Web Services Research Challenges, Limitations and Opportunities

FLORIJE ISMAILI
Computer Science Department
South East Eurpoean Univesity,
Campus Bldg. 305.20,Ilindenska nn, 1200 Tetovo
FYROM
f.ismaili@seeu.edu.mk

BOGDAN SISEDIEV
Faculty of Electronic Engineering and Technology,
Technical University of Sofia
Kliment Ohridski St. Sofia-1000
BULGARIA
Bogi@tu-sofia.bg

Abstract – Service Oriented Architecture (SOA) is an architecture style where software components that provide a piece of functionality communicate with each other via message they exchange. Within SOA these pieces are called services. Nowadays, the technology platform most associated with the realization of SOA is Web Services. Web Services have received much interest due to their ability to transcend programming language, operating system, network communication protocol, and data representation dependencies and issues.

In this paper we suggest a new XML & Web Services Framework which relies on the Service Oriented Architecture and is a way of reorganizing applications into a set of services.

Key-Words: - SOA, Web Services, Web Services Framework, Research Challenges

1. Introduction

Nowadays many distributed systems are inherently heterogeneous and geographically dispersed. Such systems emerge as a result from the cooperation between organizations inside the country or between different countries. The big question is how to improve the ability of these systems to meet new requirements. Adding a new user interface, combining multiple data sources into a single view, integrating mobile devices, or replacing an old application with a better one are common reasons for investing in new projects.

The paradigm of service-oriented development, although not new, is a term that represents a model in which automation logic is decomposed into smaller, distinct units of logic. Collectively, these units comprise a larger piece of business automation logic. Individually, these units can be distributed. Within SOA, these units of logic are known as services.

Services reflect a “service-oriented” approach to programming that is based on the idea of composing applications by discovering and invoking network-available services to accomplish some task. This approach is independent of specific programming languages or operating systems [7].

Web services are currently the most promising SOA based technology. They use the Internet as the communication medium and open Internet-based standards, including Simple Access Protocol (SOAP) for transmitting data, the Web Services Description Language (WSDL) for defining services, and the Business Process Execution for Web Services (BPEL4WS) for orchestrating services.

This means that when migrating your application architecture to an SOA, you are committing yourself to Web services design principles and the accompanying technologies as core parts of your technical environment. An SOA based on XML Web Services builds upon established XML technology layers, with a focus on exposing existing application logic as loosely coupled services. In support of this model, the design principles introduced by SOA emphasize reuse, statelessness, autonomy, abstraction, discoverability, loose coupling and composability.

For Distributed Database Applications to scale to the Internet there is a need for efficient integration in order to achieve interoperability. Distributed Database Applications integration frameworks still have several research issues which need to be addressed. These include process-based integration of services, dependable integration of services, support of standardized interactions, dynamically reconfigurable runtime architectures, Business-driven automated compositions, Self-configuring management services, security, and privacy.

In this paper we present a framework that addresses these issues in the development of a Web Services based application. We are focused on the design of a framework for developing web services in compliance with the appropriate standards. The general idea is that the user should be able to use the framework to deploy a series of pre-developed functions as a web service, also to enable the implementation, deployment, and the management of SOA based solutions.

The organization of this paper is as follows: first, we briefly discuss some of the research work related to Web services. In section 3, the basic Service Oriented Architecture is discussed. The Research Road Map is introduced in section 4. The brief introduction to Semantic Web is given in section 5. In section 6 is presented our new XML & Web Services Framework, following with the description of its components in section 7. In section 8 is given the strength of the framework and finally, the conclusion and future work can be found in section 9.

2. Related Work

Various applications in areas such as e-commerce finance, scientific computing and Grid computing have been exposed as Web Services, but there are open issues in Web Services that may limit their applicability in some situations. In order to find the open issues we have compared different papers.

Machado and Ferraz [9] proposed guidelines for performance evaluation of Web Services toolkits. But the use of the Web Services integration technology is not justified on its performance capabilities. Web Services is chosen in terms of interoperability, simplicity, flexibility, reuse of services.

Crasso, Zunino and Campo [10] presented search method for Web services called WSQBE, which combines standards for describing and publishing Web services with IR to filter available services automatically. The question raised is: what are WSQBE capabilities for query generation to consider parameter names and data-types when pulling out queries from source code?

Brocke and Linder [11] suggested a general framework for the evaluation of financial consequences of out-tasking decisions. But what will happen if additional efficiency measures are included?

Agarwal, Chafle, Mittal and Srivastava [12] classified and evaluated various approaches for web service composition and execution. The problem is that criteria such as complexity of coding and maintenance were not included.

Dietze, Gugliotta and Domingue [13] proposed Conceptual Situation Spaces (CSS) which are aligned to establish Semantic Web Services Standards, but it does not fully solve the issues related to symbolic Semantic Web (Services)-based knowledge representations.

Lee, Yang, Kim and Kang [14] developed an architecture which overcomes the drawback of request reply semantics of Web Services. The disadvantage is that it works on par with the message driven middleware solutions that are available in the market.

Differently of these studies, this paper presents a set of related open problems in Web Services and proposes a framework in order to overcome these problems.

3. The Basic Service Oriented Architecture and Web Services

The Basic Service Oriented Architecture is a way of reorganizing a portfolio of previously siloed software applications and support infrastructure into an interconnected set of services, each accessible through standard interfaces and messaging protocols [1]. This architectural approach is particularly applicable when multiple applications running on varied technologies and platforms need to communicate with each other.

Though the required implementation technology can vary, SOAs have evolved to a point where they can be associated with a set of common characteristics[3]:

- SOA is at the core of the service-oriented computing platform.
- SOA is fundamentally autonomous.
- SOA supports vendor diversity.
- SOA fosters inherent reusability.
- SOA is an evolution.

This is a list of the reasons why the IT community is going through the trouble of changing so much of its philosophy and technology in an effort to adopt SOA.

This list of common benefits is generalized and certainly not complete because SOA benefits organizations in different ways, depending on their respective goals and the manner in which SOA is applied.

SOA can result in the creation of solutions based on interoperable services which turns a cross-application integration project. It also promotes the design of services that are inherently reusable. The concept of composition is another fundamental part of SOA. This allows extension within each application environment.

The acceptance of Web Services technology by wide industry enables many legacy environments to participate in service-oriented integration architectures.

On its most fundamental level, SOA is built upon and driven by XML. As a result, an adoption of SOA leads to the opportunity to fully leverage the XML data representation platform [3]. A standardized data representation format can reduce the underlying complexity of all affected application environments.

Examples include:

- XML documents and XML Schemas (packaged within SOAP messages) are standard format for message passing between applications and typing of all data communicated. This allows extensible and adaptable communications network.
- XML's self-descriptive which the is potential for data to be read, maintained and understood easily.
- The standard format of data representation increases interoperability among different applications.

Another benefit of SOA is organizational agility. Agility is a quality inherent in just about any aspect of the enterprise [2]. So, a well designed SOA protects application during their evolution.

Finally, by abstracting business logic and technology into specialized service layers, SOA can establish a loosely coupled relationship among enterprise domains, which allows each enterprise to evolve independently according to requirements.

But, as any application, design or architecture, SOA has also its pitfalls. Following are descriptions of some of the more common mistakes [3]:

- Building service-oriented architectures like traditional distributed architectures.
- Not standardizing SOA.
- Not creating a transition plan.
- Not starting with an XML foundation architecture.
- Not understanding SOA performance requirements.
- Not understanding Web services security.
- Not keeping in touch with product platforms and standards development.

3.1 SOA Services as Web Services

A Web Service is an interoperable unit of application logic that transcends programming language, operating system, network communication protocol, and data representation dependencies and issues. It is an infrastructure for developing and deploying distributed applications [18]. Web Services are typically intended for the new generation of business-to-business (B2B) or enterprise application integration (EAI) applications [19].

To fully understand the importance of web services, you need to understand the requirements of distributed computing

Many reasons exist for distributing application logic. Some of the most important include the following:

- **High scalability**: By distributing the application logic, the load is spread out to different machines.
- **Easy deployment**: Pieces of a distributed application may be upgraded without upgrading the whole application.
- **Improved security**: Distributed applications often span company or organization boundaries.

The research increase about Web Services is as the result of their deliverance using technologies such as eXtensible Markup Language (XML), Web Services Description Language (WSDL), Simple Object Access Protocol (SOAP) and Universal Description Discovery and Integration (UDDI).

- XML: is an open standard developed by the WWW Consortium (W3C). It is a standard way for defining and describing data.
- WSDL: is an XML-based language that describes all Web Service's details. It describes the request message a client needs to submit to the web service and the response message the web service returns. It also defines the transport protocol you need to use (typically HTTP) and the location of the web service.
- SOAP: To communicate with a web service, you need a way to create request and response messages that can be parsed and understood on any platform. SOAP is the XML-based language you use to create these messages. When you run the application and actually interact with the web service, the client sends a SOAP message to trigger to the appropriate web method.
- UDDI: A standard for creating business registries that catalog companies, the web services they provide, and the corresponding URLs for their WSDL contracts.

Web services provide several technological and business benefits, a few of which include [20]:

- Application and data integration
- Versatility
- Code re-use
- Cost savings

Using XML technologies and HTTP as a transport protocol means that any application can communicate with any other application using Web services. They are versatile during their design, because they can be used by another Web-based application or by another Web Service. One service might be utilized by several clients which means saving time for creating custom service for specific requirements. All these benefits mean cost savings.

3.1.1 Service roles

A Web service is capable of assuming different roles, depending on the context within which it is used. For example, a service can act as the initiator-*service requestor*, the *service discovery agency* and the *service provider*. A service is therefore not labeled exclusively as a client or server, but instead as a unit of software capable of altering its role, depending on its processing responsibility in a given scenario [3].

Service provider hosts services which are accessible over the network. It also, defines a service description of the service and publishes it to a client or service discovery agency through which a service description is published and made discoverable. The *service requestor* search for services and, when found, a dynamic *binding* is performed. In this case, the service provides the consumer with the *service description agency* details and an *endpoint* address. The consumer then *invokes* the service.

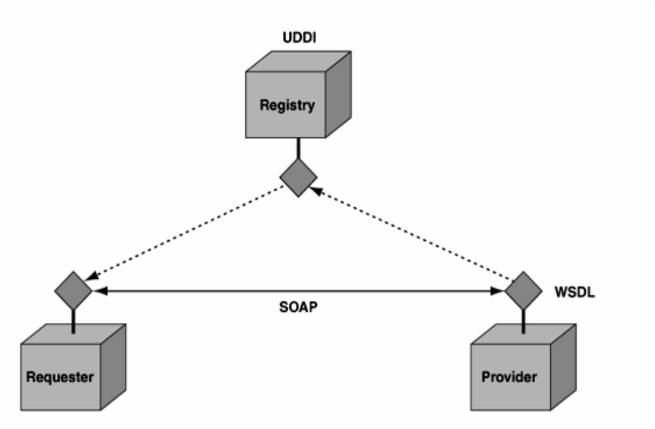

Figure1. Basic Web Services Architecture

4. Research Road Map

The Service-Oriented Computing (SOC) is the computing paradigm that uses web services as fundamental elements for developing applications/solutions. SOC research road map introduces an extended SOA that provides separate tiers for composing, coordinating and managing services. The bottom tier is called service foundations, the middle one is service compositions, and service management and monitoring on top [1][8].

This logical stratification is based on the need to separate

- basic service capabilities provided by a middleware infrastructure and conventional SOA from more advanced service functionality needed for dynamically composing services,
- business services from systems-centered services, and

- service composition from service management [15].

4.1 The Service Foundation
The service foundations plane consists of three kinds of participants: the service provider, the service discovery agency, and the service requestor (client). The interactions involve the publish, find and bind operations. This architectural approach is particularly applicable when multiple applications running on varied technologies and platforms need to communicate with each other. In this way, enterprises can mix and match services to perform business transactions with minimal programming effort [1][15].

4.2 The Service Composition
The service composition plane encompasses necessary roles and functionality for the consolidation of multiple services into a single composite service. Resulting composite services may be used by service aggregators as components in further service compositions or may be utilized as applications/solutions by service clients. Service aggregators thus become service providers by publishing the service descriptions of the composite service they create. A service aggregator is a service provider that consolidates services that are provided by other service providers into a distinct value added service. Service aggregators develop specifications and/or code that permit the composite service to perform functions that include: coordination, monitoring, conformance and QoS compositions [1][15].

4.3 Service Management and Monitoring
Service management functionality is aimed at supporting critical applications that require enterprises to manage the service platform, the deployment of services and the applications. Also, it may provide detailed application performance statistics that support assessment of the application effectiveness, permit complete visibility into individual business transactions, and deliver application status notifications when a particular activity is completed or when a decision condition is reached. Service monitoring involves monitoring events or information produced by the services and processes; monitoring instances of business process; and suspending, resuming, or terminating selected process instances [8][15][1].

Our proposed XML & Web Services will be based on the SOC research road map.

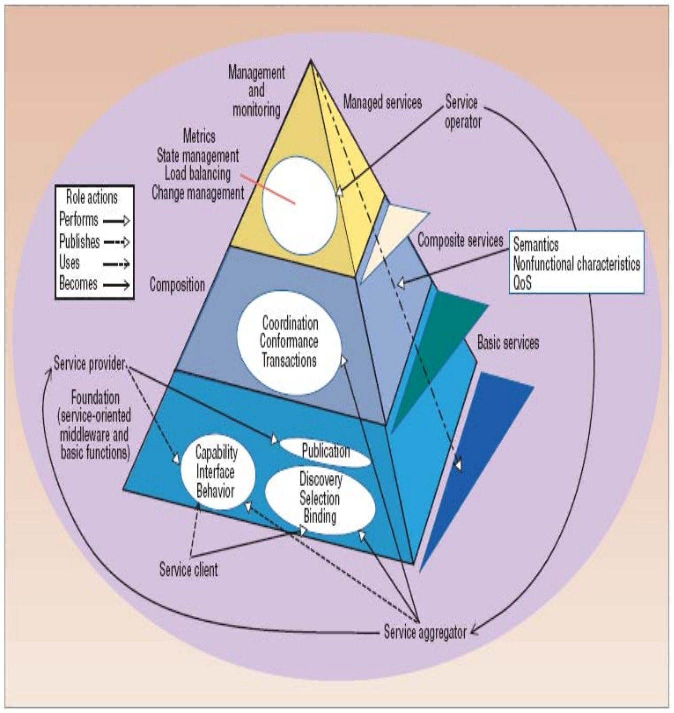

Figure2. SOC Research Road Map [15]

5. A brief introduction into Semantic Web Architecture
Understanding and addressing requirements for semantic technologies in core SOA is not simple. What requirements for languages, tools and processing at design time will dramatically improve productivity and reuse? How close a match is sufficient for real reuse when exact match fails?

In order to understand these questions, firstly a brief introduction to Semantic Web Architecture is given.

The Semantic Web vision significantly evolves the Web technology offering new technologies to the developers of Web-based applications aiming at providing more intelligent access and management of the Web in-formation and semantically richer modeling of the applications and their users [21].

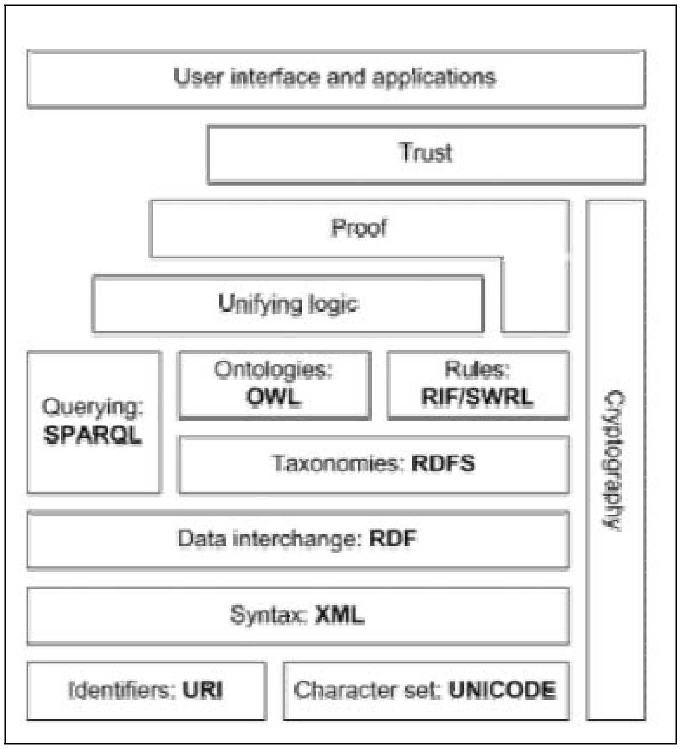

Figure3. Semantic web architecture in layers [22]

XML is the standard for data interchange on the web, but XML and their schema languages do not express semantics but rather structure. The RDF (Resource Description Framework) is used for the creation of metadata, enabling the addition of meaning to structured parts of information. It is an abstract model for metadata description and use. RDF/XML is an implementation of RDF using XML syntax. An RDF extension, RDF Schema (RDFS) allows the definition of classes and their hierarchical structure. Ontologies are used to define the basic terms and relations between domains. This is the mean for data description or metadata creation. Finally, OWL (Ontology Web Language) is used for ontology definition. [22].

The idea of semantic web is now becoming widespread and this is the reason why ontologies are also coming into the centre of interest. The main benefit of ontologies is in overcoming problems with semantic heterogeneities between data sources.

Semantic Web technology, namely ontology languages, reasoners and query languages, provides scalable methods and tools for the machine-accessible representation and manipulation of knowledge. Semantic Web Services (SWS) make use of Semantic Web technology to support the automated discovery, substitution, composition, and execution of software components, namely Web Services.

Our idea is to combine SWS with Research Road Map and to develop one consolidated technology.

6. Proposed XML& Web Services Framework

The core motivation for this research problem results from the major research challenges now and in the near future. In that sense several questions raises:

- What is the infrastructure support for data and process integration while designing and developing extensible applications?
- How to enforce service reuse and how to govern proper usage?
- What are the methods for creating a high performance SOA solution?

In order to answer these questions, we propose a new XML & Web Services Framework consisting of five layers: Data Layer, Data Integration Layer- Query Engine, Service Layer, Semantically Service Discovery Layer and Application Layer.

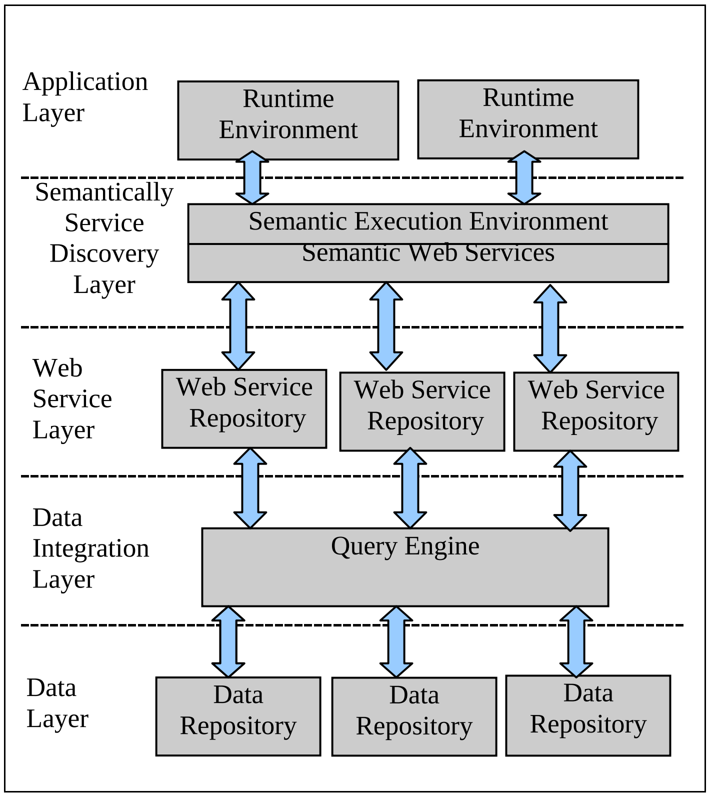

Figure4. XML & Web Services Framework

The usage of XML has increased dramatically and we can say that it has become a common part of database environments. But, still, there is a lack of understanding how and why XML can or should integrate with different relational databases because both of them are incompatible data platforms, created for different purposes [4]. The Data Integration Layer – Query Engine will combine different strategies for integrating XML and relational databases normally based on XQuery which enables the creation of data services, where the runtime infrastructure will abstract the complexity of underlying data sources and will provide consistent access to all data by all applications that require it. The Service Layer will aggregate multiple services into a single composite service which can be later used as a single service in another composite service, at the same time it should abstract away the logic at the application or business level, such as order processing, the implementation of transactions, security, and reliability policies. These services will be discovered by Semantically Service Discovery Layer by using automated means with minimal user involvement. This requires the addition of semantics in both service provider- Service Layer and service requester- Application Layer; also the Application Layer should describe the corresponding needs which are like the description of desired service in some formal language.

7. Components of the Architecture

Data Layer: represents the set of various resources to be integrated.

Data Integration Layer: will be ontology based data integration including construction, mapping between ontologies and data sources, and query processing. The ontology in our approach will serve as a global schema which presents a unified interface of a collection of data to be integrated.

To handle our requirements, the most adequate approach is Mediator- based Systems, where information integration is generalized by introducing two components: wrappers and mediators.

Every data source is connected to the integrator system via wrapper acting as an interface to handle technical differences. The mediator component accesses the wrappers and consolidates retrieved data. Access to the data access wrappers is realized via web services which provide methods for searching data. The mediator component is realized as an integration service. It locates data source services, retrieves the requested information and consolidates it to a result set.

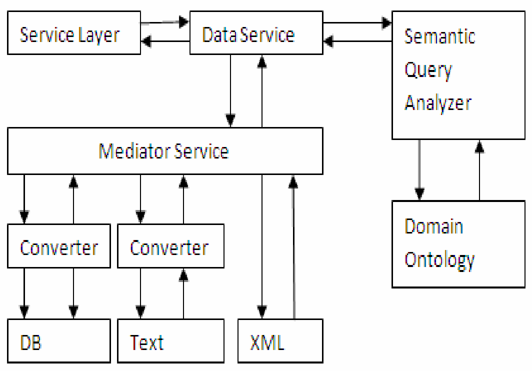

Figure5. Outline of the data integration

The Operating Architecture:
When a user submits inquiring words to user interface, the search process will be done passing through the following steps:

a) The system will send the query to Semantic Query Analyzer Service. The query analyzers will search in the field of ontology.

b) Semantic Query Analyzer Service will return the result to the Data Service.

c) Data Service delivers XML Query Language to Mediator Service, which will integrate all the heterogeneous information from the different sources in Data Layer.

d) The result will be sent to Data Service.

e) Data Service will return these results to Web Service Layer.

Web Service Layer: will be value added services by combining Data Services from Data Integration Layer. The process of web service composition consists of creation of a workflow that realizes the functionality of a new service. For example, a *travel plan service* can be developed by combining several elementary services such as *hotel reservation*, *ticket booking*, *car rental*, *sightseeing package*, etc., based on their WSDL description. Different web service compositions solutions have been proposed. However, in order to choose a suitable technique, one needs to systematically analyze the strengths and the weaknesses of each of these solutions. Such evaluation has been done by Agarwal, Chafle, Mittal and Srivastava [12]. They have present existing web service composition and execution approaches as four alternatives:

1. Interleaved Composition and Execution
2. Monolithic Composition and Execution
3. Staged Composition and Execution
4. Template-Based Composition and Execution

In this layer, we must choose an existing or develop a new tool for service composition which will cover the complete life cycle of service composition, including the description, planning, building and executing of compositions.

Semantically Service Discovery Layer: will efficiently find Web services from Web Service Layer according to Application Layer requirements.

Web service discovery is normally defined as a matching process in which available services capabilities can satisfy a service requester's requirements. The capability of a Web service is often implicitly indicated through a service's name, a method's name and some description included in the service. And this capability can be described as an abstract interface by using Web Services Description Language (WSDL). With the help of the standard descriptions of Web Services, various approaches can be used to find services.

Although various approaches can be used to locate Web Services, this research, at this layer, will be focused on the service discovery problem using clustering method and ontology-based method.

Clustering method [16] concentrates on web Service discovery with OWL-S and clustering technology, which consists of three main steps. The OWL-S is first combined with WSDL to represent service semantics before a clustering algorithm is used to group the collections of heterogeneous services together. Finally, a user query is matched against the clusters, in order to return the suitable services.

Ontology-based method [17] aims to make two XML documents interoperate at the semantic level while retaining their nesting structure. This architecture will integrate heterogeneous XML sources. The mediation integrates both the XML nesting structure and the domain structure expressed by RDFS to enable semantic interoperation between the XML sources by hiding their structural heterogeneities. This will be done by using an algorithm for translating the query back and forth between XQuery and RDQL.

At this layer this two methods will be combined, in order to establish semantic connections among the underlying composite services and **Application Layer**.

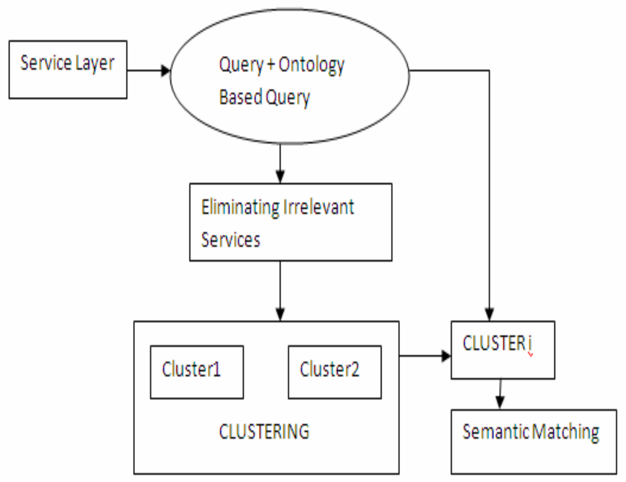

Figure6. Outline of the matching approach

Evaluation of framework implementations should be performed through the development of several web services, which form effective case studies of service development and provide live test data for functional testing.

The real-world use cases which are likely to benefit from this work are different IT infrastructures that are trying to assemble application components into loosely coupled network of services that can create dynamic business processes. Somme of them are: Financial Services, User Administration System, Scheduling System, Source Management etc.

8. The strength of the XML & Web Services Framework

1. A paradigm for uniting the diverse standards of XML-based Web Technologies like XML, Web Services and the Semantic Web by allowing them to be incorporated within a single document.

2. Using Semantic Technologies to address the search and integration challenges that services pose. But this is not an easy path. It requires much innovation and hard work.
3. Developing a new tool for service composition specification, construction and execution.
4. A novel approach for efficiently finding Web services on the Web.

9. CONCLUSION AND FUTURE WORK

Starting from the current interest in Web Services, we explored and compared the Web Services open problems. We discussed the characteristics of these problems and identified future research directions. At the same time we proposed new framework to find a response to the new key research questions posed above.

Further research should concentrate on the extension of the framework focusing on security and optimization problems as well as on its empirical use in practice.

References:

[1] M.P. Papazoglou, *Web Services: Principles and Technology*, Prentice Hall,2007.

[2] Tomas Erl, *SOA: Principles of Service Design*, Prentice Hall, 2007.

[3] Tomas Erl, *Service-Oriented Architecture: Concepts, Technology, and Design*, Prentice Hall**,** 2005.

[4] Tomas Erl, *SOA: A Field Guide to Integrating XML and Web Services*, Prentice Hall, 2007.

[5] F.Leyman, Combining Web Services and the Grid: Towards Adaptive Enterprise Applications, *Proc. CAiSE 05 Workshops*, vol. 2, 2005, pp. 9-21.

[6] T. Andrew, Bussiness Process Execution Language for Web Services, IBM Developers Work

www.ibm.com/developerworks/library/specification/ws-bpel.

[7] Eric Newcomer, Greg Lomow, *Understanding SOA with Web Services*, Addison Wesley, 2004.

[8] F.Leyman, Combining Web Services and the Grid: Towards Adaptive Enterprise Applications,

[9] A.C. C. Machado, C. Ferraz , Guidelines for performance evaluation of web services, *ACM International Conference Proceeding Series; Vol. 125*, 2005
http://portal.acm.org/results.cfm?coll=Portal&dl=GUIDE&CFID=70764332&CFTOKEN=75461838

[10] M.Crasso, A. Zunino, M. Campo , Query by Example for Web Services, *Proceedings of the 2008 ACM symposium on Applied computing*, 2008
http://portal.acm.org/citation.cfm?id=1114223.1114234&coll=Portal&dl=GUIDE&CFID=70764332&CFTOKEN=75461838

[11] J. Brocke , M.Linder, Service portfolio measurement: a framework for evaluating the financial consequences of out-tasking decisions, *Proceedings of the 2nd international conference on Service oriented computing*, 2004
http://portal.acm.org/results.cfm?coll=Portal&dl=GUIDE&CFID=70764332&CFTOKEN=75461838

[12] Agarwal, Chafle, Mittal, Srivastava, Understanding approaches for web service composition and execution, *Proceedings of the 1st Bangalore annual Compute conference*, 2008
http://portal.acm.org/citation.cfm?id=1341771.1341773&coll=Portal&dl=GUIDE&CFID=70764332&CFTOKEN=75461838

[13] Dietze, Gugliotta ,Domingue, Towards context-aware semantic web service discovery through conceptual situation spaces, *ACM International Conference Proceeding Series; Vol. 292*,2008
http://portal.acm.org/citation.cfm?id=1361482.1361488&coll=Portal&dl=GUIDE&CFID=70764332&CFTOKEN=75461838

[14] Lee, Yang, Kim, Kang, A Model for Application Integration using Web Services, *ACM International Conference Proceeding Series; Vol. 292*,2008

[15] Michael P. Papazoglou , Paolo Traverso , Schahram Dustdar , Frank Leymann , Service-Oriented Computing: State of the Art and Research Challenges, *IEEE International Conference Proceeding ,2007*
http://csdl2.computer.org/dl/mags/co/2007/11/mco2007110038.pdf

[16] R. Nayak and B. Lee, Web Service Discovery with Additional Semantics and Clustering, *In Web Intelligence, IEEE/WIC/ACM International Conference*, 2007.

[17]I. F. Cruz, H. Xiao and F. Hsu, An Ontology-based Framework for XML Semantic Integration, *Proceedings of the International Database Engineering and Applications Symposium*, IDEAS'04, 1098-8068/04,2004

[18] Gudivada, Nandigam, Enterprise Application Integration Using Extensible Web Services, *Proceedings of the IEEE International Conference on Web Services*

(ICWS'05), 0-7695-2409-5/05, 2008
[19]Lin, A Conceptual Model for Business-Oriented Management of Web Services, Proceedings *of the 6th WSEAS Int. Conf.* on *Software Engineering, Parallel and Distributed Systems*, Corfu Island, Greece, February 16-19, 2007
[20] Mahmood, Software Products and Technologies for the Development and Implementation of SOA, *WSEAS TRANSACTIONS on COMPUTER RESEARCH,* August 17, 2007
[21] Chiribuca, Hunyadi, M. Popa, The Educational Semantic Web*, 8th WSEAS International Conference on APPLIED INFORMATICS AND COMMUNICATIONS (AIC'08)* Rhodes, Greece, August 20-22, 2008.
[22] Linková, Zdenka. Integrace dat v prostredí sémantického webu. *Prague: Czech Technical University* (July 2008).
[23]F.Ismaili, B.Sisediev, Web Services- Current Solutions and Open Problems, *8th WSEAS International Conference on APPLIED INFORMATICS AND COMMUNICATIONS (AIC'08)* Rhodes, Greece, August 20-22, 2008.