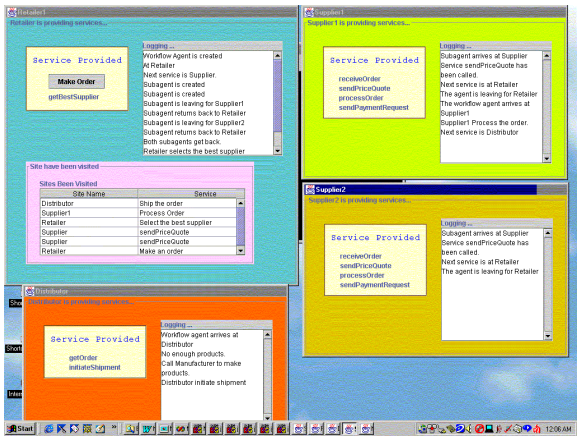


Figure 4. Services in an Ad-Hoc Workflow Instance

6 Related Work

Aglet [3] is one of the first Java-based mobile agent systems, which was created by IBM Corporation. Voyager [8] is a platform to seamlessly integrate traditional distributed computing technology such as CORBA, DCOM, RMI, with agent technology. It is the product of ObjectSpace. Concordia [7] is a Java-based mobile-agent system that has a strong focus on security and reliability. In addition, there are several other mobile agent

systems contributed by universities and can be found in the public domain.

There are a few research projects that are related to mobile agent-based workflow. Software agents for workflow support is presented in [11]. COSMOS is a joint project by Humburg University (Germany), Oracle (UK). Its goal is to provide an integrated support platform for Internet business transactions. In this project, a mobile agent-based inter-organizational workflow management system is proposed and implemented [5][6]. In the mobile-agent-based workflow system proposed in [10], control is divided among three entities: the Work Service Broker, Agent, and Place. They use Agent Pool to avoid the potential processing congestion. A virtual enterprise model (VE) by means of mobile agents has been proposed in [9]. Their idea is to use a mobile agent called *adlet* to establish VEs that involves advertising, negotiating and exchanging control information and data as well as its management. Finally, the work in [4] states an mobile agent approach to lightweight process workflow.

7 Conclusion

In this paper, we presented an architecture of an ad-hoc workflow system that is suitable for business process defined over loosely-coupled web services. An ad-hoc workflow system has several benefits over traditional workflow systems. Since it is created ad-hocly, through business rules and service brokering, no predefined workflow model is needed. This allows *workflow users* to specify minimum service specifications by relying on the unknown business rules and brokering services. The critical observation here is that ad-hoc workflow reduces to a great extent the requirement for a workflow designer.

The combined use of mobile agents, business rules, and brokering service makes the ad-hoc workflow amenable to dynamic change. It also provides the needed flexibility that would allow different, non-interoperable service providers to participate in a workflow instance autonomously, and without totally understanding the workflow technology. With the emergence of e-business, and with the ever-increasing ubiquity of the Internet, the case will be made stronger and stronger for the use of ad-hoc workflow systems, as a more viable alternative to traditional workflow, in the context of web-based services.

References

[1] C. G. Harrison, D.M. Chess, and A. Kershenbaum, *Mobile Agents: Are They a Good Idea?* Technical report, IBM Research Division - T.J. Watson Research Center, March 1995.

[2] A. Cichocki, A. Helal, M. Rusinkiewicz, and D. Woelk, *Workflow and Process Automation: Concepts and Technology*, Kluwer Academic Publishers, ISBN 0-7923-8099-1. 1997.

[3] Aglets webpage http://www.tri.ibm.co.jp/aglets.

[4] G. Kaiser, A. Stone, and S. Dossick, *"A Mobile Agent Approach to Lightweight Process Workflow,"* Proc. of International Process Technology Workshop, Sept. 99.

[5] M. Merz, B. Liberman, W. Lamersdorf, *"Mobile Agents to Support inter-organizational Workflow Management"*, http://vsys-www.informatik.uni-hamburg.de/publications/.

[6] M. Merz, W. Lamerdorf, *"Crossing Organizational Boundaries with Mobile Agents in Electronic Service Market,"* International Journal on Computer-Aided Engineering, Special Issue on Mobile Agents, 1998. (http://vsys-www.informatik.uni-hamburg.de/publications/)

[7] Mitsubishi Elc, http://www.meitca.com/HSL/.

[8] Object Space, Voyager ORB, http://www.objectspace.com.

[9] P. Chrysanthis, S. Banerjee, S. Chang, *"Establishing Virtual Enterprise by means of Mobile Agents,"* Proceedings of the Research Issues in Data Engineering Workshop, Sydney, Australia, March 1999.

[10] S. Foster, D. Moore, M. Flester, B. Nebesh, *"Control and Management in a Mobile Agent Workflow Architecture,"* Proceedings of Agents'99, Seatle, May 1999.

[11] Z. Maamar, N. Troudi, and P. Rostal, "Software Agents for Workflow Support," The Journal of Conceptual Modeling. February 2000.