AUTOMATIC INTEGRATION OF INTER-ENTERPRISE PROCESSES WITH HIERARCHICAL BROKER FRAMEWORK

Shun-Fa Chang , Li-Chen Fu, and Ming-Yu Tsai
Department of Computer Science and Information Engineering,
National Taiwan University, Taipei, Taiwan, R.O.C.

Keywords: B2B E-commerce, workflow, Inter-Enterprise, Broker, Adapter

Abstract: In recent years, the manufacturing technologies are more and more complex. Almost all production processes need cooperation among multiple enterprises. It is true that today's manufacturing process is a complex workflow forming a supply chain. Each enterprise provides their services to accomplish professional processes. With the growth of Internet usage, there are more and more services able to be processed on the web. Web-service is one of the applications on Internet and it can help enterprises cooperate with one another in their services easily. In this paper, we propose a Hierarchical Broker Framework to provide an advanced broker function for enterprises' cooperation. In this framework, we can classify all services to keep searching easier, to present the relations between two enterprises more flexibly, to match buyers and sellers more precisely, and to cut down broker's loading. On the ride of an enterprise client, we do not have to modify any existing enterprise architecture. Beside, we will also design an adapter to connect the broker server and the existing enterprises. By these designs, we try to find an automatic way to integrate these enterprise processes to improve efficiency and reduce their transaction overheads.

1 INTRODUCTION

With the progress of computers, information technology and Internet, people find a new way to undertake business by electronic media that reduce the time barrier and distance between sellers and buyers. For enterprises, they need to respond to the demands of customers quickly to gain an advantageous position as global competition and market pressures rise. They are seeking a new way to decrease cycle time and to make operation more productive. So far, there are many services that can improve workflow efficiently on the internet. How we can automatically and efficiently integrate these services in the workflow is an important issue that we need to cope with. In this paper, we propose a new framework, Hierarchical Broker Framework, to deal with this problem efficiently and elegantly.

Hierarchical Broker Framework is applied to replace the existing marketplace and to add efficient matching mechanism to it. In this framework, all services will be clearly classified, and thus customers can find their required services more easily. The broker provides useful matching mechanism to help buyers or sellers to find the suitable partners very conveniently. By the automating function, the integration and automation of business processes will be simpler and more efficient.

With the integration of inter-enterprise processes, enterprises will have no difficulties to outsource some processes to complement their own weakness and to reduce some unnecessary overhead. This framework allows enterprises to focus on their core technology while relying on partners to perform other critical activities. In order to have all enterprises easily join this Hierarchical Broker Framework, we design an Adapter Agent to help every one of them to be free of removing legacy operating systems.

The remainder of the paper is structured as follows. Section 2 summarizes the relation work. In section 3, we propose a new business framework, called the Hierarchical Broker Framework, to replace the existing marketplace. Based on this framework, we can successfully establish an Intelligent Broker for matching buyers and sellers and an Adapter Agent for connecting the new framework and the legacy operating system in every joined enterprise. We will introduce how these different enterprise parties shall communicate with one another in section 4. The implementation of this

Chang S., Fu L. and Tsai M. (2004).
AUTOMATIC INTEGRATION OF INTER-ENTERPRISE PROCESSES WITH HIERARCHICAL BROKER FRAMEWORK.
In *Proceedings of the Sixth International Conference on Enterprise Information Systems*, pages 62-69
DOI: 10.5220/0002613500620069
Copyright © SciTePress

framework is discussed in section 5. Finally, section 6 makes a conclusion of the paper.

2 RELATED WORKS

The term “workflow” originated in the mid-1980s and became popular in the early 90’s. Since then, many service providers have offered general-purpose workflow management systems (IBM, WebSpere MQ Workflow). We used these workflows in two ways. One is to lead ERP (Enterprise Resource Planning) systems to adopt a workflow component. Another one is to include workflow functionality in the enterprise application.

WISE (Workflow based Internet SErvices) is a project conducted at Swiss Federal Institute of Technology (Alonso et al., 1999). The goals of WISE are to develop and deploy the software infrastructure necessary to support business to business electronic commerce in the form of virtual enterprises. The idea is to combine the tools and services of different companies as building blocks of a higher level system in which a process acts as the blueprint for control and data flow within the virtual enterprise. From this idea, the final goal is to build the basic support for an Internet trading community where enterprises can join their services to provide added value processes.

CrossFlow is a European research project for supporting cross-organizational workflow management in virtual enterprises (Grefen et al., 1999). Its goal is to develop and implement a mechanism for connecting WfMS and other WfMS-Like systems of different organizations in cross-organization workflows and electronic commerce settings. **Crossflow** defines a service-oriented model for cross-organizational workflows. In their service-oriented model, a service provider of each service can be either an internal resource (internal service) or an external organization (external service). For an external service, service selection at run-time will be based on the QoS parameters given in service specifications.

Sangam is a universal interoperation protocol for e-service broking communities using private UDDI nodes (Helal et al., 2002) (Jagatheesan et al., 2003). It aims to achieve more relevant and interoperable brokering of e-services in a highly scalable and dynamic environment of e-services.

3 HIERARCHICAL BROKER FRAMEWORK

3.1 Architecture and Technology

Hierarchical Broker Framework is a framework which connects every service and marketplace on the internet. The main idea of this framework comes from the “domain name mechanism” on the internet and the “inheritance concept” on object-oriented programming. The former can help us create this framework more uniformly, whereas the latter can keep sellers to enter this framework with little efforts.

The traditional marketplace provides a place where sellers can publish their services or products, and buyers can search in this online catalog for those services which they need. The marketplace owner needs to maintain this online catalog. It has many different domains and different presenting methods in each marketplace, and it may be difficult for buyers to find the desired services, thus the matching mechanism is very important.

The main idea of Hierarchical Broker Framework is to connect each special marketplace by a classified framework. There are only two kinds of components in this framework, Intelligent Broker and Adapter Agent. Each Intelligent Broker can independently be extended to a sub-marketplace, and the Adapter Agent is a member in this sub-marketplace.

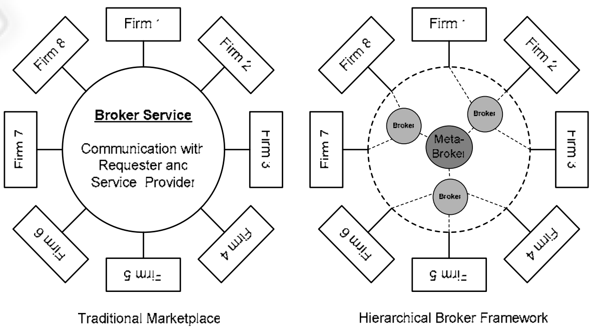

Figure 1: Traditional Marketplace & Hierarchical Broker Frameworks

In this framework, an Intelligent Broker is mapped to a sub-marketplace, whereas an Adapter Agent is mapped to a service. In Figure 2, we can see how this framework integrates these specific Intelligent Brokers. By this framework, not only services but also Intelligent Brokers have a unique place in this classified framework. The unique place stands for the position of this service and its relation with other services in this framework.

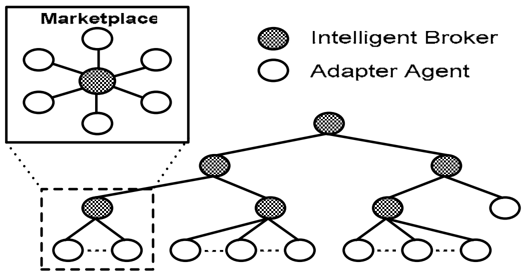

Figure 2: Hierarchical Broker Framework

In Figure 3, it presents another aspect in this framework. Components in the area surrounded by the dotted line use the same standard to communicate. One broker server can serve service providers and other broker servers only if all of them follow the same standard.

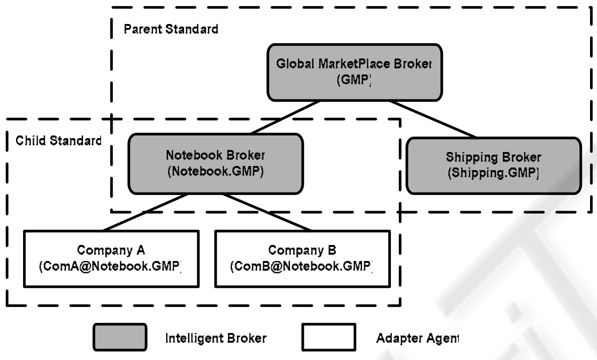
Parent Standard
Global MarketPlace Broker (GMP)
Child Standard
Notebook Broker (Notebook.GMP)
Shipping Broker (Shipping.GMP)
Company A (ComA@Notebook.GMP)
Company B (ComB@Notebook.GMP)
Intelligent Broker
Adapter Agent

Figure 3: Part of Hierarchical Broker Framework

Each Intelligent Broker must build a special marketplace into it, and hence an Intelligent Broker stands for one specific marketplace. After this broker is created, service providers can publish their services on it. Each broker server has another mission that it is responsible to manage all services and sub-brokers under it. It must check regularly if the services remain available or not. This maintenance will require large efforts in a one-tier traditional marketplace, but they will be handled by each specific broker in the Hierarchical Broker Framework.

This design keeps each standard more stateless. The child Broker Server Standard is able to flexibly describe the specific domain in detail, and the Parent Broker Server Standard is less domain-dependent and can be generally used in any domain.

3.1.1 Broker Standard Inheritance Relation

Although in Hierarchical Broker Framework each tier may have different standards, we use inheritance concept to create the relations. By this idea, the distributed broker standard will have some norms to normalize it. This idea is from OOP (object-oriented programming). The Child Broker Server Standard will inherit from the Parent Broker Server Standard. In this way, we can define a new child standard easily by referring to the parent-broker standard.

3.1.2 Service Naming Mechanism

The other concept used in the Hierarchical Broker Framework is the domain name mechanism on the internet. This mechanism arises because most people cannot remember 32-bit IP address and a mapping from the pure-digits address to natural language phrases is very helpful for us to memorize. Domain name mechanism uses a tree structure to describe the location on the internet. When we name a domain, we try to make this name more representative. We will name all Intelligent Brokers in this framework so that each service will have a unique name. This unique name mechanism will help workflow designers to design the business workflow more easily.

3.2 Definition of Data Unit

In this paper, we design several data units to help these components to communicate with one another. There are three kinds of data units: order, process, and task.

The order is a fundamental unit that companies will use to execute business actions. It is a unit used to ask a provider to offer a complete service or product. The process unit is used to dispatch this action according to whether it is going to be finished in a local place or needs to invoke a service in other places. The task is an elementary action in this framework, which connects this framework and the legacy operating system within an enterprise. One task is mapped to one traditional process in the legacy operating system.

In Figure 4, we see the relations among orders, processes, and tasks. It is clear that workflow is constructed by tasks. These tasks directly communicate with the legacy operating system, maybe a manufacturing system. A sequence of tasks can form a meaningful process, and a sequence of processes stands for a complete workflow of a product or a complete service. Figure 3.4 Workflow data unit

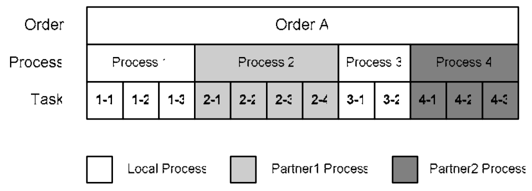

Figure 4: Workflow data unit

3.3 Basic Components Architecture

3.3.1 Intelligent Broker Node Architecture

The Intelligent Broker plays the role as a manager in the framework. It is used to manage every service under it. Besides, it manages sub-brokers under it, too. A broker server has the following functions:

- Manage provider registry system;
- Store each service and maintain the service table;
- Store each template and maintain the template table;
- Provide the interface for inquiring services;
- Provide the matching mechanism;
- Provide a function to communicate with upper and lower brokers.

All broker architectures look like that is shown in the following figure:

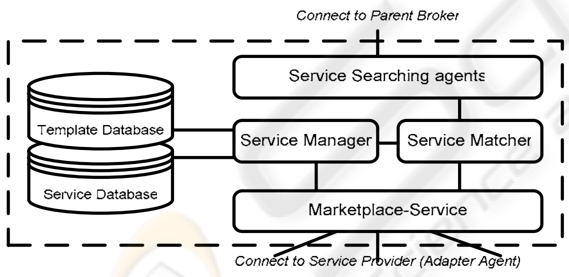

Figure 5: Intelligent Broker Architecture

Marketplace-Service provides a basic marketplace function. It can receive members' requests of registrations and it can query members' services actively. There are three kinds of actions that a Marketplace-Service will receive from the Adapter Agent. They are "register" action, "match" action and "order" action. **Service Manager Agent** is a bridge between the database and the main program in the Intelligent Broker. All data including broker information, service and template information, service provider information, and child-broker information will be loaded by the Service Manager Agent. Service Manager Agent is an information center in the broker. **Service Matching Agent** is the control center in the Intelligent Broker. All matching and ordering action will pass through it. The Service Matching Agent function is used to match the providers and buyers. **Template & Service Database** explicitly store all services and templates that belong to this broker. Template is a kind of service specification, which defines standard input, output and all variables. A real process set is provided by a provider and follows a template specification. So, one template may have many services, and one service must have one service provider associated with it. **Searching Agent** is a component which is responsible for communications with the Parent broker. It is a finder because it finds other brokers in different domains. Sometimes we need this component when one broker can not find another broker and this target broker is not under the inquired broker.

3.3.2 Adapter Agent Node Architecture

The **Hierarchical broker framework** is used to connect all services on the Internet and to assist to coordinate the service partners. In fact, each firm generally has its own legacy operating system, so it is a challenge to pass information to other systems. Legacy operating systems generally invoke older mainframe and minicomputer systems to manage the key business processes in a firm covering a variety of functional areas from manufacturing, logistics, finance, and human resource, etc. Generally, it is very costly and time-consuming to convert these old systems to new systems.

The Adapter Agent has two kinds of interfaces, one for the Intelligent Broker and the other for legacy operating systems. The Adapter Agent rule translates all actions between the Intelligent Broker and the legacy operating system. There are many necessary functions in the Hierarchical broker framework. If the legacy operating system fails to support some of those functions, our Adapter Agent must provide it. The goal of the Adapter Agent is to provide a complete interface to let enterprises successfully join this framework.

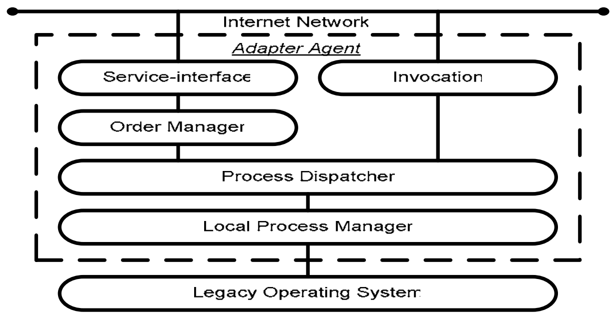

Figure 6: Adapter Agent Architecture

We decompose this agent into many components. The **Service-Interface** provides an interface that allows outside users to inquire services, make orders, and trace the order's progress. It must follow communication standard in the framework. After receiving a request, it translates the request and then invokes the **Order Manager** inside the Adapter Agent.

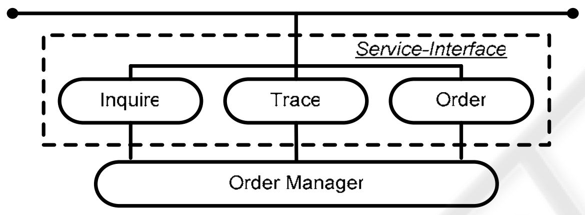

Figure 7: Service-Interface

The **Order Manager** is a component to manage all orders received from the Service-Interface. When the order manager gets a new order, it will create a unique order name and store the order in the order book. It will also send the order to the Process Dispatcher to classify and to dispatch each work.

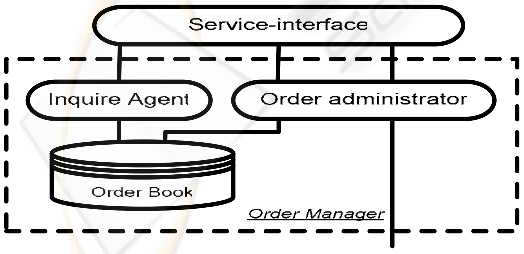

Figure 8: Order Manager

The **Process Dispatcher** receives order from the order manager, and the main mission is process definition and process dispatching. When an order enters this dispatcher, the first job for Process Dispatcher is to define all processes in the order. In other words, it will decompose the order to many processes.

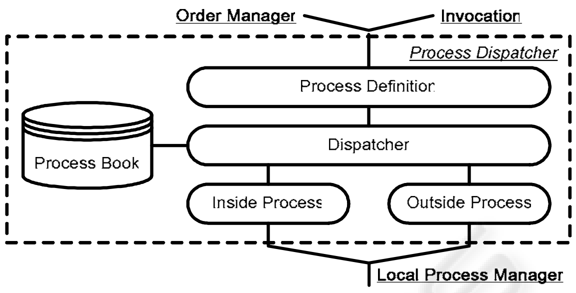

Figure 9: Process Dispatcher

The **Local Process Manager** is the only component which actually invokes legacy operating systems. It is used to connect the Adapter Agent and the **legacy operating systems**. Because there are many different legacy systems, the local process manager must be designed case by case.

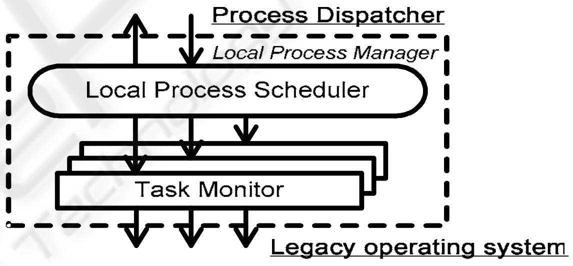

Figure 10: Local Process Manager

4 INTEGRATION OF ENTERPRISE PROCESSES

There are three kinds of elementary actions in this framework. In Figure 11, it can be seen that all these actions are communications between the **Broker** and the **Enterprise**.

(1) Publish / unPublish: Service providers advertise (publish) their e-services to one or more broker servers. They may also unpublish the advertisements of services from the registry.

(2) Match request: A buyer needs a seller. He can set some requirements and send it to the broker to request a matching list of suitable sellers.

(3) Match result: The broker will return a suitable result for the buyer.

(4) Invoke: After matching the suitable provider and verifying its quality, this service will be invoked.

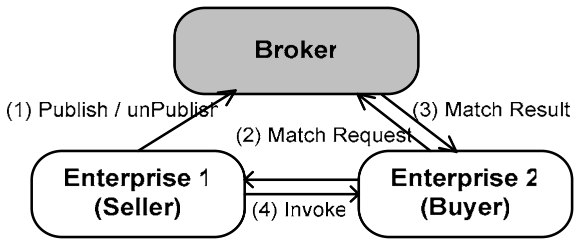

Figure 11: Elementary Broker Mechanism

Generally, an enterprise can publish its services on the brokers, and these brokers will collect these services and make an online catalog. If a buyer, not only an individual customer but also an enterprise, needs some services or products, he can ask the broker to search for a suitable enterprise that can provide the service. After matching, the buyer can invoke these services that he needs.

Actually, these elementary actions can construct more complex processes to present the real situation. These buyer-seller relations connect enterprises to be a chain. Production of a product may need cooperate with many company's processes, and accomplishment of one company's process may also need coordination with more other companies' processes. Under this concept, each enterprise will play more than one role given one order simultaneously.

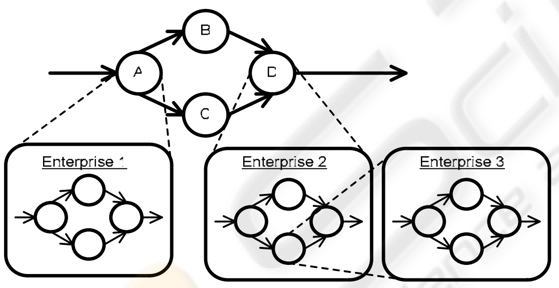

Figure 12: Integrated Inter-Enterprise Process

Figure 12, we aims to explain relations among different enterprises in this integrated inter-enterprise process from the process perspective. We know that each complete service involves many enterprises, and the Intelligent Broker and the Adapter Agent will help us to create this coordination process more easily.

4.1 Publication Mechanism

The publication mechanism gives the enterprise the ability to publish their services on the internet to wait for being invoked. There are two kinds of information that will be published, templates and services.

In the Intelligent Broker, some service providers will cooperate to design a standard template for some designated services. It is a general specification, and any service should follow so that it will be easier to be used by buyers. When the company has a new service, it will check if there is a suitable template to use. If the Intelligent Broker does not have a suitable template, the provider must create a new one. The creating action is show below.

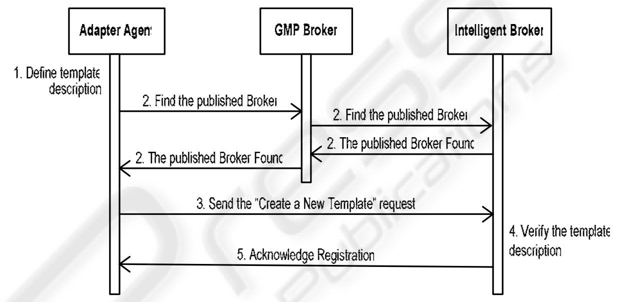
Adapter Agent
GMP Broker
Intelligent Broker
1. Define template description
2. Find the published Broker
2. Find the published Broker
2. The published Broker Found
2. The published Broker Found
3. Send the "Create a New Template" request
4. Verify the template description
5. Acknowledge Registration

Figure 13: Template Publishing Action

In order to keep buyers able to find the service, the enterprise must publish their services on the broker. Before publishing services, the enterprise must choose a suitable broker and a suitable template to store their services. These are the steps that a provider needs to take to publish a service.

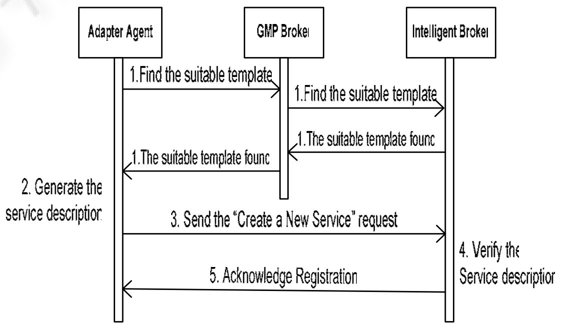
Adapter Agent
GMP Broker
Intelligent Broker
1.Find the suitable template
1.Find the suitable template
1.The suitable template found
1.The suitable template found
2. Generate the service description
3. Send the "Create a New Service" request
4. Verify the Service description
5. Acknowledge Registration

Figure 14: Service Publishing Action

4.2 Locating Broker Mechanism

In the one-tier marketplace, each service is published on the same online marketplace. If a user needs to find the service, he just needs to search only one catalog in the broker. In the Hierarchical Broker Framework, in order to use the tree-like framework, we separate one main broker from other sub-brokers. The Locating Broker Mechanism is used to help users to find the broker they need.

4.3 Matching Mechanism

The Matching Mechanism is used to help a buyer to match a suitable seller (service provider). Different industries or even the different services may need different Matching Mechanisms to achieve more precise matching. A suitable Matching Mechanism can let the broker find the service provider while satisfying the buyer's constraints more precisely.

In Figure 15, we show a complete flow that matches and invokes a service. First we will use the matching method to find a service and then invoke it. As for the invocation mechanism, we will introduce in it in the next sub-section. When a user needs a service, first he needs to find a suitable broker, and then he can ask it to create a matching session. The Intelligent Broker will invite the providers to join the matching session and help the user to find a suitable provider.

4.4 Invocation Mechanism

The Invocation Mechanism is the most important mechanism in integrating inter-enterprise processes. The automatic integration includes matching and invoking of the services automatically.

Sometimes, the Invocation Mechanism is used after the matching mechanism. The matching mechanism provides a suitable service provider and a suggested workflow. If the buyer accepts this workflow, the mechanism will submit it and invoke subsequent actions.

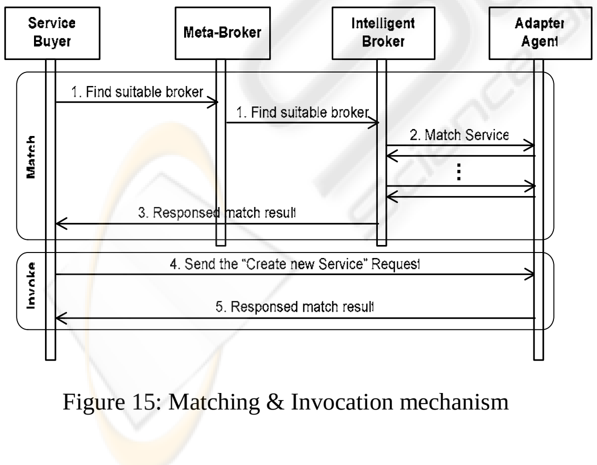

Figure 15: Matching & Invocation mechanism

5 IMPLEMENTATION

We use the Java programs and the web-service technology to accomplish the hierarchical broker framework. The former can execute in any platforms and the later is adopted by many organizations and enterprises. On the strength of these characteristics, clients and servers do not have special requirements in your legacy system. The hierarchical framework will be operated very easily and successfully. In summary, the major advantages of our Hierarchical Broker Framework to perform the automatic integration workflow are listed here.

The Intelligent Broker and the Adapter Agent are built as basic components. If we want to add a new broker/adapter, we just need to reuse these existing basic components to design a new broker/adapter to join the Hierarchical Broker Framework.

- Because of using the browser as the communication tool, users can manage and monitor processes in an easy and friendly way.
- The Web-service standard (SOAP, WSDL and UDDI) are open standards supported by W3C and OASIS, adopting these standards reduces the interoperability problems between different systems that follow the standards.

6 CONCLUSION

Nowadays, producing a complex product needs cooperation with many enterprise processes. How to find suitable service providers to establish a perfect workflow is a critical issue and is solved in this paper. In order to provide more efficient matching environment to deal with automatic integration of processes, we designed a Hierarchical Broker Framework. In this framework, we provide many useful functions that a traditional broker can not provide. It decomposes a main broker to many sub-brokers and provides a tree-like framework. The decomposition method uses a domain concept like DNS on the Internet. It also provides for each sub-broker a very flexible environment where the sub-broker can design suitable data format and matching method in its marketplace. The buyer and the seller in the Hierarchical Broker Framework will find each other more precisely. In the hierarchy, the parent broker has the responsibility to validate the child-brokers' specification.

Then, based on this Hierarchical Broker Framework, we established two main components, the Intelligent Broker and the Adapter Agent. The Intelligent Broker is a broker node in the framework.

It not only matches sellers and buyers, but also manages sub-brokers and member providers. The Adapter Agent is used to help some legacy operating systems to be integrated into this framework more easily. By these mechanisms, connecting business processes between enterprises are more convenient and efficient. Furthermore, enterprises have more opportunities to find suitable partners in net-marketplace.

REFERENCES

[1] IBM, WebSphere MQ Workflow (formerly MQSeries Workflow), http://www-3.ibm.com/software/integration/wmqwf/.

[2] G. Alonso, U. Fiedler, C. Hagen, A. Lazcano, H. Schuldt, N. Weiler: WISE: Business to Business E-Commerce. 9th International Workshop on Research Issues on Data Engineering (RIDE-VE'99). Sydney, Australia, March 23-24, 1999.

[3] P Grefen, Y Hoffner, CrossFlow - Cross-Organizational Workflow Support for Virtual Organizations, Proceedings of 9th International Workshop on Research Issues on Data Engineering, Information Technology for Virtual Enterprises, Sydney, Australia, March, 1999.

[4] Sumi Helal, Stanley Su, Jie Meng, Raja Krithivasan, and Arun Jagatheesan, The Internet Enterprise, IEEE Applications and the Internet, 2002. (SAINT 2002). Proceedings. 2002 Symposium on , Page(s): 54 -62

[5] A. Jagatheesan and A. Helal, Sangam: Universal Interop Protocols for E-service Brokering Communities using Private UDDI Nodes, Submitted to the IEEE Symposium on Computers and Communications - ISCC'2003, to be held in Kemer-Antalya, Turkey, Hune/July 2003

[6] Kenneth C. Laudon and Carol Guercio Traver, E-Commerce business. Technology. Society , Addison Wesley, 2002.

[7] IDEF Family of Methods, http://www.idef.com/default.html

[8] M. Wahl, T. Howes, and S. Kille. "Lightweight Directory Access Protocol (v3)," IETF RFC 2251, December 1997. http://rfc-editor.org/rfc/rfc2251.txt

[9] K. W. Edwards. Core Jini. Prentice Hall.

[10] Roy, J. and Ramanujan, A., Understanding Web services, IT Professional , Volume: 3 Issue: 6 , Nov/Dec 2001 Page(s): 69 -73

[11] Curbera, F., Duftler, M., Khalaf, R., Nagy, W., Mukhi, N.and Weerawarana, S., Unraveling the Web services web: an introduction to SOAP, WSDL, and UDDI, *Internet Computing, IEEE , Volume: 6 Issue: 2 , Mar/Apr 2002, Page(s): 86 -93*

[12] Simple Object Access Protocol (SOAP) 1.1, http://www.w3.org/TR/SOAP/

[13] Web Services Description Language (WSDL) 1.1, http://www.w3.org/TR/wsdl

[14] Universal Description, Discovery and Integration of Web Services, http://www.uddi.org/

[15] Weiming Shen;, Distributed manufacturing scheduling using intelligent agents, *Intelligent Systems*, IEEE [see also IEEE Expert] , *Volume: 17 Issue: 1 , Jan.-Feb. 2002 Page(s): 88 -94*