Component Based System Framework for Dynamic B2B Interaction

Jinmin Hu Paul Grefen
Department of Computer Science, University of Twente
P.O. Box 217, 7500 AE Enschede, the Netherlands
E-mail: {jimhu, grefen} @cs.utwente.nl

Abstract

Business-to-Business (B2B) collaboration is becoming a pivotal way to bring today's enterprises to success in the dynamically changing e-business environment. Though many business-to-business protocols are developed to support B2B interaction, none are generally accepted. A B2B system should support different B2B protocols dynamically to enable interaction between diverse enterprises. This paper proposes a framework for dynamic B2B interaction. A B2B transaction is divided into the interaction part and business implementation part to support flexible interaction. A component based system framework is proposed to support the B2B transaction execution. To support dynamic B2B services, dynamic component composition is required. Service and component notions are combined into a composable service component. The composition architecture is presented as well.

1. Introduction

The evolution of the Internet has brought a new way for the enterprises to interact with their partners. Many infrastructures and enterprise information systems are developed to extend businesses and value added services based on the Internet. Since the market is rapidly changing, the business collaboration is becoming more dynamic. For example, in a virtual enterprise environment, the cooperation parties can be dynamically selected and the business relationships among these parties are forged and dismantled very quickly. So, supporting dynamic B2B interaction has become an important functionality of enterprise information systems.

Business protocol standards play a key role in terms of B2B interaction. Only when the enterprise information systems implement some kind of business protocol standard, B2B interaction can be carried out among the cooperation parties. A B2B protocol standard in general is the description of the message formats exchanged, bindings to transport protocols, the sequencing, the process, the security to be provided and many more properties [5]. There already exist some B2B protocol standards under development or in use, for example EDI, RosettaNet, ebXML, OBI, UDDI and TPA. However, most of theses standards are still evolving and none are generally accepted. While different companies adopt different business logics, they may also adopt different B2B protocol standards as well. Hence, the information systems that aiming at flexible B2B interaction should support multiple B2B protocols at the same time to interact with other parties that adopt different B2B protocols. When collaboration parties are changed, the system must deploy new components that support new B2B protocol standard to replace the old ones.

Flexible B2B interaction requires the support of the internal business process implementation. Some existing systems, such as workflow management systems (WFMS) and ERP systems can support the internal business process ("private process") execution, but they do not support the B2B interaction that is viewed as "public process". In order to make inter-enterprise processes work, each involved enterprise has to implement not only its internal processes ("private processes"), but also its external behavior ("public processes") [4]. The B2B interaction support modules and the business process execution system must be integrated to support B2B business collaboration. This paper focuses on these two aspects: dynamic B2B interaction support and the integration framework.

A tightly coupled system is difficult to change and extend to meet the dynamic B2B interaction requirements. The current trend in the information system is moving away from tightly coupled systems and towards systems of loosely coupled, dynamically bound components [11]. From a traditional software engineering view, the software design cycle starts from business requirements to software component implementation and deployment. But so long a cycle cannot meet the dynamic requirements. CBSE (Component Based Software Engineering) approaches provide a possible solution. Components can be composed, reused, removed and replaced to support component based system configuration. The component-based approach potentially provides a good basis for support of dynamic B2B interactions.

Proceedings of the 26 th Annual International Computer Software and Applications Conference (COMPSAC'02)
0730-3157/02 $17.00 © 2002 **IEEE**

This paper proposes a component-based system framework to support dynamic B2B interaction. A B2B transaction is divided into two parts, the external part and internal part. The external part describes the B2B interaction and the internal part describes the steps for handling the incoming request. B2B interaction components are proposed for implementing the B2B protocol to support the execution of the external part of the transaction. Service components are used to implement business services to support the internal part of the transaction. When the enterprise information system has to interact with a new party's system that supports a different B2B protocol, some new B2B interaction components are developed, deployed and integrated into the existing system. The service components can be high level business components that implement a business process or a business activity or low level and generic software component that provide basic functions. A high level service component is a composite component created by combining low level components.

The remainder of this paper is organized as follows. Section 2 introduces a three-level B2B interaction framework. Section 3 describes B2B transaction with an example. A B2B transaction is divided to two parts: the interaction part and implementation part. Section 4 presents a component based system framework that implements the B2B transaction. Service component concepts are introduced to combine the service and component notions. The concept of service component composition is introduced. Section 5 presents the system architecture that enables service component composition. Some related work is introduced in Section 6. The last section gives a summary and presents future work.

2. Three-level B2B interaction framework

Business-to-Business (B2B) interaction is the exchange of information, service or goods and money between two business parties. One type of B2B interaction is the interaction between buyer and supplier. Another example of B2B interactions is the service exchange between service consumer and service provider. The B2B interaction framework shown in Figure 1 is vertically separated into three levels: business interaction model, business process level and information system level.

The Business Interaction Model describes the high level business collaboration. It provides a quick way to develop a strategic view of the business collaboration. A BIM describes the interaction participants and the roles they play, business functions they perform or services they deliver, the communication and exchanges of goods, services and information that cross the organizational boundaries. The business interaction model can be described in a UML sequence diagram. The collaboration objects are the two business parties; their roles can be viewed as the classes of the objects. The exchanged entities (information, goods, service) are described as sequenced messages. Figure 2 shows as an example of a business interaction model in a UML sequence diagram. The cooperation objects are the two parties: Enterprise A acts as a buyer and Enterprise B acts as a seller. They exchange information (Order, Order ACK, Order state request and Order state), money and goods when interacting with each other. Sequence diagrams that describe the BIMs are too abstract to be directly used in software implementation. However, through further elaboration and refinement, the final elaborated interaction model can be implemented by a couple of interaction activities. These activities belong to the external part of a B2B transaction that we will discuss in the next section.

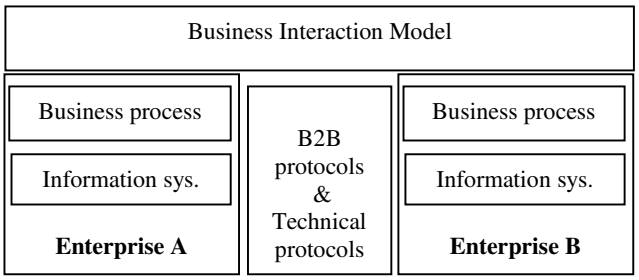

Figure 1. B2B interaction framework

The business process level describes the business transaction activities, including the business activities that describe the business logic of handling the requests from the opposite party and the interaction activities that implement the BIM. These two kinds of activities together describe a B2B transaction that we discuss in next section.

The information system level is the implementation level. Through information exchange and function invocation between the information systems, the B2B transaction is implemented. We show a component-based information system framework in section 4.

Shown in Figure 3, the protocols that enable business-to-business interaction plays the bridge role to connect the two involved business systems. These protocols include

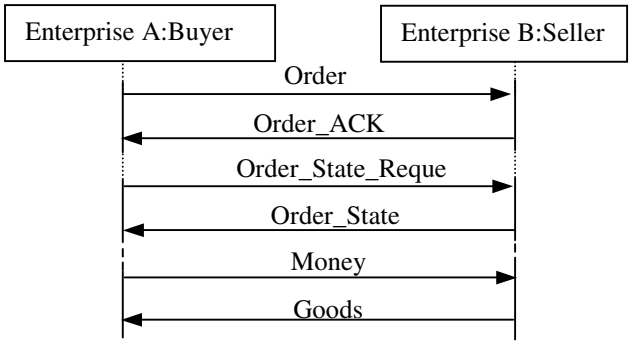

Figure 2. Business interaction model

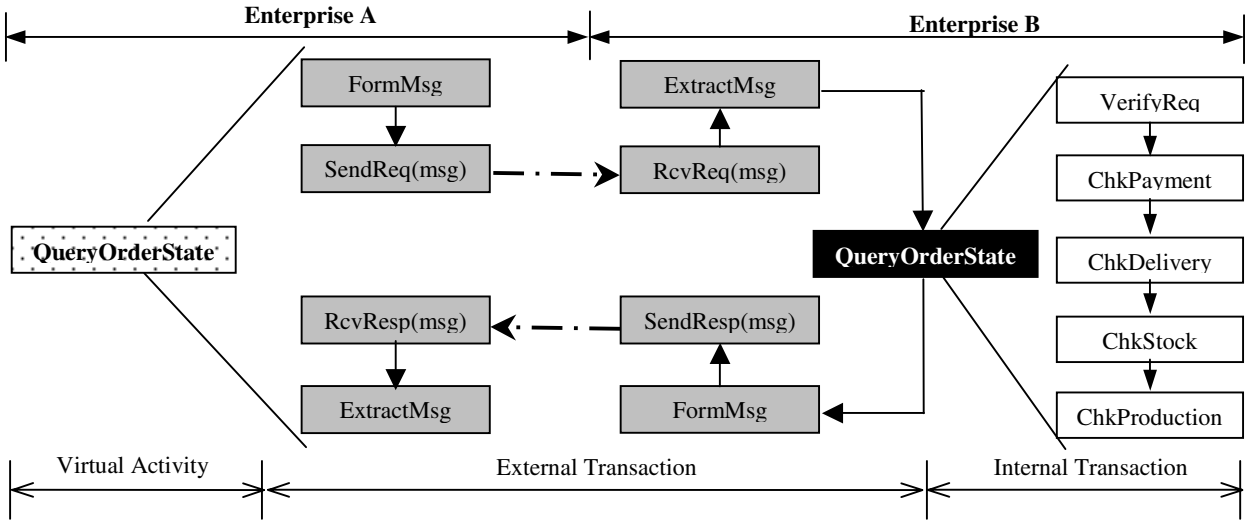

Figure 3. A B2B transaction example

business (process) level B2B protocols and technical level protocols. The technical protocol, like RMI, CORBA or SOAP/HTTP, enables the information system to exchange data and to support remote invocation.

3. B2B Transaction

A B2B transaction is a process that handles a business or service request placed by a business party (the buyer or service consumer). A B2B transaction consists of two parts, the interaction part and the implementation part. The interaction part is the external part that implements the elaborated business interaction model. This part consists of the interaction activities for receiving request and delivering the response. The external part of the transaction is visible to the cooperation party. Most B2B protocol standards define the external view of the B2B transaction. For example, in UMM N90 [14] and ebXML [8], 6 transaction patterns are given. The implementation part is the internal part that is hidden from the other party. It consists of the business activities that handle the incoming request and generate the response. The internal part of the B2B transaction usually is described as the business process or business workflow.

Figure 3 shows an example of a B2B transaction. This example describes a B2B transaction to implement order state query interaction shown as part of Figure 2. Enterprise A executes the activity "QueryOrderState" to query the detail state of an order that has been accepted by Enterprise B. Because the service "QueryOrderState" is provided by Enterprise B, the activity is not implemented in Enterprise A but in Enterprise B. We called the activity "QueryOrderState" within Enterprise A a virtual activity, which is mapped to a B2B interaction or a B2B transaction. According to the interaction model shown in figure 2, Enterprise A send a request message "Order_ State_Request" with some parameters such as order ID to Enterprise B. When Enterprise B receives the request by executing the activity "RcvReq", it extracts the request information and parameters from the incoming message. From the service request information, Enterprise B knows that the activity "QueryOrderState" is to be executed to provide this service. This activity is the real implementation of the order state query service. It is mapped to a set of concrete internal business activities (internal transaction): Activity "VerifyReq" is executed to verify the request. If the request is valid, the process including activities "ChkPayment", "CheckDeliver",

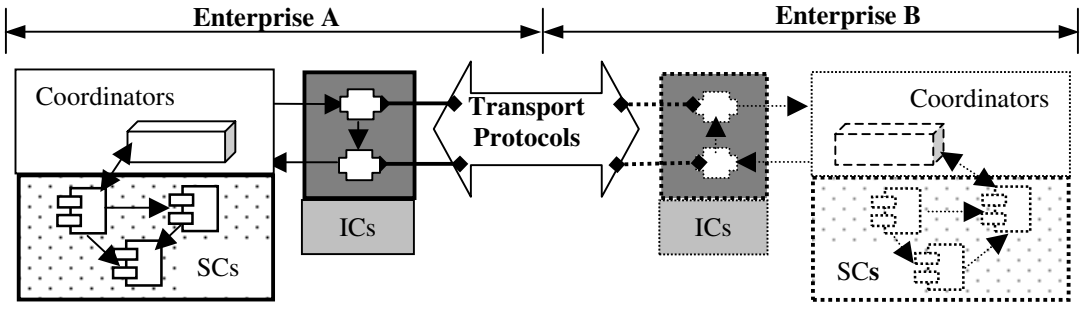

Figure 4 Component based system interaction framework

"ChkStock" and "ChkProduction", is executed to gather the detail information about the order handling. After the internal transaction is completed, the in-progress details of the order are put into a formatted message as "Order_State" and sent back to Enterprise A. The message is formatted and created according to the agreed B2B protocol. The message is sent and received based on the agreed transport protocol. If Enterprise A successfully receives the response massage, the order state information will be extracted from received formatted message and will be used as input data by other activities within Enterprise A. So far, the whole B2B transaction is completed.

In the next section, we propose our component-based framework to implement the B2B transaction.

4. Component Based B2B System Framework

To support dynamic B2B interactions and flexible business logics to respond to different business requirement, the system must be rapidly constructed or configured to realize fast change response. Component Based System (CBS) approach provides a possible way to construct such a system.

Since a B2B transaction consists of interaction activities and business activities, a component based system that supports B2B transaction execution can be constructed from two types of components: interaction components that executes interaction activities and service components that support business activities execution. Figure 4 shows an overview of our CBS interaction framework.

The interaction components (ICs) implement the B2B protocols. They interact with the collaboration party's interaction components based on certain specific transport protocols to exchange messages or documents. To support different B2B protocols, different ICs that implemets different B2B protocol are used, for example, EDI, RosettaNet, ebXML. The interaction components may be developed based on different transport protocols to support multiple access channels, for example, an ebXML business documents can be sent by using MIME/SMTP or SOAP/HTTP. The interaction components only execute the interaction part of a B2B transaction. The other part is executed by the service components (SCs) The service components fulfill business workflow or business activity execution. The coordinator can dynamically connects the service components with the interaction components to support dynamically B2B interaction based on the agreed B2B protocol.

From a service oriented view, component and service are the two sides of the same coin. To express this double meaning and to distinguish from the generic software component definition, we introduce the concept of service component. Before describing service components, we give a description of service. A service is some kind of function. However, not all functions can be considered as services. Two kinds of entities are concerned with a service: the function provider and the function user or consumer. A service must be accessible.

Hence, A service component is a component that implements a service that can be accessed or invoked by other components or applications. The service components can be high level business components that implement a business process or a business activity. It can also be a basic service component that provides generic functions, such as a database access component. A service component can be composite: a high level component can be composed from low level ones. Figure 5 shows the levels of service components used to implement our B2B interaction framework. Based on the business function and their granularity, we classify service components into four levels. From low level to high level, there are Basic Service Component (BSC), Business Activity Oriented Service Components (BAOSC), Business Process Oriented Service Components (BPOSC) and Business Interaction Oriented Service Components (BIOSC).

A Basic Service Component (BSC) is a domain independent component that can provide generic and well-defined functions. Usually, it is implemented based on some standard component implementation specifications. For example, Microsoft data access component ADO is a BSC, which is a COM component that provides services

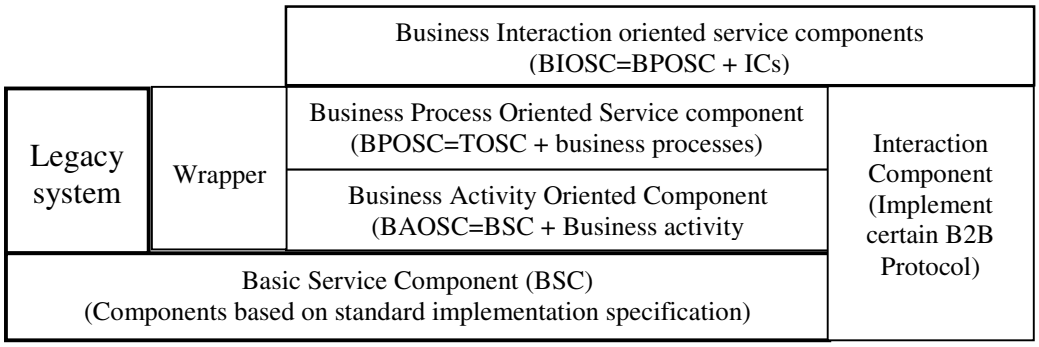

Figure 5 Service component levels

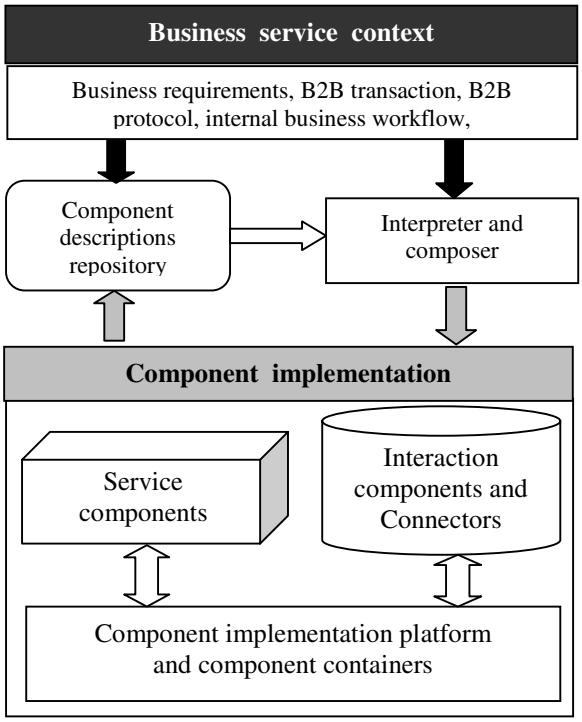

Figure 6 Composition architecture

to access databases.

A Business Activity Oriented Service Component (BAOSC) is a software component that can autonomously carry out a business activity in a certain business domain. A BAOSC can be developed or composed from some basic service components by adding some business activity information. For example, an order-handling component can invoke a service provided by a database access component to store order information in a database.

A Business Process Oriented Service Component (BPOSC) is a composite component that can be composed from BAOSCs based on a business process. The business process can be presented as workflow that shows the execution sequence of the business activities. A BPOSC can enact the internal business process of a B2B transaction. Usually, a BPOSC includes a internal coordinator to route the workflow execution and coordinate the BAOSCs' invocation.

Both BAOSC and BPOSC are B2B protocol independent but domain specific components. Their functions are handling the internal part of a B2B transaction. Both BAOSCs and BPOSCs can be created through wrapping legacy systems. For example, a traditional process based system can be wrapped into one or more BPOSCs. Legacy systems that can carry out some business activities but not the whole business process can be wrapped into some BAOSCs.

The highest level service component to implement our B2B interaction framework is the Business Interaction Service Component (BIOSC). It is composed from a BPOSC and the B2B interaction components that implement a certain B2B protocol (for example, ebXML). A BIOSC provides an accessible service to the business parties. It can be published as a service on the Internet and provides access channels to the consumers. As discussed above, service is accessed and provided by initiating and executing a B2B transaction that includes interaction and business process execution, so, a BIOSC implements a B2B transaction. Apparently, a BIOSC is B2B protocol dependent and domain dependent component.

As shown in the framework (Figure 5), a coordinator operates as a high level connector to connect the high level service component BPOSC and ICs. It coordinates service component invocation according to the incoming business service request. Through coordinators, dynamism of the B2B interaction is attained. Different interaction components can be dynamically connected with the service components through the coordinators, hence, the same service can be delivered through different B2B protocols channels.

5. Composition architecture

To support dynamic B2B interaction, we require service component composition. Through service component composition, new required services can be rapidly implemented and delivered. Our target is to create high level composite service components to meet new B2B interaction needs.

Figure 6 shows the overall architecture for service component composition. The existing service components or the ones wrapped from legacy systems are described in a unified form. The descriptions of the service components are stored in a repository to be queried by the composer. The interpreter interprets the business context to find the business requirements. According to these requirements, the composer queries the service component descriptions in the repository to obtain the proper service components that meet the business requirements. Then, connectors or coordinators are to be used to glue these selected service components to create high level composite components. The composite service components can be re-composed to a new service component when the business logic or business protocol is changed. For example, a new Interaction Component that supports new business protocol can replace the old one within a composite BIOSC to meet the dynamic interaction requirements.

Component description is very important for service component composition. Traditionally, an Interface Definition Language (IDL) is used as signature of a software component. For a service component, IDL is not adequate to represent its complicate properties because the business property, the implementation property,

deployment property and other properties should be represent in such a description. XML is used as a basis to describe the properties of a service component. The component descriptions themselves are XML documents that follow a prescribed but extensible XML schema definition language. In the Internet and e-service domain, WSDL (Web Service Description Language) [6] is used to describe web services. We extend WSDL by adding related business information (business workflow, business activities and business roles), implementation specification (for example, JAVA class or CORBA object), deployment information (for example, java class packages), composition information (for composite component) to describe the service components. Details about the service component description can be found in our internal report[1].

6. Related work

Some B2B protocols are developed to support B2B e-commerce, for example ebXML [7] and RosettaNet [12]. To support multiple B2B protocols, [5] proposes a B2B protocol engine to master the communication between trading partners. Currently, e-service is the new paradigm for e-business [11]. In this direction Web service [10] attracts a lot of attentions. WSDL is created to describe web service and UDDI [15] to support web service registry. But current web services focus more on application system interaction level and less on business level. WSFL [9] and WSCL [3] are concerned with B2B interaction process descriptions but have not been finished. High level B2B integration based on electronic contract and service frameworks is becoming an interesting research topic. [1] describes a conceptual framework for electronic contracting. The framework described in this paper can be viewed as the support system for e-contract enactment.

7. Summary and future work

Aiming at B2B collaboration, this paper proposes a B2B interaction framework and describes a B2B transaction. To support dynamic B2B interaction, a component based system framework is proposed to support the B2B transaction execution. The interaction components that implement the B2B protocol are used to support interaction activities execution. Service components can be composed to support business workflow and business activities execution. An overall architecture of service component composition is proposed.

The E-service computing paradigm and Web service technologies have created new opportunities to find a solution to supporting dynamic e-business. B2B integration in the service level based on service agreement or service contract has become the trend. Our future work is focusing on this perspective. The existing framework will be improved in the service-oriented direction. A system for contract based B2B integration will be researched. A prototype system will be built based on IBM MQSeriese Workflow and current web service technologies and our electronic contracting framework.

[1] http://www.ctit.utwente.nl/publications/Tr02/

6. References

[1] S. Angelov, P. Grefen. "A Conceptual Framework for B2B Electronic Contracting", Collaborative Business Ecosystems and Virtual Enterprise, Proceeding of IFIP PRO-VE'02, Kluwer Academic, Sesimbra, Portugal, 2002.

[2] W. Brown and K. C. Wallnau, "An Examination of the Current State of CBSE: A Report on the ICSE Workshop on Component-Based Software Engineering", 1998. http://www.sei.cmu.edu/cbs/icse98/summary.html.

[3] D. Beringer, H. Kuno and Mike Lemon. "Using WSCL in a UDDI Registry 1.02, UDDI Working Draft Technical Note Document", 2001. http://www.uddi.org/pubs/wscl_TN_forUDDI_5_16_011.doc .

[4] C. Bussler, "The Role of B2B Protocols in Inter-Enterprise Process Execution", Proceedings of Technologies for E-Services, Second International Workshop, TES 2001, Rome, Italy, 2001, LNCS 2193, pp.16-29.

[5] C. Bussler, "B2B Protocol Standards and their Role in Semantic B2B Integration Engines", IEEE Bulletin of the Technical Committee on Data Engineering, Special Issue on Infrastructure for Advanced E-Services, 2001, vol. 24 no. 1, pp.3-11.

[6] E. Christensen, F. Curbera, G. Meredith and S. Weerawarana, Web Services Description Language (WSDL) 1.1, http://www.w3.org/TR/wsdl .

[7] ebXML. http://www.ebxml.org .

[8] ebXML Business Process Team. "ebXML E-Commerce Patterns v1.0", May 2001, http://www.ebxml.org/specs/bpPATT.doc.

[9] IBM Software Group. "Web Services Flow Language, Version 1.0.", 2001. http://www-4.ibm.com/software/solutions/webservices/pdf/WSFL.pdf .

[10] IBM Software Group, "Web Services Conceptual Architecture (WSCA 1.0)", 2001. http://www-4.ibm.com/software/solutions/webservices/pdf/WSCA.pdf .

[11] T. Pilioura and A. Tsalgatidou, "E-Services: Current Technology and Open Issues", Proceedings of Technologies for E-Services, Second International Workshop, TES 2001, Rome, Italy, 2001, LNCS 2193, pp1-15.

[12] RosettaNet. http://www.rosettanet.org.

[13] SOAP, Simple Object Access Protocol (SOAP) 1.1 2000, http://www.w3.org/TR/2000/NOTE-SOAP-20000508 .

[14] Techniques & Methodologies Working Group (TMWG). "UN/CEFACT's Modelling Methodology (N090)",DRAFT, CEFACT/TMWG/N090R10, Nov 2001. http://www.gefeg.com/tmwg/n090r10.htm.

[15] UDDI. http://www.uddi.org .