A Conceptual Architecture for Semantic Web Enabled Web Services

Christoph Bussler
Oracle Corporation
500 Oracle Parkway, Redwood Shores, CA 94065, USA
chris.bussler@oracle.com

Dieter Fensel
Leopold Franzens Universitaet Innsbruck, Institut fuer Informatik
6020 Innsbruck, Austria
dieter.fensel@uibk.ac.at

Alexander Maedche
FZI Research Center for Information Technologies at the University of Karlsruhe
76131 Karlsruhe, Germany
maedche@fzi.de

ABSTRACT

Semantic Web Enabled Web Services (SWWS) will transform the web from a static collection of information into a distributed device of computation on the basis of Semantic Web technology making content within the World Wide Web machine-processable and machine-interpretable. Semantic Web Enabled Web Services will allow the automatic discovery, selection and execution of inter-organization business logic making areas like dynamic supply chain composition a reality. In this paper we introduce the vision of Semantic Web Enabled Web Services, describe requirements for building semantics-driven web services and sketch a first draft of a conceptual architecture for implementing semantic web enabled web services.

1. INTRODUCTION

Semantic Web Enabled Web Services (SWWS) will transform the Web from a static collection of information into a distributed device of computation on the basis of Semantic Web technology making content within the World Wide Web machine-processable and machine-interpretable. Over the past few years, we have seen a growing interest in the potential of Web Services and the use of more machine-processable and machine-interpretable content on the Web, involving the development of ontologies. At the same time, to date, nearly nothing has been done to combine the full power of both of these approaches.

The application of ontologies[1] is increasingly seen as key to enable semantics-driven data access and processing. There are many applications of such an approach, e.g. document and content management, information integration or knowledge management, to name just a few. Within the Semantic Web ontologies play a central role being conceptual models for describing machine-processable and interpretable metadata.

[1] An ontology is a shared and machine-executable conceptual model in a specific domain of interest.

Web service technology as it is today may profit from semantics-driven data processing within different aspects. Figure 1 depicts the Web service stack associated with semantics. Within SWWS we consider semantics as a vertical layer that may be exploited by the horizontal layers service description, publishing, discovery, trading partners as well as service flow and composition.

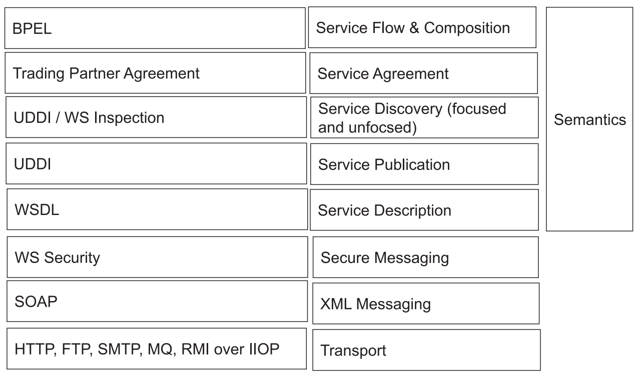

Figure 1: Web Service Stack and Semantics

The future vision of E-commerce on the basis of Web services is that anybody must be able to trade and negotiate with everybody else. However, such an open and flexible approach to E-commerce has to deal with many obstacles before it becomes reality. To bring Web services and fully enabled E-commerce to reality, mechanized support is needed in dealing with numerous and heterogeneous data formats as well as numerous and heterogeneous business logics. Machine processable semantics of information is a key essential element in mechanizing these tasks. Therefore, beside exploiting semantics for the basic Web service stack, we also emphasize that semantics is essential within heterogeneous environments allowing for flexible mediation on a conceptual basis.

To summarize, fully enabled E-commerce based on workable Web Services requires a conceptual modeling framework and a associated conceptual architecture on top of existing Web Service standards and architectures that is centered around two complementary principles: First, one requires a strong de-coupling of the various components that realize an E-commerce application. This de-coupling includes information hiding based on the difference of internal business intelligence and public message exchange protocol interface descriptions. Second, strong mediation service enabling anybody to speak with anybody else in a scalable manner are required. This mediation service includes the mediation of different vocabularies as well as the mediation of different interaction styles.

Organization. The paper is organized as following: Section 2 defines requirements for SWWS. As a basis for the conceptual framework we shortly describe in Section 3 the Web Service Modeling Framework (WSMF) building on and extending recent standard proposal like WSDL and ebXML. Section 4 introduces the conceptual architecture that underlies our overall approach. Section 5 concludes with a short summary and an outlook on future work.

2. SWWS REQUIREMENTS

This section introduces our vision of Semantic Web Enabled Web Services by providing the following B2B use case: A newly hired employee requires a laptop for his workplace. In order to buy one he defines the characteristics of the laptop like processor speed and disk size. The purchasing process makes sure that the new employee's manager authorizes the purchase. Based on the configuration of the laptop, alternative hardware vendor's offerings have to be collected and a price comparison has to be done. Once the cheapest laptop is determined, a service contract for three years has to be found and purchased together with the laptop. The service contract might be from the laptop vendor or might be from an independent insurance company. It might be that a same or better-equipped laptop together with the service contract is cheaper then if both are bought separately. Once the cheapest combination is found, the laptop purchase order is issued. Finding alternative laptops and service contracts as well as the final purchase should be supported fully automated without human user involvement.

The mediation related problems that have to be solved within this use case are:

- The whole process has to be modeled and has to be executed as a business process. This includes human as well as non-human tasks.
- Transmissions over networks have to be secured according to the trading partner requirements and the B2B protocols involved.
- Possible laptop vendors as well as service contract vendors have to be discovered and messages have to be exchanged with them.
- Different document type formats have to be understood and transformed into each other to make the prices and products comparable. This is one of the essential requirements for explicit representation of semantics through ontologies.
- Alternative purchase approaches are necessary in case a separate company buys the service contract because in this case purchases, the laptop and the service contract have to succeed both or none at all.

Various aspects can be identified as being necessary to achieve such an intelligent and automatic web service discovery, selection, mediation and composition into complex services.

Document types. Document types describe the content of business documents like purchase orders or invoices. The content is defined in terms of elements like an order number or a line item price. Document types are instantiated with actual business data when a service requester and a service provider exchange data. The payloads of the messages sent back and forth are structured according to the document types defined. Ontologies are the ideal concept to describe the structure and semantics of document types.

Syntax. Documents can be represented in different syntaxes available. XML is a popular syntax, although non-xml syntax is used in many cases, too (e.g. EDI [www.x12.org]).

Semantics. First, the elements of document types must be populated with correct values so that they are semantically correct and are interpreted correctly by the service requesters and providers. This requires that a vocabulary is defined that enumerates or describes valid element values, for example, a list of product names or products that can be ordered from a manufacturer. Further examples are unit of measures as well as country codes. Second, "rules" may allow generating consistent values automatically. Ontologies [2] provide a means for defining the "concepts" and therefore the semantics of the data to be exchanged. If ontologies are available then document types refer to the ontology concepts. This ensures consistency of the textual representation of the concepts exchanged and allows the same interpretation of the concepts by all trading partners involved. Finally, the intent of an exchanged document must be defined. For example, if a purchase order is sent, it is not necessarily clear if this means that a purchase order needs to be created, deleted or updated. The intent needs to make semantically clear how to interpret the sent document.

Transport binding. Several transport mechanisms are available like HTTP/S, S/MIME, FTP or EDIINT. A service requester as well as a service provider have to agree on the transport to be used when service requests are executed. For each available transport the layout of the message must be agreed upon and how the document sent is represented in the message sent.

Exchange sequence definition. Communication over networks is currently inherently unreliable. It is therefore required that a service requester and a service provider make sure themselves through protocols that messages are transmitted exactly once.

Process definition. Based on the assumption that messages can be exchanged exactly once between service requester and service provider the business logic has to be defined in terms of the business message exchange sequence. For example, a purchase order might have to be confirmed with a purchase order acknowledgment. These processes define the required business message logic in order to derive to a consistent business state.

Security. Fundamentally, each message exchange should be private and unmodified between the service requester and service provider as well as non-reputable. Encryption as well as signing ensures the unmodified privacy whereby non-repudiation services ensure that none of either service requester or service provider can claim not to have sent a message or a different one.

3. THE MODELING FRAMEWORK WSMF

This section introduces the Web Service Modeling Framework (WSMF) that provides the model for developing and describing web services and their composition. The philosophy of WSMF is based on the two following principles *(i)* maximal de-coupling, *(ii)* complemented by scalable mediation service. A model in WSMF consists of four main different elements:

- **Ontologies** that provide the terminology used by other elements;
- **Goal repositories** that define the problems that should be solved by web services;
- **Web services descriptions** that define various aspects of a web service; and
- **Mediators** which bypass interoperability problems.

As mentioned earlier ontologies are a key enabling technology for the Semantic Web. They interweave human understanding of symbols with their machine-processability. In our framework, ontologies are used to define the vocabulary that is used by other elements of WSMF specifications (document types). Therefore, they enable reuse of terminology as well as interoperability between components referring to the same or linked concepts.

The description of a goal specifies objectives that a client may have when he consults a Web service. Goal specifications should be kept separate from actual web service descriptions because there is an *n2m* mapping between them, i.e., the same web service can serve different goals and obviously different (competing) web services can serve the same goal. For example, Amazon.com could be used to buy a book, however, in the same way it can be used as an information broker on bibliographic information about books. Conversely, different bookstores may subscribe to the same goal.

Many Web service description languages distinguish between elementary and complex Web services. Elementary Web services are simple input/output boxes, whereas complex Web services break down the overall process into sub-tasks that may call other web services. WSMF identifies more than ten aspects in describing such Web services.

Mediating components are of general importance for component-based software development. [6] introduced an adapter pattern in his textbook on design patterns for object-oriented system development. Such adapters enable reusable descriptions of objects and make it possible to combine objects that differ in their syntactical input and output descriptions. [4] introduced the concept of an adapter in architectural descriptions of knowledge-based systems to (1) de-couple different elements of this model, (2) encapsulate these different elements and (3) explicitly model their interactions. Work on software architectures describes systems in terms of components and connectors that establish the proper relationships between the former (cf. [9]). Here, the focus is to mediate between different interaction styles of components (which we call the business logic of a web service). Finally, work on heterogeneous and distributed information systems developed the concepts of wrappers and a mediator. Instead of assuming a global data schema, heterogeneous information and knowledge systems have a mediator [11] that translates user queries into sub-queries on the different information sources and integrates the sub-answers. For an open and flexible environment such as web-based computing, adapters are an essential means to cope with the inherent heterogeneity.

In this paper we will not further focus on WSMF, we refer the interested reader to [3, 2].

4. CONCEPTUAL ARCHITECTURE

In this section we introduce the SWWS Conceptual Architecture that has been designed to enable semantics-driven Web Service applications. Figure 2 depicts the overall SWWS conceptual architecture.

In our conceptual architecture we distinguish between

- SWWS front-end tools that allow for modeling, deploying, simulating semantics-based Web services and for administrating and managing the overall system.
- SWWS components that provide the required functionality for realizing Semantic Web enabled Web services.
- SWWS storage and external components allow *((i)* to persist and to access the data mainly relying on an ontological representation and *(ii)* to access external engines such as a workflow engine.

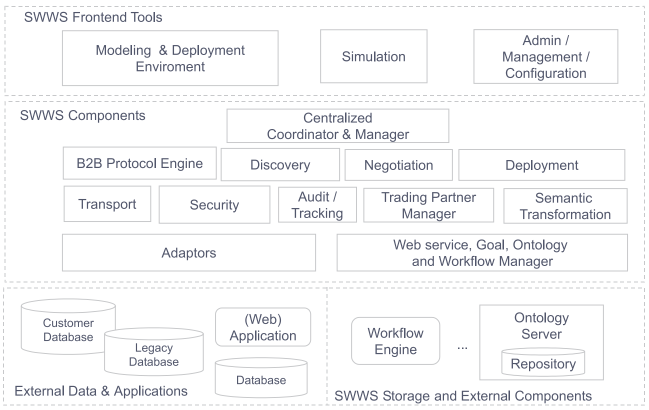

Figure 2: SWWS Conceptual Architecture

In the following we discuss only the SWWS components with an emphasis on the ontology and mediation functionality.

4.1 SWWS Components

SWWS core components provide the required functionality for realizing Semantic Web enabled Web services.

Centralized coordinator and manager. This component is, as it names says, a management component that manages all other components in the components layer (like a runtime system). It makes sure that incoming requests are route appropriately between the component so that the requests are executed according to the execution semantics as described in the WSMF [3].

Semantic transformation. Semantic transformation will be supported by this component on different levels: Data, Business Logics, Message Exchange Protocols and Dynamic Service Invocation. Subsection 4.2 will describe this component in more detail.

Discovery. Within SWWS and its underlying de-centralized model and complex semantics-based service descriptions, further means for discovery will be provided. The discovery component allows to support discovery of SWWS. In its simplest form it connects to a centralized UDDI repository. However, according to the P2P paradigm a centralized UDDI repository should prevented and therefore discovery within SWWS is a challenging task.

Audit/tracking. Trust is a central feature for the successful application of Web Service technology. This component allows to track the different service providers and consumers to provide means for auditing, etc.

Trading partner manager. To provide means for tracking, the trading partner management component stores trading partner profiles including identification, name, address, security keys and identifiable trading partner endpoints. Furthermore, it includes trading partner agreements (e. g. exchange particular type of content over specific types of protocols using a particular transport.)

Negotiation. The negotiation component exploits the semantics available within SWWS to support two trading partners negotiating and agreeing on businesses. Match-making is part of the negotiation component, where support for automatically matching service offers with service requests is provided. It provides means for computing semantic similarity along different dimensions between two SWWS-based complex services (cf. [10]).

Security. Clearly, a component for authorization and authentication of users storing user profiles and their credentials has to be provided. Furthermore, security also is of highest importance for messages, including channel security management like SSL and message security via encryption/decryption, signing of messages or sequence numbers.

Workflow, Web service, Ontology, Goal Manager. The managers are APIs that allow for standardized and simplified access to the underlying data and processes. Thus, they hide the overall complexity by providing a proper defined and extensible application programming interface.

B2B Protocol engine. B2B protocols are a defined sequence of message exchange over networks, including support for quality of service, reliability, security, etc. Typically a protocol consists of several layers, including information about document formats, sequencing, packaging, transport binding. Examples of protocols are RosettaNet, OAGIS, and ebXML.

Deployment. Web Service deployment is the process of making the defined services available for service consumers. This component supports easy deployment of the defined services onto different platforms like J2EE or .NET.

Adapters. Adapters enable reusable descriptions of objects and make it possible to combine objects that differ in their syntactical input and output descriptions. Adapters can be distinguished in *(i)* protocol adapters implementing existing protocols like EDI, SWIFT or RosettaNet. *(ii)* Application adapters that adapt to legacy system or existing home-grown systems (legacy and non-legacy systems). *(iii)* New adapters that support an adaptation friendly way using interface standards like WSDL or J2EE Connector Architecture.

Transport. The transport component implements the network and protocol connectivity. It implements protocols like HTTP, SMPT, FTP as well as other protocols like VANs (Value added networks) or SWIFT (protocol of the banking industry).

4.2 Ontology and Mediation Components

This subsection describes how Semantic Web technology is exploited within SWWS with respect to the ontology and mediation component.

Ontology Support in SWWS. Section 2 mentioned two SWWS concepts that require ontology support: document types and semantics. Document types are the definition of the structure of the business data that are communicated between trading partners. Document types refer to concepts, attributes of concepts and relationship between concepts that are ideally managed by and represented within ontologies. Ontologies can in a very precise manner define as well as manage concepts, attributes and relationships between concepts.

Semantics refers to the correct population of attributes with correct domain values. Not every value that an attriubute can contain is semantically correct. For example, if a purchase order contains the reference to a model then this reference (in terms of a model number) must match a laptop model. A model number for a desktop or a mainframe would be syntactically correct, but not semantically.

In the architecture ontology servers and associated repositories are used for storing and accessing domain and application concepts (e. g. product names, measures and country codes) represented in the form of ontologies. The technical representation of ontologies will be tightly aligned with recent W3C Semantic Web standardization activities[2] such as RDF(S) and the ongoing standardization initiatives like

[2]http://www.w3.org/2001/sw/

OWL. Implementation of the ontology engine will be provided as described in [8].

Mediation Support in SWWS. The WSMF assumes a peer-to-peer communication structure between trading partners. This means that trading partners must be able to communicate directly with each other without any intermediary between them. Consequently, SWWS must be able to connect trading partner with each other. Of course, intermediaries are possible (e. g. marketplaces or hubs). However, in this case there is again a P2P communication structure between a trading partner and the intermediary.

A consequence of the point-to-point structure is that each involved trading partner needs to be able to mediate differences in document types, semantics or even business process behavior. An internal definition of a document like a purchase order (e. g. dictated by an application system) might not correspond exactly with the one that is exchanged between the trading partners. [2] enumerates all possible mismatches that can exist. The same applies for the semantics where values differ. A simple example is the internal convention to use long state names like 'California' whereby the short version like 'CA' is used for the document exchanged.

In addition, business processes can mismatch between trading partners. One might use acknowledgments internally whereas none are used between trading partners when exchanging messages. Again, [2] outlines all possibilities.

In order to mediate between the differences the semantic transformation component is included in the SWWS architecture that addresses these mismatches and allows for the mediation of those. The semantic transformation engine provides for each possible mismatch a rule set that overcomes the mismatch. For example, 'CA' is transformed into 'California' in the case of document content. In case of mismatch of processes it either generates the missing acknowledgments or consumes superfluous ones.

The underlying idea of ontology-based mediation is that we lift as well as the schema as the data onto a conceptual level in the form of an ontology and associated instances. Subsequently, we provide means for declarative ontology mapping associated with procedural processing information. A more detailed description how mediation of business communication on a conceptual level is achieved is described in [7].

4.3 Using the Conceptual Architecture

In this subsection we shortly introduce a round-trip example that shows how to use the SWWS Conceptual Architecture for a Purchase Order (PO) and Purchase Order Agreement (POA) use case. In the laptop purchasing use case described in the beginning of Section 2 the final activity is to send out the purchase order after all approvals and alternative product comparisons took place. The sending (i. e. buying) company is the "buyer" and the laptop vendor is the "seller" in Figure 3. This particular message exchange is described in the following in context of the architecture. Figure 3 is an alternative representation of Figure 2 that only shows the for the example relevant components.

Once the decision has been made which laptop to purchase, a purchase order is signed-off within the enterprise resource management system (ERP). The signed-off purchase order is extracted from the ERP system by an appropriate adapter. Once the adapter has the purchase order (PO) it is audited in the audit component. This is necessary to later on being able to prove that the PO was received by the SWWS architecture. After the audit of the PO happened, the PO is given to the coordinator that coordinates the subsequent processing of the PO. The workflow manager is notified about the PO in order to initiate any business process that might have to happen. In our use case all business processing took place already and no other activity then sending out the PO is to be done. The business process for this is very simple containing a send step in the business process that is responsible for sending out a PO. Once the workflow manager executed the business process the PO is given to the B2B engine in order to accomplish the message exchange with the trading partner ("seller"). The B2B engine looks up trading partner information of that trading partner to which the PO must be sent. This is done by invoking the trading partner management component. The lookup determines the appropriate trading partner agreement and the agreement contains a reference to the B2B protocol to be used. Based on this data the message format as required by the receiver is known. The transformation component is tasked to transform the PO message in the format as required by the receiving trading partner. This is done with the help of semantic transformation based on ontologies. At this point the mediation takes place. Once the transformation happened, the message is in the format according to the agreement with the trading partner. After the transformation the message is given to the transport component that is responsible for sending out the message. However, before the message is sent it has to be signed and encrypted (in most cases) to apply message security. This again is defined in the trading partner agreement. Once the message is all ready to be sent it is audited again fully secured in order to later on be able to prove that the sent happened. Finally, the message is sent to the seller. The seller in turn processes the message internally. This is invisible to the buyer and the buyer only knows that it needs to receive a return message at some point in time, the purchase order acknowledgment message (POA). Once the POA is received, the message processing takes place in the opposite execution order as indicated by

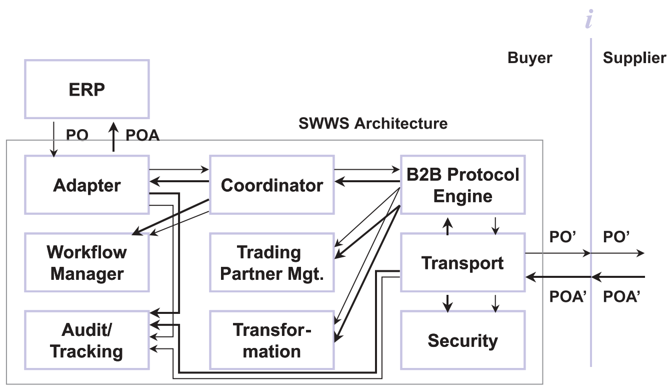

Figure 3: SWWS Scenario: Processing of two messages.

the thick sequence of arcs (not further discussed here).

5. RELATED WORK

Recently, it has been recognized that bringing semantics to web services is an essential step. However, there is not much active work in the field of semantics-based web services, especially when focusing of bringing semantic web and web service technology together.

The work that is closed to ours is done by the DAML-S consortium [1]. Our framework as well as DAML-S use the Semantic Web's key enabling technology of ontologies as their basis. In contrast to DAML-S, the underlying principles of our framework are the ideas of de-coupling and mediation. Furthermore, DAML-S misses important modeling constructs like the distinction between business logic and message exchange protocols. Thus, we allow for a more flexible framework including greater abstraction and encapsulation of proprietary business information. The vision is to establish a mediated P2P environment providing a maximum of efficiency and flexibility. A detailed comparison between DAML-S and our approach is provided in [5].

ebXML, being developed primarily by OASIS and the United Nations, approaches the overall problem from a workflow perspective. ebXML uses two views to describe business interactions, a Business Operational View (BOV) and a Functional Service View (FSV). The BOV deals with the semantics of business data transactions, which include operational conventions, agreements, mutual obligations and the like between businesses. The FSV deals with the supporting services: their capabilities, interfaces and protocols. Although ebXML does not concentrate on only Web services, the focus of this view is essentially the same as that of the current DAML-S effort. It has the concept of a Collaboration Protocol Profile (CPP) "which allows a Trading Partner to express their supported Business Processes and Business Service Interface requirements such that they are understood by other ebXML compliant Trading Partners", in effect a specification of the services offered by the Trading Partner. A Business Process is a set of business document exchanges between the Trading Partners. CPPs contain industry classification, contact information, supported Business Processes, interface requirements etc. They are registered within an ebXML registry, in which there is discovery of other Trading Partners and the Business Processes they support. In this respect, UDDI has some similarities with ebXML. However, in contrast to our work ebXML's scope does not extend to the manner in which the business documents are specified. This is left to the Trading Partners to agree upon a priori by the creation of a Collaboration Protocol Agreement. Thus, we claim that for real-world deployment, B2B protocols have to support document standards, business protocols, packaging, sequencing and transport combined with powerful transformation, workflow and adapter technology.

6. SUMMARY AND OUTLOOK

In this paper we have introduced the vision of SWWS. First steps towards accomplishing the vision have been done with a first conceptual model as described in [2] and a conceptual architecture as shown in this paper. The EU funded project SWWS[3] (IST-2002-37134) has been set up to implement a first incarnation of the vision.

Much work remains to be done in the context of SWWS. First, we will have to map the components of our conceptual architecture to concrete technical components. Furthermore, it is important to emphasize that our framework is working and aligned with existing standards and proposals like WSDL and upcoming standards like BPEL. In general we have to stress that the overall framework as well as the architecture focuses on being able to adapt to any standard out there because this is what mediation is all about.

7. REFERENCES

[1] A. Ankolenkar, M. Burstein, T. C. Son, J. Hobbs, O. Lassila, D. Martin, D. McDermott, S. McIlraith, S. Narayanan, M. Paolucci, T. Payne, K. Sycara, and H. Zeng. DAML-S: Semantic Markup For Web Services, online available at http://www.daml.org/services/. 2001.

[2] D. Fensel and C. Bussler. The web service modeling framework wsmf. In *White Paper and Internal Report Vrije Unversiteit Amsterdam, 2002, available at www.cs.vu.nl/ swws/download/wsmf.paper.pdf*, 2002.

[3] D. Fensel, C. Bussler, and A. Maedche. Semantic web enabled web services. In *Proceedings of the International Semantic Web Conference 2002, LNCS, Springer*, pages 1–2, 2002.

[4] D. Fensel and R. Groenboom. A software architecture for knowledge-based systems. *The Knowledge Engineering Review (KER)*, 14(3), 1999.

[5] A. Flett. A comparison of daml-s and wsmf. In *Internal Report, Vrije Universiteit Amsterdam*, 2002.

[6] E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design Patterns.* Addison-Wesley Publishers, 1995.

[7] A. Maedche, N. Silva, B. Motik, and R. Volz. MAFRA – A Mapping Framework for Distributed Ontologies in the Semantic Web. In *Proceedings of the 13th Iternational Conference on Knowledge Engineering and Knowledge Management, EKAW, Siguenza, Spain*, 2002.

[8] B. Motik, A. Maedche, and R. Volz. A conceptual modeling approach for semantics-driven enterprise applications. In *Proceedings of the First International Conference on Ontologies, Databases and Applications of Semantics, Irvine, USA, 2002*, 2002.

[9] M. Shaw and D. Garlan:. *Software Architectures.* Prentice-Hall, 1996.

[10] K. Sycara, J. Lu, M. Klusch, and S. Widoff. Matchmaking among heterogeneous agents on the internet. In *Proceedings of the AAAI Spring Symposium on Intelligent Agents in Cyberspace, Stanford, USA, 1999.*, 1999.

[11] G. Wiederhold. Mediators in the architecture of future information systems. *IEEE Computer*, 25(3), 1992.

[3]http://swws.semanticweb.org