KATHOLIEKE
UNIVERSITEIT
LEUVEN

DEPARTEMENT TOEGEPASTE ECONOMISCHE WETENSCHAPPEN

RESEARCH REPORT 0304

IDENTIFYING WEB SERVICE INTEGRATION CHALLENGES

by
F. GOETHALS
W. LEMAHIEU
J. VANDENBULCKE

D/2003/2376/04

Identifying Web Service Integration Challenges

Frank Goethals, Wilfried Lemahieu, Jacques Vandenbulcke

Katholieke Universiteit Leuven – Faculty of Economics and Applied Economics
Management Information Systems Group
SAP-leerstoel Extended Enterprise Infrastructures
Naamsestraat 69, B-3000 Leuven, Belgium
{Frank.Goethals, Wilfried.Lemahieu, Jacques.Vandenbulcke}@econ.kuleuven.ac.be
Phone +32 16 326880, Fax +32 16 326732

Abstract. Web services technology promises well for the future of Business-to-Business integration (B2Bi). However, this technology is still in its infancy and the community is facing many challenges. In this paper we discuss some important B2Bi issues and look how web services could play their part in these. Currently, many web services related standards are being drawn up, but most of these are still immature and do not bring a real answer to the proposed challenges. Consequently, many topics for future research can be identified.

Accepted for the Information Resources Management Association (IRMA)
International Conference on Information Technology and Organizations: Trends, Issues, Challenges and Solutions
May 18 - 21, 2003, Philadelphia Pennsylvania, USA

INTRODUCTION

Throughout the last decade, companies became aware of the many benefits collaboration with other companies could render. Information that used to be dispersed throughout the Supply Chain is now being shared. By sharing information and processes, companies form extended enterprises. This way, companies render their processes more efficient and more effective, which improves the performance of the whole Supply Chain.

There has been quite some evolution in the way companies have been trying to automate their communication. During many years, big organisations have been establishing EDI (Electronic Data Interchange)-connections between systems. EDI has some advantages (such as cost savings) over the classical way of conducting business, but it is still very limited in its applicability. The biggest drawbacks of EDI are that it is complex, expensive and difficult to implement. In contrast, web services technology appears as a flexible and cheap technology for integration.

For a long time, ICT (Information and Communication Technology) has been restricting employees, forcing them to work (informally) around problems in the system. However, ICT should be *enforcing* business people. Modelling and implementing ICT in terms of 'business events' and 'services' could prove useful to realise this. In our opinion, web services technology could manifest itself as a concept that gives business-people more control over processes. The idea of flexibly modelling (and implementing) processes as an ensemble of (distributed) services is a futuristic view, which the community is trying to realise and which we will research ourselves in future projects. However, first the basics of the web services infrastructure should be well considered and stable. In what follows, we discuss web services technology in more detail and present some basic B2Bi (Business-to-Business integration) challenges.

WEB SERVICES

The idea behind web services is that IT-systems can offer services to one another. Currently these are mostly query services (e.g. stock checking), but action based services (such as purchasing) can also be achieved through web services. The use of web services is mostly situated in the B2Bi realm. However, web services cannot only be used for B2Bi, but also for Enterprise Application Integration (EAI). With the latter we mean the integration of the systems within the company walls (this in contrast to B2Bi). Many companies seem to deploy web services in the first place for EAI purposes [1]. Nowadays, only about 12 percent of the web services projects are aimed directly at B2Bi [2]. This practice is justified by many authors who state that companies should first start using web services for their internal integration projects, before they venture using web services in B2Bi projects (see for example [3]). Nevertheless, some authors (e.g. [4]) advocate the *outside in* approach for deploying web services, i.e., they suggest that web services should primarily be used for offering services to the outer world, before using web services within the company walls. The main argument for such statement is the provision that companies which do not immediately offer web services for external use will '*be left breathing the exhaust of more forward-thinking competitors*' [4].

WHAT ARE WEB SERVICES?

The meaning of the term 'web service' is not unambiguously defined. The Internet is strewn with definitions of the term. The W3C (World Wide Web Consortium) defines the term as follows [6]:

"*A Web service is a software application identified by a URI, whose interfaces and binding are capable of being defined, described and discovered by XML artifacts and supports direct interactions with other software applications using XML based messages via internet-based protocols*"

A more restrictive description is given by Leymann [7]:

'*self-contained, modular business process applications that are based on the industry standard technologies of WSDL (to describe), UDDI (to advertise and syndicate), and SOAP (to communicate)*'.

SOAP (Simple Object Access Protocol), WSDL (Web Service Description Language) and UDDI (Universal Description, Discovery and Integration) are three standards web services technology gets constantly associated with (see below). In what follows, we discuss the technology requirements for realising flexible B2Bi. Nowadays, this integration always involves one or more humans in the loop, be it for searching web services, for binding web services or for any other related activity. Note that the integration technology would be more powerful if it would enable software agents[i] to search and discover the needed services and to compose services automatically (see below).

B2Bi Challenges

Web services technology promises well for the future of B2Bi. In the remainder of this paper, we discuss *what* is needed to realise a flexible integration of systems through web services technology (without claiming this discussion is exhaustive), rather than examining *how* to bring this about. The broad outlines of the discussion can also be found with Glass [12], and with the WSMF (Web Service Modeling Framework, [13]). We have three remarks about the discussion. First, during the first years to come, web services are expected to be used (in a B2B context) primarily for *partner*-integration [2, 11]. However, in the long term, web services may be used for integrating systems of unknown parties as well. To realise the latter, the infrastructure used for partner-integration should be built in such a way that it can easily be adapted to new requirements, i.e., there should be *functional scalability* in the infrastructure. Therefore, in our discussion, we start from the long-term vision. Secondly, it will become clear that when it comes to B2Bi, web services *alone* will not do the trick. There is also an (often underestimated) need for semantics [15]. In our discussion, we point at this necessity. Thirdly, *Standards*, drawn up to ease the integration of systems and to augment interoperability, form an important aspect of our discussion. This way, the discussion forms a framework for situating different industry standards.

1. To allow for the highest possible flexibility, we assume that services will not only be used by partners, but also by parties that are unknown upfront. Therefore, (1) it should be **described** what the result of the execution of the service will be, and (2) it should be defined how the service can be called. WSDL [16] is a very basic language for fulfilling both goals. Once the service is described, its description should be made available somewhere. The classical solution to this problem is to offer access to the description via a globally accessible database, namely a UDDI [17] registry. In such a registry, anyone can enter information about the web services he offers, and anyone can **search** (and find?) web services.
   UDDI is definitely not the best means for realising web service discovery. For one thing, the fact that the UDDI registries are accessible to anyone has resulted in a pollution of these registries (e.g. through the insertion of non-working services)[ii]. Above this, not everyone wants a global discovery infrastructure, i.e. some parties (or should we say *most* parties?) only want to use services offered by trusted partners, and some service-providers only want their services to be known to a specified group of parties. Therefore, they could start using 'private UDDI' [11] or WSIL (Web Service Inspection Language) [18]. WSIL relies on a more decentralised mechanism for finding web services, this in contrast to the centralised UDDI. Note that the Internet is a totally decentralised system, over which no one has complete control. Would any enterprise want a few big companies – those who house the global UDDI registries (like Microsoft and IBM) – to get a hold on their way of doing business?
   Web service description and discovery through WSDL and UDDI relies on a human in the loop to realise the integration. As stated, an integration technology would become more powerful if it would allow (at least to some extend) the automated discovery and binding of web services. However, for this purpose computers should be able to understand the *meaning* of concepts. The problem of semantics is a very important one when considering B2Bi. Pollock [15] states that most problems contributing to the high failure rates of integration projects are not

technical in nature, but logical. Computers should be able to *understand* the information they get. In a B2B context, this problem becomes more apparent as one company may for example use the term 'clientcode' to denote the concept 'customerno' in another organization. In the web services domain, the problem of semantics shows in describing web services and their parameters. DAML-S (DARPA Agent Markup Language-Services, [19]) is a standard under development that aims at describing web services in a semantically rich way. Note that DAML-S builds on DAML+OIL, and is as such related to semantic web efforts. Paolucci Massimo et al. [20] are developping a matching engine for searching web services based on DAML-S.

2. As we said, a web service description should define the way a service can be called. To ease this task, it is preferable to have a standard way to call web services (and to get a response from called services). This way, the web service description can be restricted to a minimum. Besides this, it should be possible to call a web service from any platform and from any programming language, no matter the platform the service is running on or the language it is programmed in. Sending SOAP [21] messages (usually over HTTP) is *the* standard when it comes to **communication** with web services. As such, SOAP settles communication problems between JavaBeans, CORBA components and DCOM components. Note that SOAP is typically used over HTTP, avoiding the firewall problems CORBA's IIOP was confronted with.
   As already stated, SOAP, WSDL and UDDI are three important standards in the web services arena. In fact, these three industry standards form the basic web services stack, as depicted in Figure 1. To make web services technology more powerful, the stack has to be expanded, i.e., functionality has to be added. Myerson [22] gives an overview of some proposed (expanded) web services stacks.

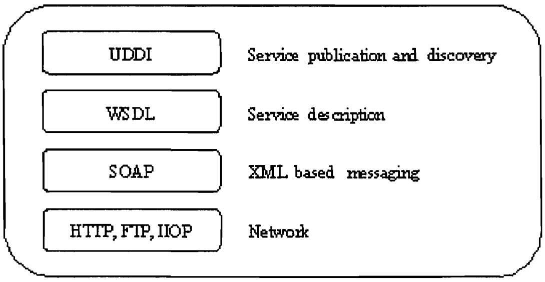

Figure 1: basic web services stack.

3. Currently, web services are mostly used for information exchange. However, if web services technology is to be *the* technology for B2Bi, it should also allow for the realisation of business **transactions**. With a 'business transaction' we mean the all-or-nothing situation of a traveller for example who wishes to go on a trip and wants to book a flight as well as a hotel room. Such a traveller should not only be able to *book* a flight and to *book* a hotel room, but also to *undo* a flight booking when he cannot find a hotel room, or vice versa. Unfortunately, realising

transactions in a B2B context can get complicated. For one thing, the use of classic locking-protocols is not always realistic, as companies do not want other companies to have a lock on their data and as the completion of transactions might take quite some time. These transactions are often called 'long running' or 'long-lived' transactions: '*business processes that run over an extended time period*' [23]. It might for example take 24 hours before the hotel booking gets confirmed, which complicates the booking of the airplane seat.
While transactions are traditionally characterised by the ACID (Atomicity, Consistency, Isolation and Durability) properties, these properties (especially the Isolation-property [23]) might be relaxed in a B2B context.
The realisation of transactions through web-services is the subject of many research and standardisation efforts. Relevant standards are the Business Transaction Protocol (BTP, [24]) and 'WS-transaction' [25]. Choreography-languages (see below) also keep the realisation of transactions in mind.

4. One of the biggest challenges in the B2B domain is the offering of services with a coarse-grained functionality, i.e. services that are **composed** of several other services. These smaller services are then called in parallel or in sequence and the call may be dependent on some conditions. Also, the *big* service may use *small* services of different companies. Note that the choreography should allow (and take care of?) the realisation of transactions.
Obviously web services technology gets more powerful if a *dynamic* coupling of services is possible. Such flexibility could be reached in two ways. First, the specific parties offering the services and the implementation of the specific services could remain undefined until the time of execution (for example by using *roles* that offer services). Secondly, the choreography itself could be composed as late as possible. Currently, many languages are under development to describe the choreography of web services. Recent initiatives include BPML (Business Process Modeling Language, [26]), WSCI (Web Service Choreography Interface, [27]), WSCL (Web Services Conversation Language, [28]) and BPEL4WS (Business Process Execution Language For Web Services, [29]). Although many standards exist, a good solution for realising a flexible choreography of services is still in the future. Note that the dynamic composition of services largely relies on semantics.

5. Currently, the *supply* of web services is not really focussing on the *demand* of web services. People searching a UDDI registry for some service might not find it (although it could be there in some other form, see below), while the service-providers do not get any information (feedback) on the services that are being sought. Suppose supplier A is offering a web service for looking up the available stock of a product of which the service-requestor specifies the productnumber. However, company B (a customer of A) wants to know which of A's products currently have a stock larger than 10 units. It is clear that both services, the one offered and the one needed, are quite related to each other, although there is no perfect match between both. Therefore, it should be possible to tailor the services to the wishes of the service-requestor. This may be realised by offering requestors the possibility to **customize** (themselves) a generic web service. This may include the adaptation of a choreography of web services.

6. **Reliability** is one of the key elements in doing business. Not only should service-providers be trustworthy, the services they offer should also be reliable. This is not too big a problem if the service providers are well-known partners. However, our goal is to achieve a global, generic structure for B2Bi, not assuming all business contacts are well-known companies. Consequently, the web services infrastructure should offer the possibility to *manage* the reliability of providers and services in one way or another. Service Level Agreements and certificates may play their part in this matter.

7. Communication between the systems of different companies could be conducted over a (virtual) private network, but will most probably use the public Internet. In this public domain, it is important to take care of **security**-measures. The techniques that may be applied to this purpose include authentication, encryption and digital signatures. 'WS-security' [30] is a recent standard in this area.

8. The infrastructure should be **scalable** and **manageable**, and should offer possibilities for **testing** and **debugging**. Scalability is needed at the level of the implementation of the web service (as the use rate of the service may fluctuate tremendously) and at the level of the *discovery-infrastructure* (which may become very complex in the future when related to customisation of services and the like). The management, the debugging and the testing of web services are challenges that become particularly difficult when composing web services out of smaller web services, offered by different parties.

9. In the digital era, companies want to get rid of paper versions of **business documents** (such as invoices). One could easily digitise (XML-ize) the currently used documents. However, this would be a burden on interoperability, as documents would be so diverse that companies would not be able to automatically process the incoming documents. After all, the meaning of the terms in the documents is not clear to computers (the problem of semantics described above). To solve this problem, it seems interesting to create standard documents. However, to be broadly applicable, a standard should take into account the different needs of different companies and different industries. For that reason, it seems more reasonable to prescribe some document schemas and to describe reusable business document components. xCBL (XML Common Business Library, [31]) and UBL (Universal Business Language, [32]) are two relevant standards in this domain.
   Note that web services actually implement some kind of RPC (Remote Procedure Call) mechanism by sending SOAP-messages. Currently, these SOAP-messages are not directly linked to business documents. Rather, they contain parameters to call the service. Time will tell how business-documents will find their way into the web services world. One way to think about this domain is in terms of 'business events', which could be fired and cause the execution of some web services. Note that the concept of business events can be used to bridge the gap between business people and ICT-experts (who know the concept of event handlers).

10. The solutions for the above-presented challenges become more powerful if they can be used in combination. B2B-frameworks, such as ebXML (e-business XML, [33]), **combine different standards** to resolve diverse B2Bi challenges.

CONCLUSION

Through the years, IT architectures have evolved from mainframes with terminals to distributed systems. A flexible and cheap integration technology makes it possible to easily integrate systems belonging to different parties. Web services technology, although still in its infancy, promises well for the future. To ease the B2Bi, companies should be able to rely on well thought-out standards, reducing the need for involvement of IT-personnel in integration efforts. As stated in the introduction, web services technology might give business people more power over business processes. From the above discussion it is clear that this is a far-away dream, as the whole web services domain is still unstable. Even the three basic standards - SOAP, WSDL and (especially) UDDI - still have to prove they are indispensable.
To realise flexible IT-systems, web services alone will not do the trick. One of the most important aspects in integrating systems concerns semantics. The W3C has frequently been criticised for its semantic web efforts because it concerns a vision that is hard to realise and (consequently) may not pay off in the short term. However, the importance of a semantic web of data and services is becoming clearly visible.

ACKNOWLEDGMENTS

This paper has been written as part of the 'SAP-leerstoel'-project on 'Extended Enterprise Infrastructures' at the K.U.Leuven, sponsored by SAP Belgium.

[i] The term 'software agent' is not unequivocally defined. Wooldridge [14] describes a software agent as '*a computer system capable of autonomous action in some environment*'.

[ii] One may even notice the arising of a new form of SPAM. It is not difficult to imagine an advertised 'get stockquote' service that actually returns the message 'Get real-time stockquotes from *xxx* for $1 a day', instead of the actual stockquote.

REFERENCES

[1] Ambler Scott (2002), Deriving Web services from UML models, http://www-106.ibm.com/developerworks/webservices/library/ws-uml1/.

[2] Hildreth Sue (2002), B2B web services projects still on the horizon, http://b2b.ebizq.net/ebiz_integration/hildreth_5.html.

[3] Gunjan Samtani (2002), Integration Brokers and Web Services, http://www.webservicesarchitect.com/content/articles/samtani03print.asp.

[4] Patricia Seybold Group (2002), An executive's Guide to Web Services, How to Optimize Web Services Investments to Improve Your Customer Experience, www.psgroup.com.

[5] IBM, http://publib-b.boulder.ibm.com/Redbooks.nsf/RedbookAbstracts/sg246407.html?Open

[6] W3C (2002), Web Services Architecture Requirements, http://www.w3.org/TR/2002/WD-wsa-reqs-20020429.

[7] Leymann Frank, Roller Dieter (2002), Business processes in a Web services world, http://www-106.ibm.com/developerworks/webservices/library/ws-bpelwp/.

[8] Linthicum, David (2000), B2B Application Integration: e-Business-Enable Your Enterprise, pp 464, Addison Wesley.

[9] DiGenova Ron (2001), Value-chain Optimization: Finding Real Profits Through Collaboration, http://www.supplychainebusiness.com/archives/12.01.opinion2.htm?adcode=30.

[10] Gunjan Samtani (2001), EAI and web-services, http://www.webservicesarchitect.com/content/articles/samtani01print.asp.

[11] Rick van der Lans (2002), SAI-workshop, Webservices Hét nieuwe webparadigma.

[12] Glass Graham (2000), The web services (r)evolution, part 1 - Applying web services, http://www-106.ibm.com/developerworks/webservices/library/ws-peer1.html.

[13] Fensel D., Bussler C. (2002), The web service modeling framework WSMF, http://www.cs.vu.nl/~dieter/ftp/paper/wsmf.pdf.

[14] Slides 4th European Agent Systems Summer School (Bologna), 8-12 July 2002.

[15] Pollock Jeffrey (2002), Dirty Little Secret: It's a Matter of Semantics, http://eai.ebizq.net/str/pollock_2a.html.

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

[17] UDDI.org, www.uddi.org.

[18] Web Services Inspection Language (WS-Inspection) 1.0, http://www-106.ibm.com/developerworks/webservices/library/ws-wsilspec.html.

[19] DAML Services, http://www.daml.org/services/.

[20] Paolucci Massimo et al. (2002), Semantic Matching of Web Services Capabilities, http://www.daml.org/services/ISWC2002-Matchmaker.pdf.

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

[22] Judith Myerson (2002), Web services architecture - how they stack up, http://www.webservicesarchitect.com/content/articles/myerson01.asp.

[23] Roxburgh Ulrich (2001), BizTalk Orchestration: Transactions, Exceptions, and Debugging, http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnbiz/html/btsorch.asp.

[24] Business Transaction Protocol, http://www.oasis-open.org/committees/business-transactions/documents/primer/Primerhtml/BTP%20Primer%20D1%2020020602.html.

[25] Web Services Transaction (WS-Transaction), http://www-106.ibm.com/developerworks/webservices/library/ws-transpec/?dwzone=webservices.

[26] BPMI.org, http://www.bpmi.org/bpml.esp.

[27] Web Service Choreography Interface (WSCI) 1.0, http://www.w3.org/TR/wsci/.

[28] Web Services Conversation Language (WSCL) 1.0, http://www.w3.org/TR/wscl10/.

[29] Business Process with BPEL4WS: Understanding BPEL4WS, Part 1, http://www-106.ibm.com/developerworks/webservices/library/ws-bpelcol1/.

[30] WS-security, http://msdn.microsoft.com/ws-security. .

[31] xCBL, http://www.xcbl.org/.

[32] UBL, http://www.oasis-open.org/committees/ubl/.

[33] ebXML, http://www.ebxml.org/.