Towards Dynamic Integration: Deployment of UDDI Registries in a Service-Oriented Architecture

Adam Nowicki, Andrzej Niesler

Department of Management Information Systems Engineering
Wrocław University of Economics
ul. Komandorska 118/120
53-345 Wrocław, Poland
{adam.nowicki, andrzej.niesler}@ae.wroc.pl

Abstract: The important premise of adopting service-oriented architecture (SOA) is the ability to increase enterprise agility through composition and reuse of coarse-grained services. However, without dynamic integration and alignment with business processes SOA would be nothing but yet another abstract architecture proposal. The response to that issue is Universal Description, Discovery and Integration (UDDI) protocol and deployment of UDDI registries. Several years of building universal and publicly available business registry showed, that the global monolithic approach is not what the market is expecting. This paper discusses the problem of dynamic integration in an SOA-enabled enterprise and provides insight on the reasons why the decentralised approach and private registries should be considered as an alternative solution. A proposal of the enhanced Web services registry model has been presented.

1 Introduction

Adaptability is a necessary quality in an ever-changing business environment. Through the recent years of increasing globalisation and pervasive connectivity, it has become extremely important for the companies to evolve their information systems in order to exploit new opportunities and face the upcoming competitive challenges. While many of these enterprise systems have been successfully transformed from centralized to distributed environments, they still lack the standardised outside world connectivity. Therefore, the new information architecture is developed mainly towards openness and interoperability. The most desired attribute of modern enterprise nowadays is the ability to develop new or redesign existing business processes in a highly flexible manner. Performing such a combination of technological and organisational changes is to result in a more agile enterprise–i.e. enterprise being capable of cooperating dynamically with its partners and adapting easily to the changes in the business environment.

The novel view on the IT architecture extends commonly accepted process-orientation with another level of abstraction–the layer of services. The service-oriented architecture (SOA) is based on the assumption, that essential enterprise functions and resources can be grouped together into atomic units of business functionality, called services. These services are then to be exposed as the fundamental constituents for developing both internal and external business processes. They are loosely-coupled and coarse-grained building elements, as opposed to the generic tightly-coupled and fine-grained functions (or interfaces) of enterprise applications. Such an approach may yield significant design benefits, like better flexibility or scalability. A service can be reused and redesigned many times, and that kind of change does not affect any of the corresponding processes. Furthermore, services are also programmatically discoverable, which means that they can be used to create the really dynamic and automated business processes.

In this paper we focus mainly on the issue of dynamic integration in an SOA-enabled enterprise. That encompasses such facets as discovery, composition and invocation of simple and compound services. Some of the approaches to service discovery and integration have been revised and compared. Furthermore, a service registry has been analysed, as the most universal and advanced discovery solution. We also discuss the reasons why the global monolithic approach has not been accepted by the market, and what kind of solutions should be considered in the future. Finally, a proposal of an enhanced Web service registry model has been provided as a conclusion.

2 Web services in an SOA-enabled enterprise

2.1 Defining Web services

The concept of Web services is still quite new and remains under constant development. As is usual in such cases, many definitions of the term can be found, and different characteristics are considered crucial. Web services are also very commonly confused with the SOA model itself (which, indisputably, is a much broader category,) and these terms are often referred to interchangeably. For the purposes of this paper, we define Web services as a framework that enables interoperation of business software components, running in a distributed heterogeneous computing environment using public Internet-based infrastructure and common open standards and protocols. The proposed definition is consistent with the World Wide Web Consortium (W3C) terminology and does not imply the use of any particular implementation technology.

As a very close realisation of the SOA concept, Web services are based on the same premises and aim at the similar business goals. The main category is a 'web' service, which is a self-describing piece of code, that can be published, discovered, invocated, and used as a part of the higher-order composition. Such approach supports creating modular and reusable business components, and allows shifting from code to service reusability. It facilitates the company to redesign or extend ably the existing computing scenario in accordance with emerging business needs. Moreover, as was mentioned before, Web services should rely only on open, publicly available standards and should not be vendor-specific. Because of that platform and vendor independence, they may be considered as a universal and flexible solution for either enterprise internal and external (i.e. business to business) systems' integration. In this paper, we ponder on that versatile nature of Web services, rather than focusing on any specific kind or implementation.

2.2 Web services as a reference IT architecture for modern enterprise

Web services as a framework, and SOA as a computing paradigm can be considered both a universal add-on integration tool and a reference IT architecture. The former approach has been proved successful in many integration scenarios, mainly while integrating variety of legacy systems. In such cases, Web services act as a middleware layer, simply wrapping legacy application interfaces and providing necessary outside-world connectivity. Despite the usage of the service-aware protocols, the result of such integration is yet predominantly nothing but a sophisticated point-to-point connection. The latter approach implies, that the service orientation should be implemented in the most, if not all, of the enterprise systems. It requires dedicated infrastructure, that supports messaging communication for applications and services, a registry to publish and discover services, and other tools–e.g. for providing centralised management capabilities. The range of changes to perform may be wide and comprise cost-intensive tasks, but most of the work can be conducted in an evolutionary and incremental way.

Investing in the SOA technology can be very beneficial for the company and may result in better flexibility and lower redundancy on IT resources. It can't be achieved by a one-off purchase, on the contrary, in most cases service-orientation requires appropriate strategic planning and the long-term investments. According to a recent IDC survey of more than 280 U.S. organisations, reported in [Du06], more than one-third had, in 2006, SOA projects or application-level initiatives under development or planned for the next 12 months. The SOA-driven software spending worldwide is estimated to reach nearly $9 billion by the end of 2009. As the numbers show, the Web services technology has been accepted by the market and reached sufficient maturity level for wide deployment. For many enterprises, launching own service-oriented infrastructure seems to become a necessity nowadays, because sooner or later it will be required in order to cooperate with their present and future business partners.

2.3 Web services and business process management

The main objectives of Web services deployment in a modern enterprise are to improve flexibility in managing company's IT assets and provide a platform for collaboration with its business partners. Service orientation enhances the overall scalability and helps to hide the increasing level of complexity of the underlying systems. It allows to streamline business processes without delving too deeply into technical details, and supports automation of formerly manual tasks. The true potential of this technology is in enabling insight into functionality of enterprise systems from the business-process perspective, and providing means for dynamic integration with outer systems.

Business Process Management (BPM) refers to a field of knowledge and a category of products that concern modelling, execution, monitoring and automation of enterprise business processes. From that perspective, Web services provide a particularly well-suited architecture for supporting implementation of the enumerated activities. In a service-oriented approach, services are filling the gap between business processes and functions of enterprise applications and systems. By defining a layer of arbitrarily-grained services, it becomes possible to model business processes and monitor business activities in a very flexible and almost unrestricted way. The foundation of Web services communication model is the system of exchanging XML messages, therefore, it is relatively easy to retrieve a necessary control information in every phase of execution of a business process. The new possibilities of process automation can be explored due to dynamic discovery and integration capabilities.

Web services deliver solid model for process decomposition and assembly. Through the associated technology, like Business Process Execution Language (BPEL), it is possible to execute a composite service, that can interact with external entities. BPEL provides means for combining simple services into compound ones, and allows to orchestrate the overall process by managing the exchange of messages and monitoring the control flow. A composite service can then act again as a simple one, and be reused as a part of another, higher-order compound service. The BPM-enabled Web services infrastructure should support writing the type of long-running asynchronous processes, that perform execution of external services and are able to provide only limited information on the current state of the whole process. In order to take full advantage of the automated collaboration, some technological prerequisites have to be fulfilled.

2.4 Technological foundations of Web services

Although the proposed definition doesn't impose the use of any specific technology, Web services may yet be considered as a set of technological standards, that implement the main guidelines of the service-oriented IT architecture. These standards form a protocol stack, which serves as SOA technological framework. The number of abstract layers and corresponding protocols is increasing steadily, as the whole architecture constantly evolves. However, there are four major standards that provide the core functionality, i.e. defining, describing, accessing, and discovering of Web services. Those are, respectively, XML (Extensible Markup Language), WSDL (Web Services Description Language), SOAP (Simple Object Access Protocol), and UDDI (Universal Description, Discovery and Integration). On the ground of the basic components, more sophisticated activity, like e.g. specifying and orchestrating composite services, or managing their semantic context, can be developed. A good example of the protocol implementing such a higher-level functionality is already mentioned BPEL.

The foundation of the whole framework is XML meta-language. XML is used for both, describing the data being transmitted and defining higher-level protocols operating on it. Due to its flexible and extensible nature, it is possible to represent almost any information in a human-readable and hierarchically structured plain text XML document. The standard facilitates the exchange of any structural and semi-structural data in a heterogeneous and distributed computing environment, and is, therefore, very suitable as the universal data description format. Besides the data, it is the web service interface that has to be explicitly defined and described. That kind of metadata is stored in the related WSDL file, and includes such information as the name of the service, possible transport bindings and expected message format. The well-described and subsequently published service can be invoked using SOAP protocol. In many specific implementations there are other protocols that can be used, e.g. SOAP over JMS.

3 Dynamic discovery and integration of Web services

3.1 Different approaches to the discovery and integration of Web services

The Web services interoperability depends on weather a service can be discovered and integrated into application. In different computing scenarios, the discovery process can be performed in a more or less dynamic way. Considering the most static approach, all what an application has to be aware of, is the location of WSDL files, containing all the necessary information about how to invoke services and communicate with them. The main cost of that kind of simplicity is the necessity of distributing the description files to many different places every time a single change has been made. It is also very error-prone, barely scalable and rather difficult to keep the files up-to-date, if the number of nodes becomes significant. The typical reasonable improvement of such an approach is building a centralised Web services metadata repository. It solves the data updating problem and brings in some new features. The contents of the repository can be easily accessed through a variety of different protocols, in particular via the standard internet web page (HTTP/GET) or file server (FTP), which can be accessed regularly in order to get the latest list of WSDL files. Furthermore, it introduces a simple dynamics to the discovery process, as the services can be added, modified or removed from the central repository and the discovery routine on the application side does not change.

The so far presented implementations of services discovery and integration do not reflect the real expectations and needs of emerging global distributed computing environment. One of the main promises announced by the Web services believers was to deliver the real loosely-coupling between the producer and the consumer of the service, and that requires true dynamic binding capabilities, not just limited to the enterprise local systems only. Moreover, exposing services to the company's business partners, or even to the public domain, requires considering many additional aspects of the discovery process, including, but not limited to, remote access policy, activity monitoring, or other security issues, like levels of authorisation and privacy.

In response to these issues, the metadata repository has been superseded by the concept of a service registry. Both terms appear to be quite similar, and usually the only distinction between them may be the fact, that registries store metadata and references to services, while repositories contain the services themselves. However, the actual set of features depends on the implementation, and though most of the registries available on the market are offering many advanced functions, it is not very unlikely to find the comparable level of functionality in a product named repository.

3.2 The UDDI-based service discovery

The UDDI protocol is the foundation in creating the truly interoperable, dynamic Web services environments. In default of any better alternative, it is a dominant standard for building registries of business services. The UDDI registry maintains a central database of services and provides standard API for performing queries on the data. The service providers register their services with the registry, by publishing both the services technical description and personal information about the service owner. The stored data may be classified by many different characteristics, e.g. service type, geographical location or provider information, which allows to formulate more specific queries. UDDI is based on a common set of industry standards and provides means for building registries dedicated to either publicly available services and services exposed only internally within an organization [U04].

3.3 The UDDI-based service registry model

The main constituents of the UDDI V3-compliant registry are the core and extended data structures, and the standard node and client API sets. The entities: businessEntity, businessService, bindingTemplate and tModel are connected with the 'contains' association, and represent, respectively, a service publisher, a particular family of services, technical information about a service entry point and implementation specs, and descriptions of specifications for services or value sets [U04]. The publisherAssertion entity describes possible relationships between businessEntities, and subscription entity represents the extended mechanism for tracking changes in the chosen data structures. Figure 1 illustrates the information model and standard API sets of the UDDI version 3-compliant registry.

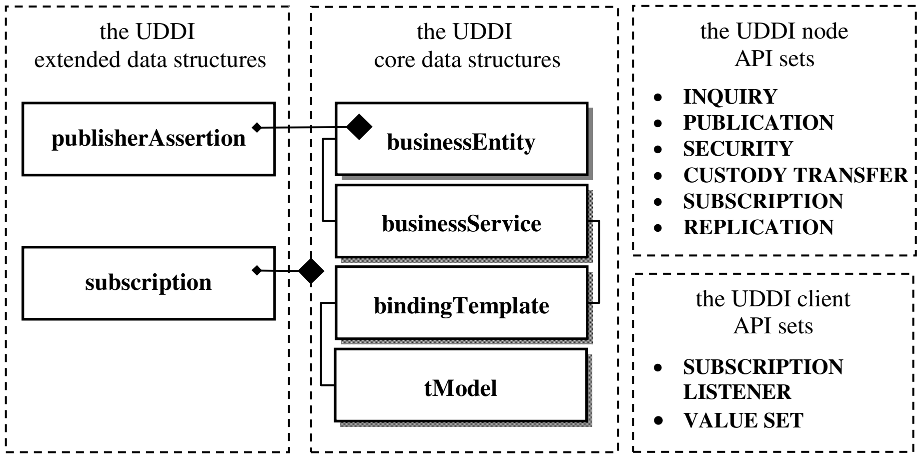

Figure 1: The UDDI V3-compliant registry information model and API sets

Through the functions organised in API sets, it is possible to operate the registry, i.e. to publish services, perform queries, replicate data structures between nodes, and so forth.

3.4 Replication, distribution and registry federations

The UDDI registry may consist of one or more nodes, and can be part of a group known as 'affiliation'. The purpose of such grouping is to permit controlled copying of core data structures between registries in the same group. The model of the data partitioning, proposed in the UDDI standard, is based on replication, instead of distribution. The approach maintains the overall scalability of the registry structure and preserves data integrity. It also enables inspection of the data in UDDI registries along multiple perspectives. However, it is quite reasonable to consider distribution, as it appears to be a more natural way to organise the registry data. The distributed, hierarchical structures reflect more appropriately such facets, as geographical location of service provider, branch of industry or business activity classification [Ou05].

Despite some technical and organisational problems, it is quite clear, that the future in development of global registry infrastructure is heading towards decentralised, distributed approach. Whether it will be a hierarchical structure or a federation of autonomous cooperating registries, shall depend on the particular business conditions.

4 The Universal Business Registry vs. private registries

4.1 The Universal Business Registry

According to the latest web information, the Universal Business Registry (UBR) was launched in 2000, with the primary goal to provide a central master directory of publicly available Internet-based services. The first operators of the global registry were IBM, Microsoft and SAP, later joined by the fourth operator, NTT Com. The main advantages of running the project for several years were defining consecutive versions of the UDDI protocol (up to version 3.0), and proving its strength and interoperability through a shared implementation and operational reliability. Although the practical demonstration of the technology itself was unquestionably successful, the UBR had never won a broader market acceptance. As the operators claim, the registry in its best time was managing over 50,000 replicated entries. Most of these entries, however, were actually nothing but a virtual data, entered by the developers for testing purposes. The project has been officially shut down in January 2006, i.e. after five years from the launch date.

The example of UBR initiative reflects the early approach to the issue of Web services discovery and integration. The idea was to create a solid public environment for automated discovery and execution of business transactions, encouraging companies to show interest in emerging electronic marketplace and new information technologies. The core component of the environment was meant to be a global, centralised services repository, maintained by several selected operators, but remaining monolithic through the mechanism of data replication. In that scenario, it was possible to truly realise the concept of universal service discovery, i.e. all what an application was needed to know, was the address of at least one of the top level registries, and the whole worldwide spectrum of Web services become accessible. For some reasons, which are discussed in the next section, it turns out, that the global monolithic approach is to be superseded by the locally managed private registries or the business domain oriented federations of registries. As long as there will not be any technical solution to link these separate directories into a searchable structure, the UBR primary idea of the global automated Web services discovery does not seem to be possible to realise. Another issue is whether that kind of functionality meets the current needs of the market.

4.2 Drawbacks of global monolithic approach

The first approach to dynamic discovery and integration assumed that there will be a large number of services, provisioned by companies worldwide and exposed to almost everyone interested. However, as the business reality showed, in most cases the amount of initially formed services was very low and barely justifying the usage of the service registry. Moreover, most of Web service application was not intended for public use, and, therefore, companies had no interest in registering them with the global registry. In normal business practice, there is required a specific minimal level of reliability to develop meaningful business relationship. The UBR did not provide any mechanisms to verify neither credentials of the information publishers, nor the relevance or validity of their services. That led to the situation, in which the registry was offering services, that do not work, or their publishers did not even exist [HR04].

The lack of reliability was not the only security-related problem of the UBR project. There were no widely adopted Web services security protocols and standards [Ch05]. The typical approach was to adapt existing security technologies, e.g. HTTPS for securing transmission channels, or implement a proper cryptography solution to the registry, to provide authentication, non-repudiation, privacy, and so forth [BCF04]. The version 3 of UDDI protocol introduced digital signatures, which made the data integrity and authenticity verification possible to conduct in a standardised way. The security issues will become more and more important, which will result in propagation and development of such specifications, like WS-Security, WS-Authorization, WS-Privacy, WS-Policy, WS-Trust or SAML (Security Assertion Mark-up Language).

Another, in many cases primary, reason for poor interest in deploying publicly available global registry was the scope of usage of the new technology. Many organisations concentrated only on providing integration between internal legacy systems, and did not intend to expose anything to outside world, or use any external Web services. In this scenario, the most suitable solution was a private registry–the whole sensitive data remained under full control of the company and only selected functionality was provided to a specific group of authorised users [MS03].

Private registries become more and more popular, and may be considered a standard component of the Web services environment. The cost of running a registry is relatively low, especially in comparison with the benefits of centralised management and achieved service governance capabilities. Once a company publishes its Web services in a private registry, the usage of alternative solutions lose significance.

4.3 The challenges to the private registries

The ever-growing popularity of private registries rises the problem of the direction, in which the Web services discovery and integration should evolve. As the global, publicly-available directories become rather obsolete, the concepts of semi-private and federated registries should be taken into consideration. In order to realise the vision of fully automated and dynamic Web services integration, the first and probably most challenging task will be providing mechanisms to perform queries and to interchange the data between autonomous private registries. Forming a federation of registries will allow companies to share their data while maintaining their privacy [Ou05].

The next challenge concerns the process of Web services composition. It should be possible to create, register, discover and invoke compound services on the same basis as the simple ones [ZCZ05]. Therefore, a registry should provide a module for composing complex services, and the execution engine to orchestrate invoked processes. The functional capabilities of the query processor depends on the quality of available taxonomies, which may include diverse sets, e.g. geospatial classifications.

4.4 Features comparison and future deployment

Table 1 presents comparison of selected features of the different types of Web service registries. It confronts global approach with private one, represented by federated and local registries. The last row contains present and future anticipated deployment areas.

Table 1: Features comparison for the different types of Web service registries

Features	**Type of registry**		
	Global	**Private-Federated**	**Private-Local**
Accessibility	unlimited scope	business domain	enterprise
Maintenance & control	restricted to several global operators	full control of one node, delegation of rights	full control & maintenance
Number of services	very large	large to medium	small
Discovery routine	universal, simple query	distributed, recursive query	local, simple query
Security mechanisms & constraints	no internal mechanisms for credential verification	domain specific security policy	local security policy or rules
Deployment	root nodes for global network of registries	domain-restricted usage for business collaboration	internal usage, integration with fixed partners

The changes made in the last version of the UDDI protocol show, that the direction of future deployment of Web service registries is going towards federation, rather than forcing global monolithic approach. Despite of the data replication problem, one of the most important development issues is enhancing registry interface search capabilities.

5 Three-tier model of the enhanced Web service registry

Considering the discussed issues, a proposal of the enhanced Web services registry model has been formed. Figure 2 illustrates three main tiers, i.e. the data tier, the logic tier and the interface tier. In the data tier, besides the standard UDDI core data structures, there is a set of taxonomies, classification schemas and identification systems. In order to perform semantic queries, there must be also a set of appropriate ontologies.

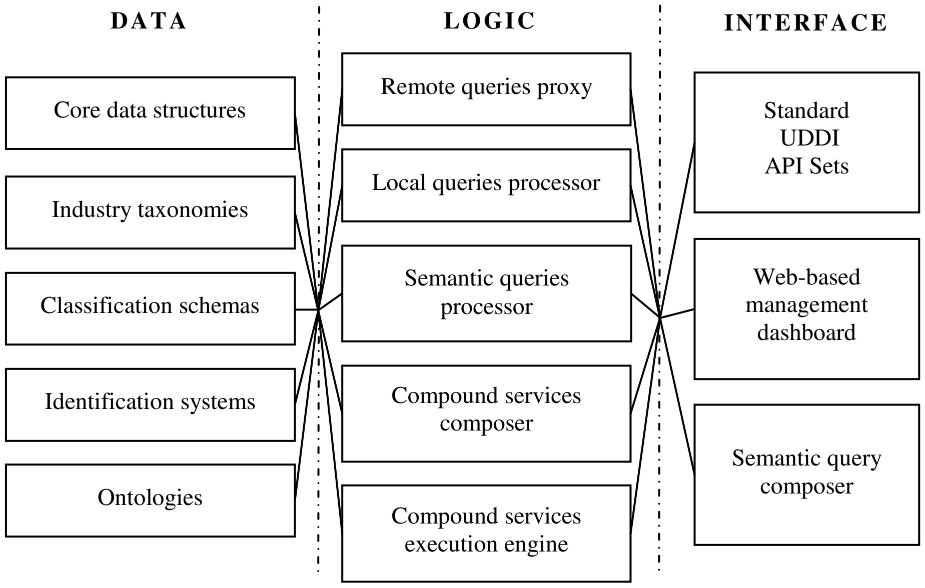

Figure 2: Three-tier model of the enhanced Web services registry

In the logic tier, there are modules for processing local and remote queries, and a dedicated processor for translating semantic queries. The compound services composer and execution engine, along with the ontology and managerial interface, allow for manual and semi-automatic Web services composition. The registry is also accessible programmatically through the standard UDDI API sets. The semantic query composer provides web-based, human-friendly interface for constructing queries, that are executed subsequently by the semantic processor using the registry ontology sets.

The model depicts future anticipated role of the Web service registry in an SOA-enabled enterprise integration scenario. The registry should be considered as a universal tool for dynamic discovery, supporting preferably both, automated machine-to-machine interaction, and semi-automatic integration performed by humans. Beyond its primary function as the repository of services, it now becomes the compound services' composition and execution unit and provides means for business process management. As the number of services grows, the crucial feature becomes the ability to perform advanced queries. In order to provide an effective query mechanism for humans as well as machines, it is necessary to establish appropriate taxonomies, classification schemas, and ontologies. That will enable the truly dynamic integration environment.

6 Conclusions

The purpose of this paper was to discuss the problem of dynamic integration in an SOA-enabled enterprise. With the Web services as the implementation technology, it becomes quite clear, that the real dynamic integration depends on dynamic discovery capabilities of the service-oriented infrastructure. We considered essential discovery scenarios, and pointed deployment of the UDDI-based registry as the most advanced and progressive approach. Several years of building the Universal Business Registry showed, that the private registries, local and federated, are more aligned with the present business needs and have been better accepted by the market. However, in order to achieve the true business processes management capabilities, some important architectural changes have to be made. The provided model addresses most of the discussed issues.

Acknowledgements

The research referred in this paper was supported by the State Committee for Scientific Research (KBN) in Poland, under Grant No. 1 H02D 086 30 for the years 2006/2007.

References

[BCF04] Bertino, E.; Carminati B.; Ferrari E.: Merkle Tree Authentication in UDDI Registries. International Journal of Web Services Research; Apr-Jun 2004; 1, 2; ABI/INFORM Global.

[Ch05] Chen, M.: An analysis of the driving forces for Web services adoption. Information Systems and E-Business Management, 2005, ISeB 3: 265-279 DOI 10.1007/s10257-005-0013-6.

[Du06] Dubie, D. et.al.: 6 Hot technologies for 2006. In: Network World; Jan 9, 2006; 23, 1; ABI/INFORM Global pg. 32.

[HR04] Hartman, F.; Reynolds, H.: Was the Universal Service Registry a Dream? SOA Web Services Journal. SYS-CON Media; Dec 2, 2004; http://www.sys-con.com/

[MS03] McGovern, J.; Sameer T.: Java Web Services Architecture. Morgan Kaufmann Publishers, 2003.

[Ou05] Oundhakar, S. et.al.: Discovery of Web Services in a Multi-Ontology and Federated Registry Environment. International Journal of Web Services Research; Jul-Sep 2005; 2, 3; ABI/INFORM Global.

[U04] Universal Description, Discovery and Integration Version 3.0.2 Technical Report, OASIS UDDI Spec TC, 2004. http://uddi.org/pubs/uddi_v3.htm.

[ZCZ05] Zhang, D.; Chen, M.; Zhou, L.: Dynamic and Personalized Web Services Composition in E-Business. Information Systems Management, Summer 2005; 22, 3; ABI/INFORM Global pg. 50.