Implementation of an interoperable interface to exchange B2B messages between heterogeneous computer platforms

Giacomo Paschina[1], Andrea Clematis[1], Gabriele Zereik[1] and Daniele D'Agostino[1]
[1]Institute of Applied Mathematics and Information Technologies
National Research Council of Italy
Genoa, Italy
(giacomo.paschina||andrea.clematis||gabriele.zereik||daniele.dagostino)@ge.imati.cnr.it

***Abstract*-In a Business-to-Business (B2B) scenario strategic goals are *integration* and *interoperability*. *Integration* regards the process, within an enterprise, of linking different software systems to become part of larger systems. *Interoperability* is strictly linked to the standardization of commercial messages and communication protocols. In this work we present the implementation of an environment that provides a common and coherent interface between different trading partners and that can be easily integrated with the computer systems and any B2B middleware used by the enterprises involved in the e-commerce communication. We show how to apply it to connect two B2B middlewares: Oracle B2B, a commercial platform, and Hermes 2.0, an open source project developed by the University of Hong Kong. The commercial transactions are based on the electronic business eXstensible Markup Language (ebXML) standard and the releated ebXML Messaging Services (ebMS) communication protocol. The interactions take place through web applications that provide a common view of the system to the trading partners involved in the communication.**

I. INTRODUCTION

The Internet has radically changed the way of communication between companies in electronic commerce (e-commerce) scenarios. Almost all the companies relied on the Internet in order to use a common platform for processes automation and global visibility [1, 2, 3]. This caused a platforms fragmentation in the field of business messages, making the role of *interaction* strategic for the e-commerce. *Interaction* consists of *interoperability* and *integration*: organizations spent a lot of time and economic resources in solving integration and consistency problems between several software systems [4, 5].

While *integration* typically attempts to build a monolithic view of the enterprise [6], *interoperability*, that can be viewed as inter-enterprise *integration*, is an essential requirement for today's successful business [7] and it is focused on the exchange of meaningful, context-driven data between autonomous systems [6]. Its goal is to simplify the management of business transactions between trading partners creating an common platform, based on standards, in order to reduce inter-enterprise integration costs and improve business trading. *Integration* and *interoperability* strongly depend on standards such as the eXstensible Markup Language (XML) [8], the United Nations rules for Electronic Data Interchange for Administration, Commerce and Transport (UN/EDIFACT) [9], the electronic business eXstensible Markup Language (ebXML) [10] and other standards for data format and inter-enterprises exchanges.

In this paper we present the design and the implementation of an interoperable environment that provides a common and coherent interface between trading partners, hides the underlying technology, and can be easily integrated with computer systems and in Business-to-Business (B2B) middlewares.

DOI: 10.978.819252/12229

II. STANDARD USED IN B2B COMMUNICATION

The goal of the standards used in B2B communication is to electronically connect trading partners through the exchange of structured messages containing business data [11]. All interactions take place over networks like the Internet or Value Added Network (VAN) in a completely automatic way [12]. Internet with its high connectivity and low costs provides a standard and economic platform to companies and their partners [13]. There are many B2B protocols available that define various aspects of inter-companies communication to provide an explicit specification of requirements in order to implement the automation of the collaboration between companies [19]. Typically their basic concepts are: messages format, activities for sending and receiving of messages, business messages and acknowledgment messages, time-out and retry logics, duplicate check and avoidance and roles [14].

In this section we describe one of the most important standard used to implement UN/EDIFACT transaction: the ebXML standard and the related ebXML Messaging Services (ebMS) [15] protocol, that are used in this work to implement commercial exchanges between the heterogeneous platforms considered.

A. ebXML

ebXML is a set of standards based on XML and proposed by the Organization for the Advancement of Structured Information Standards (OASIS) [16] and the United Nations Centre for Trade Facilitation and Electronic Business (UN/CEFACT) [17]. In this way, organizations have a standard method to communicate in common terms, define their business processes, and establish trading relationships [18].

ebXML is based on the concept of *trading partnership*, that allows companies to find new trading partners and to agree on the technical aspects to implement the electronic communication. A *trading partnership* is realized through two XML based documents: the *Collaboration Protocol Profile* (CPP) and the *Collaboration Protocol Agreement* (CPA). With CPPs, companies can indicate the line of business, process, exchange techniques, supported technologies, and security methods implemented. Merging their CPPs, companies create the CPA document, where they define business processes, messages structure, and technologies used to implement the commercial transaction. A typical ebXML creation of *trading partnership* scenario consists of three phases: an implementation phase, in which the trading partner analyzes his business processes and publishes them into a central public server called *registry*, a discovery and recovery phase, in which trading partners access to the *registry* to discover business processes and interfaces published by other trading partners, and an execution phase, in which trading partners merge CPP into CPA documents and start the exchange of ebXML messages through the ebMS protocol. An ebXML message consists of two parts: a *header* that includes all the information about routing and delivery, and a *payload* that contains the commercial data to be exchanged between trading partners. *Payload* can contain any type of data: XML document, binary data and so on.

B. ebMS

ebMS is a standard protocol defined by OASIS, that implements the message enveloping and header document schema used to transfer ebXML messages over communication protocols such as the HyperText Transfer Protocol (HTTP) [20] or the Simple Mail Transfer Protocol (SMTP) [21], and manages the sending and receiving ebXML messages [22]. ebMS is defined as a set of layered extensions to the base Simple Object Access Protocol (SOAP) [23] and it provides the message packaging, routing, and transport facilities for the ebXML infrastructure, using the

DOI: 10.978.819252/12229

information contained in the trading partners' CPA documents. To recover this information and thus allow the exchange of ebXML messages, a Message Sevice Handler (MSH) is needed: a software platform that implements the ebXML architecture and is able to send and receive messages in accordance with the specifications described in the ebMS protocol.

III. DESIGN AND IMPLEMENTATION OF THE INTEROPERABLE INTERFACE

In a B2B messages exchange scenario, two enterprises want to send business messages over the Internet. The communication must be carried out in a totally independent way from their computer systems and B2B middlewares. A B2B messages exchange relies on the definition of the standard used for the message format and the related communication protocol, on the B2B middlewares used by the enterprises, and on the *integration* issues with trading partners' computing platforms. We developed a web application that implements a common Graphic User Interface (GUI) to provide an interoperable and coherent environment between trading partners, and manages the issues related to the *integration* with a company's computer systems and B2B middleware. The web application essentially consists of two parts: the common GUI, and a specific module that is different for any B2B middleware and organization and consists of two *adapters*, that are software components involved, on one hand, to manage the *integration* with the company's computer systems, and, on the other hand, to interact with the B2B middleware in a completely transparent way for the end user. The web application handles both the sending and receiving of commercial messages. It is worth noting that the web application has the important characteristic of being totally independent from the computing platform on which it runs. In fact, given its web-oriented nature based on generic servlet containers, it can be used on any computer system and accessed from any computer on the same network. In addition the web application is completely unrelated from the standard used in the B2B communication because it depends only on the methods exposed by the B2B middleware to allow access to its services. The standards and the communication protocols are managed directly by the B2B platform. The fact that the web application is independent by both the computer systems and the standards used for B2B exchanges, makes it fully portable to any platform used to exchange business messages.

A. The common GUI component of the web application

The common part, which provides a simple and intuitive graphical interface for the insertion of the information to be sent or the recovery and integration of the messages received, has also the purpose of hiding the interaction with the company's computer systems and the B2B middleware in order to provide a monolithic view of the entire environment for the exchange of commercial messages.

B. The adapters of the web application

As mentioned previously, the *adapters* are software components involved in the *integration* stage of the web application. They are specific for any B2B middleware and computer system and must be modified during the setup phase of the environment in order to allow the web application to interact with the company' computer system and B2B middleware, solving the *integration* issues. The setup stage involves two phases: the analysis of the methods exposed by enterprise's B2B platform to interact with it and of the *integration* issues with the company's computer system, and the adaptation phase that consists of the modification of the *adapters* in order to make them able to

DOI: 10.978.819252/12229

exchange information with the B2B middleware and to integrate them with the enterprise's system. In a typical scenario an end user (via GUI) or the computer system ask the web application to send or retrieve a message. Fig. 1 shows the generic architecture of the interoperable environment and its usage via GUI: in this scenario the trading partner A accesses the web application via browser and inserts the data to be transferred, including any attachment, through the interface provided by the application (stage 1). Then the *adapter*, related to the B2B middleware, recovers the information and delivers the data to the platform in the format required by it (stage 2 and 3). At the end of the interaction and at the complete retrieval of all the data required, the B2B platform starts the communication (stage 4). For the reception the trading parner B accesses the web application to retrieve a message by its unique id (stage 5). Following the request the web application connects to the B2B middleware through the *adapter* (stage 6 and 7) and retrieves the message with any attachment in a location specified by the end user (stage 8).

IV. An Interoperable Environment Prototype

In this section we present a brief description of the methods of interaction and integration used to modify the *adapters*, of the computer platforms adopted to implement the interoperable environment, and of the development of web applications to provide an interface for the exchange of ebXML messages. The creation and the configuration of CPA documents related to the *trading partnerships* for both platforms are omitted.

A. Oracle B2B

Oracle B2B is an e-commerce gateway enabling the secure and reliable exchange of business documents between trading partners [24]. Together with the Oracle SOA Suite it provides an architecture enabling a unified business process platform, end-to-end instance tracking, visibility, auditing, process intelligence, governance, and security.

Oracle B2B consists of two servers: the administrative one and the managed one. The former deals with the system general management including the creation of new trading partners, the partnership documents and the general state of the platform services. The latter handles directly the dispatch and the retrieval of commercial messages. The managed server can not be accessed directly, but only through a Service Component Application (SCA) implemented specifically for each partnership and to which our web application interfaces in order to send B2B messages.

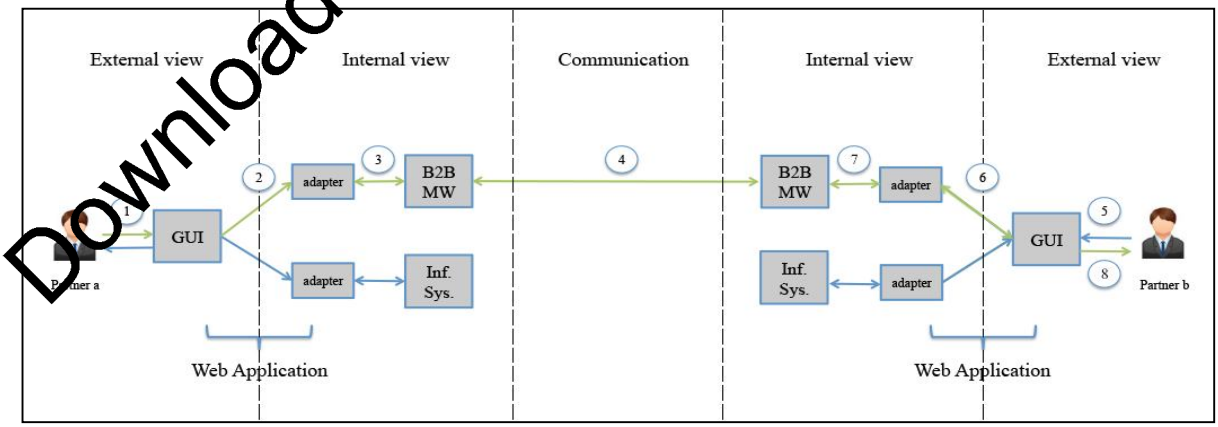

Figure 1. The generic architecture of the interoperabile environment and an example of B2B message exchange.

DOI: 10.978.819252/12229

B. Hermes 2.0

Hermes 2.0 is an open source gateway, developed by the Center for E-Commerce Infrastructure Development (CECID) of the University of Hong Kong, that provides a secure and reliable information transfer using common communication standards such as ebMS 2.0 [25]. This gateway implements a web service that handles either incoming and outgoing requests and can be directly called using SOAP Application Programming Interface (API) within the web application.

C. Development of the web application adapter to send messages from Hermes 2.0 to Oracle B2B

To implement its web service, Hermes 2.0 uses Apache Tomcat [26]: an open source servlet container developed by the Apache Software Foundation (ASF) [27] and is able to host web applications. To setup a SOAP messages exchange it is necessary to recover the information about the *trading partnerships* stored within Hermes SQL database. In particular, it is necessary to get the following values: *CPA ID*, *Service*, *Action*, and *Transport endpoint*.

The web application retrieves these values and provides them to the Hermes web service through an instance of a class, which implements the SOAP communication and extended SOAPRunner class. The SOAP message contains the *payload*, which represents the real messages to send, the *attachments*, and the *body* that includes the information, recovered from database, needed by the Hermes web service in order to process and send ebXML messages. It is worth noting that the message can contain every type of *attachment* with only limitation for image size that must not be larger than 5MB. The *attachments* are sent as binary string of data.

D. Development of the web application adapter to send messages from Oracle B2B to Hermes 2.0

To send an ebXML message, Oracle needs the creation of a web service associated with the *trading partnership*. As described in section IV.A, to create the web service a SCA application must be implemented. It consists of a set of pre-configured software modules that provide an external interface to connect to Oracle B2B. The principal software component of a SCA application is the *Mediator* that manages the recovery of the message and the interface with Oracle B2B in the middleware infrastructure. The SCA application also defines the Web Services Description Language (WSDL) [28] schema that is essential to create an own web application that exploits the created web service. Unlike Hermes 2.0 that uses the SOAP API to communicate with its web service, it is worth noting that in Oracle B2B only the methods exposed by the SCA application in the WSDL schema must be used to send messages through an external component.

E. Receiving a ebXML message

In this section we discuss about the receiving methods implemented for Oracle B2B and Hermes 2.0. Oracle B2B implements automatically the retrieval of the message for the textual part and for many Multipurpose Internet Mail Extensions (MIME) [29] *attachments* through an appropriate configuration of the environment. As regards Hermes 2.0, it is necessary to implement other modules of the web application in order to extrapolate data from the Hermes web service and to provide a GUI to the end user. The web application recovers the ebXML file indicated by the user and parses it to catch the text content and to implement the conversion of any *attachment* to the original format.

DOI: 10.978.819252/12229

V. CONCLUSIONS AND FUTURE WORK

In this paper we show the design and the implementation of a common interface between the trading partners, that can be easily integrated with any computer system and B2B middleware architecture used by the enterprises involved in the e-commerce communication. The strenght of this environment is that the web applications are completely independent by both the enterprise's computer systems and the standards used for B2B exchanges, making it fully portable to any platform used to exchange business messages.

As future developments it could be taken into account the possibility of using the cloud computing technologies to made the interoperable interface completely free from the enterprise's on premises resources and available from all over the world.

REFERENCES

[1] B. Medjahed, B. Benatallah, A. Bouguettaya, A. H. H. Ngu, A. K. Elmagarmid, "Business-to-business interactions: issues and enabling technologies", The VLDB Journal — The International Journal on Very Large Data Bases, Vol.12 NO.1, p59-85, May 2003.
[2] A. Dogac, "Special Issue on Electronic Commerce", ACM SIGMOD Records, Vol. 27, No. 4, December 1998.
[3] A. Dogac, "Special Issue on Electronic Commerce", Distributed Parallel Databases, Vol. 7 No. 2, 1999.
[4] Fastwater, *www.fastwater.com*, 2014.
[5] R. Nagappan, R. Skoczylas, R. P. Sriganesh, "Developing Java Web Services: Architecting and Developing Secure Web Service Using Java", Wiley Publishing, 2002.
[6] J.T. Pollock, "The Big Issue: Interoperability vs. Integration", eAI Journal, October 2001.
[7] M. Zaremba, M. Herold, R. Zaharia, T. Vitvar, "Data and Process Mediation Support for B2B Integration", Evaluation of Ontology-based Tools and the Semantic Web Service Challenge , in conjunction with ESWC 2008, 2008.
[8] W3C, eXtesibile Markup Language (XML), *htttp://www.w3.org/XML*, 2014.
[9] UN/EDIFACT, *http://www.unece.org/trade/untdid/welcome.html*, 2014.
[10] ebXML, *htttp://www.ebxml.org*, 2014.
[11] C. Bussler, "Modeling and Executing Semantic B2B Integration", Proceedings of the 12th It'l Workshop on Reasearch Issue in Data Engeneering e-Commerce/e-Business System (RIDE'02), IEEE, 2002.
[12] VAN, *htttp://www.edi-info-center.com/html/vans.html/*, 2014.
[13] W. K. Chong, B. L. Tan, E. M. Tadjouddine, M. Shafaghi, "On developing Interoperable B2B e-Commerce Model for SMEs", 2010 International Conference on E-business, Management and Economics, Hong Kong, 2011.
[14] C. Bussler, "The Role of B2B Protocols in Inter-Enterprise Process Execution", F. Casatim D. Georgakopoulos, M. C. Shan (ED.): Tes 2001, LNCS 2193, p. 1629, 2001.
[15] ebMS, *http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/core/cs02/ebms_core-3.0-spec-cs-02.html*, 2014.
[16] OASIS, *htttp://www.oasis-open.org/*, 2014.
[17] UN/CEFACT, *http://www.unece.org/cefact.html*, 2014.
[18] OASIS, "ebXML enabling a global electronic market", *http://www.ebxml.org/specs/ebMS.pdf*, 2014.
[19] M. Bernauer, G. Kappel, Gerard Kramler, "B2B Protocol Specification by Example Using WSDL and ebXML", 2003.
[20] HTTP, *http://tools.ietf.org/html/rfc2616*, 2014.
[21] SMTP, *http://tools.ietf.org/html/rfc5321*, 2014.
[22] UN/CEFACT org., OASIS org. (2002). Message Service Specification ebXML Transport, Routing and Packing version 1.0. In ebXML official web site. Retrieved October 2013, from http://www.ebxml.org/geninfo.htm
[23] W3C, Simple Object Access Protocol (SOAP), *htttp://www.w3.org/TR/soap*, 2014.
[24] Oracle, "Oracle Fusion Middleware User's Guide for Oracle B2B, 11g Release 1 (11.1.1.5.0)", *http://docs.oracle.com/cd/E21764_01/integration.1111/e10229.pdf*, 2014.
[25] CECID, *http://www.cecid.hku.hk/hermes.php*, 2014.
[26] Apache Tomcat, *http://tomcat.apache.org/*,2014.
[27] ASF, *http://www.apache.org/*, 2014.
[28] W3C, Web Service Description Language (WSDL), *htttp://www.w3.org/TR/wsdl*, 2014.
[29] MIME, *http://tools.ietf.org/html/rfc1521*, 2014.

DOI: 10.978.819252/12229