A Survey of Quality Assurance Frameworks for Service Oriented Systems

Maria Allauddin, Farooque Azam , Mehmooda Jabeen Zia
Department of Computer Engineering, National University of Sciences and Technology (NUST),
College of Electrical and Mechanical Engineering (CEME)
Islamabad, 44000, Pakistan
Corresponding Author Email: maria09@ce.ceme.edu.pk

Abstract

Quality assurance has vital role in building a software system because it provides confidence and lowers the risks associated. Assurance in service-oriented systems is a challenging problem, which requires a flexible and dynamic solution. When we talk about quality of a service oriented system we have to consider all the included services that are interdependent to provide that service, including all the limitations of resources and runtime situation. In order to state different quality expectations in terms of specification and advertisement significant quality networks are still required. In the frameworks there is also a need to compare and select between alternative services. In this paper we will survey different quality assurance frameworks used in service oriented systems, to determine that which framework is the most efficient and reliable to face the challenges and implications of the characteristics of these systems.

Keywords: *Service Oriented, Quality, Assurance, Framework, Survey.*

1. Introduction

Quality has been typically defined as a degree of excellence, conformance to specifications, meeting requirements, distinguishing attribute, state of being free from defects, deficiencies, and significant variations for customer expectations. It is an active way to practically check and produce a product which contains minimum or no error and defect free.

B. W. Boehm et al. [1] says that in relation to software engineering, quality is defined as “A planned and systematic pattern of all actions necessary to provide adequate confidence that the item or product conforms to established technical requirements”. (IEEE Std730-1998).Software quality assurance (SQA) contains, different ways of having a continuous check on software engineering processes, and methods used to certify quality. Assuring and verifying quality of software has gain major importance in research since 1970s.

Today numerous software applications are there which are complex enough but play major role in many areas of life. It is clear that most reliable software applications are needed which are safe, dependable, and efficient.

The concept of service oriented software is stimulated by industry as the basis for the next computing age said by M. Turner et al. [3]. Service-oriented architecture (SOA) allows software systems to be dynamically composed and organized using services that are discoverable (T. Erl et al. [2]). Also T. Erl et al. [2] and I. Sommerville et al. [4] reported that conventional model of software deployment has substantial drawbacks as compared to

Service oriented systems deployment as these systems needs less cost investment and dynamic integration.

With time service oriented systems are growing to provide service in many terms of software products so the requirements for dynamic features are rising. Customers also consider the providers repute and cost spent for the service as quality measurements. Assurance of service-oriented systems is similar to assurance of any distributed software system. However, service oriented environments have new challenges, (A. Barbir et al. [5] and Y. Le Blevec et al. [7]), because of less control of organizations consuming the services, reduced visibility into infrastructure and services that may consist of independently developed products from different vendors and sources, a dynamic development and execution environment with frequently changing services their infrastructure and network loads, specific configurations of services, network and infrastructure that become available only at runtime.

The more distributed and loosely coupled a service-oriented system is, the harder it is to provide assurance, because of decreasing control. Assuring quality in modern service-oriented systems is a challenging problem, which requires a flexible and dynamic solution that can adapt to emergent properties in the runtime environment. Numerous quality frameworks are given to specify quality expectations. We will compare and observe similarities between various quality models given; we will review these and analyze them to find out the better framework. We will advance the research by keenly checking out relevant concepts that are already recognized for dependency and priority information for quality measurement.

1. Problem Description and Scope

The subject of Quality in SOA systems has dedicated many of the researches to its own. When an SOA system is running, it must make various tradeoffs among the quality attribute requirements as it selects services to be connected or disconnected. Quality in SOA systems is a combinatorial issue. The total quality of a system is quality attributes of all the services it is composed of. So for having quality management in this architectural approach, new tools, technologies and methods are needed to handle the run-time issues of quality.

To check the dependencies between different system services and the resources available, with their constraints with in the running environment there is still need for efficient quality frameworks. In the service execution environment there is a requirement for dynamic approach for quality assurance which must be able to find out and solve the evolving problems. In summary, present frameworks provide the customer (Daniel Robinson et al. [8]):

_ Poor support for expressing quality. Commonly used service description languages focus on the structural properties of a service, and provide little scope for expressing non-functional properties.

_ Partial control over service quality. Consumer of service has slight control over the quality of service due to third party characteristics of software services.

_ Poor support for execution quality. Quality during execution is currently checked with monitoring and reporting the failure this is not efficient. There is a need of effective negotiation and recovery techniques.

Present quality frameworks provide partial control on quality of the services oriented system. So there is still a space for some added strategies to assure quality of dynamic service oriented architecture. To find out the needs of supplementary quality assurance requirements we are going to survey different quality models that already exist. And then to find out their pros and cons then by analyzing them find out some better strategy for QA of Service Oriented Systems.

3. Survey of Models

3.1- Self- Managing Brokerage Model -Quality assurance framework (Daniel Robinson et al. [8]):

The quality framework consists of three components which provide procedures for brokering, monitoring and rating services and their providers (see Figure 1).

3.1.1. Brokerage model

When requested the brokerage model makes separate brokers for users of service and their providers (Figure 2).

Consumers and providers of service supply their broker's patterns which define the negotiation models, decision procedures and plans for making and assessing proposals. For service resource management additional information is also provided by the providers of service. Broker contains an engine for negotiation of messages and proposals which is provided by the patterns to an engine builder.

Many types of negotiation models and decision procedures are present. At one time brokers use common negotiation model for exchange of information.

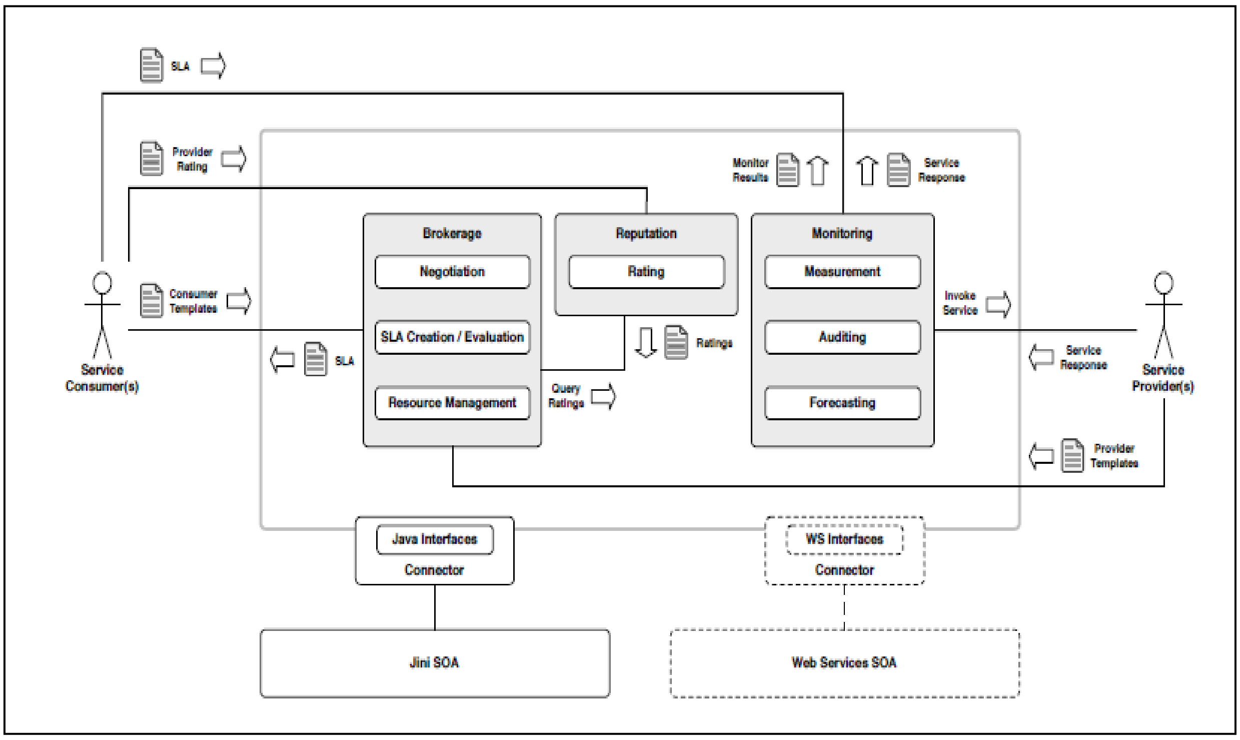

Fig.1 Illustrates the Framework of "A Self-Managing Brokerage Model for Quality Assurance in Service-Oriented Systems" (Daniel Robinson et al. [8]).

3.1.2. Monitoring

In this framework the monitoring system is passive. Therefore on consumer and provider of service there is no extra load. But the provider cannot understand the difference between the consumer and monitor request.

3.1.3 Renegotiation

When a provider broker is not able to deliver an expected quality the consumer assumes there is a need for renegotiation. Rather than expected, any improvement in other quality should be acceptable by the consumer.

3.1.4 Forecasting

Inspection performed by service monitors is complemented by Forecasting. Forecasting model is specified by the consumer of the service. Leanings in the quality measurement are inspected during forecast through which it can be estimated when some specific quality of this service will fail.

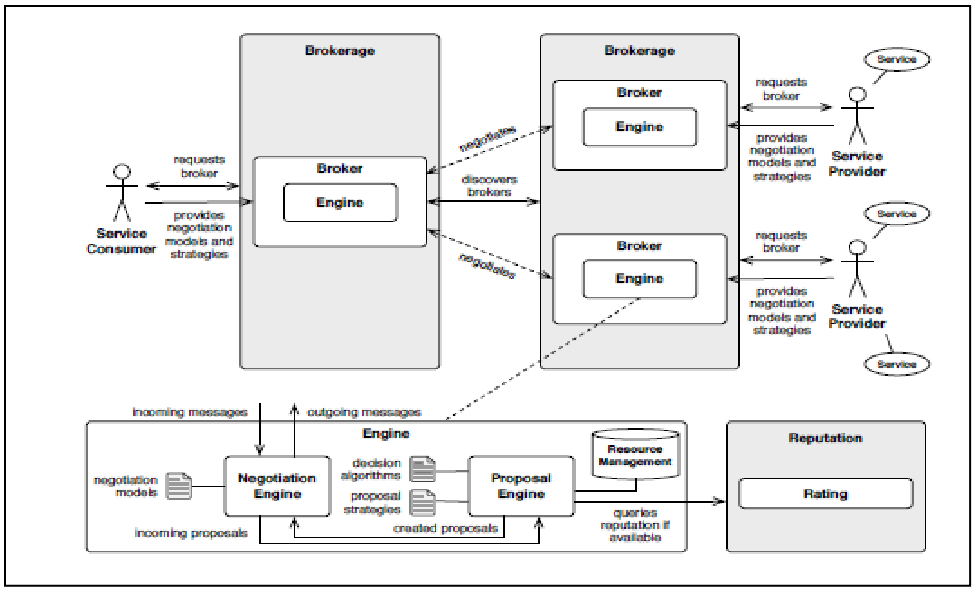

Fig.2 Illustrates the Brokerage model of "A Self-Managing Brokerage Model for Quality Assurance in Service-Oriented Systems" (Daniel Robinson et al. [8]).

3.1.5 Reputation

For a given service performance of service is rated by the reputation service through query method. The providers having good reputation do not need many negotiation sessions due to this method.

3.1.6 Defining service strategy and acceptability

In order to automate the advertisement, discovery, negotiation, and monitoring of services within a service-oriented architecture (SOA), participants in the architecture must share a common set of terms for describing service qualities and constraints.

The strategy schema then enables the expression of strategies, which describe ideal services and service compositions. Strategies also specify acceptable limits on the values of service- and operation-level quality constraints.

3.1.6.1 Quality schema

The Quality XML schema facilitates the description of arbitrary qualities, quality constraints, measurements, and values. The Quality element is used to describe a quality.

3.1.6.2 Service schema

The Service XML schema facilitates the description of services and service contracts in terms of functional attributes and non-functional qualities. The Service element provides a method of describing a service instance.

3.1.6.3 Strategy schema

The Strategy XML schema facilitates the expression of an ideal service or service composition, and the expression of limits on the acceptability of services and their nonfunctional qualities. It also provides the means to express relationships between different qualities.

3.1.6.4. Service composition Framework

Service consumer has to compose the services. The framework only negotiates a composition. Within the compositions individual service is weighted according to the criticality of that service.

3.2. A Framework for Assurance in Service-Oriented Environments (*Soumya Simanta et al. [9]*)

The framework in Figure 3 shows the five dimensions of assurance that need to be considered in order to successfully assure service-oriented systems. At a high level, **strategies** are high-level and abstract approaches for providing assurance, **elements** represent components of service-oriented environment that require assurance, **methods** are specific assurance tools and techniques that are instantiations of assurance strategies, **aspects** of a service-oriented system that require assurance, and **roles** represent the roles and responsibilities of participants in a service-oriented system. The goal of the framework is to present a holistic view of assurance in a service-oriented environment.

3.2.1. Assurance Strategies

Assurance strategies are high-level approaches that check and assure a service in a service oriented environment.

A. Testing

It is not exhaustive but often provides an acceptable level of confidence. Currently, testing is performed both manually and with automated tools. In most cases, testing is performed on the actual implementation (i.e., executable code).

B. Monitoring

Given the dynamic nature of service-oriented environments, monitoring is critical for providing runtime assurance. Monitoring complements other assurance strategies because it is applicable at runtime on the actual deployed version of executing code. Because monitoring is a continuous activity, it is usually automated.

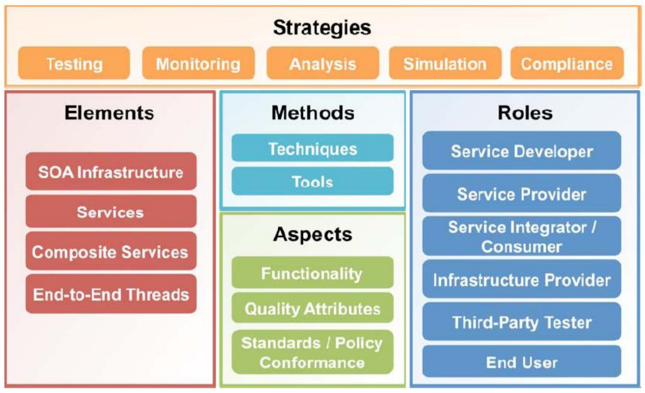

Fig.3 Illustrates the Framework for Assurance in Service-Oriented Environments [9]

C. Analysis

This assurance strategy works mostly on models that represent some aspect of the actual implementation. The one exception is static analysis which works on the actual code.
Most analysis techniques are automated.

D. Simulation

Simulation as an analysis strategy can be useful when all system elements are not available and some form of assurance is required. While testing actual implementations it is being used and is complex and it is expensive to use other forms of assurance.Good simulation often requires substantial modeling and results are only as good as the models being simulated.

E. Compliance

Compliance can be treated as a weaker form of third-party certification. It is difficult to achieve because of the distributed, loosely coupled, and dynamic nature of service-oriented systems. Also, it is not easy to apply compliance to all properties of a service oriented system. Compliance to a set of standards is the most common form in which it is currently used.

3.2.2 Service-Oriented Architecture (SOA)

The elements of the framework that require assurance are the SOA infrastructure, services, composite services and end-to-end threads.

A. SOA Infrastructure

The SOA infrastructure represents software middleware components that provide business-independent capabilities such as service discovery, service routing, identity management, reliable message delivery, metadata management, etc.

B. Services

Services are the core of a service-oriented system because they provide reusable, business-level capabilities.
Assurance of a web service requires assuring all service components.

C. Composite Services

One of the biggest advantages of using service orientation is that multiple services can be composed to create a new service. Assurance techniques should account composite services.

D. End-to-End Threads and Business Processes

Services are often used in business processes or end-to end threads—composites of humans, software applications, services, back-end applications, and databases—that utilize the SOA and network infrastructure to perform a business or mission task. It is not enough to provide assurance of each of the individual components of a service-oriented system unless we can say something about the properties of the end to-end thread that ultimately supports the business process and mission.

3.2.3. Assurance Roles And Responsibilities

This section discusses the roles that are associated with performing assurance in a service oriented environment.

A. Service Developer

Service developers are responsible for developing an individual service from scratch.

B. Service Provider

Service providers host services, and they are also responsible for enabling access to these services.

C. Service Integrator / Consumer

Service integrators use existing services to create composite services or to create an end-user application.

D. Infrastructure Provider

Infrastructure providers are responsible for providing the necessary infrastructure middleware and infrastructure capabilities. They develop guidelines and governance processes for testing and verification of new and revised infrastructure capabilities.

E. Third-Party Service Tester

The role of third-party service testers is similar to an independent QA team in a software project. They validate and potentially certify whether a service (individual or composite) works as expected in a given execution context.

F. End User

End users are the primary users of applications that invoke services. In many cases, end users can become part of the assurance process by participating in beta-testing and by reporting errors and faults.

3.2.4 System Aspects That Require Assurance

A. Functionality

This aspect refers to the functional or business capabilities that must be supported by the service-oriented elements—such as registering a service for an SOA infrastructure, getting a credit history for an individual service.

B. Quality Attributes

Assurance of non-functional or quality attributes is essential for determining the fitness of SOA elements for operational use. Non-functional features include performance, security, reliability, interoperability.

C. Standards and Policy Conformance

Any successful SOA implementation and its elements must conform to a set of standards and policies. Ensuring that the implementation elements show this conformance is an essential part of SOA assurance.

3.2.5 Assurance Methods

Assurance methods represent the *how* of bringing together all the four dimensions—strategies, aspects, roles, and elements. Methods are concrete tools and techniques are used to achieve assured service-oriented systems.

A. Assurance Tools

The current state of assurance tool support is limited to commercial tools that are mostly testing-centric and focused on web services. Assurance tool capabilities include support for testing multiple protocols (e.g., HTTP, HTTPS, WSDL, SOAP, and REST)

B. Assurance Techniques

1) Mocks and Stubs

Mocks and stubs are used to test a composition (orchestration/choreography) in the case that all participating services are not ready. Invocation sequence and input output values are verified by them.

2) Runtime Monitoring

Service monitoring extends verification to runtime. Monitoring capabilities are usually part of the service deployment and execution environment. Service consumers and service providers should work with infrastructure providers to implement the necessary notification mechanisms to support runtime monitoring.

3) Assuring Quality Attributes

Quality attribute assurance should be based on a specific quality attribute model and take context into account.

4) Model Checking

Model checking provides an exhaustive proof-style certificate that the model, if not the actual software, satisfies certain properties. Model checking in service oriented systems is focused on service compositions where atomic services participating in the composition are treated as black-boxes.

5. Application Areas

A quality assurance model has several purposes. Noticeable among them are (i) When presenting and reasoning a quality attribute it highlights the information to be considered, (ii)For assessment of a software, quality measurement procedures are indicated, (iii) quality needs of clients and software engineers are organized and saved.

Following are some areas where service oriented systems are being adopted[10].

1. Information systems formed mainly as a coalition of information systems of organizational autonomous sub units. Examples are:

- Information systems of *e*-government (sub units – offices – having their information systems);
- Health care information systems (subunits are almost independent health care units like hospitals, physicians, laboratories, etc.);
- Information system of a decentralized enterprise (subunits are autonomous divisions);
- The kernel part of Environmental Information System (IS of institutions and enterprises responsible for environmental politics and environmental activities.

2. Software coalitions developed via decomposition of existing systems. Typical cases:

- Reengineering of "worn out" systems tending to be too instable after a long lasting maintenance.
- The functionality must be enhanced and new requirements imply the change of the system architecture.

3. *Software developed as a coalition from scratch* (but some real-time systems or large systems being so complex it is not feasible to develop them as system having a SOA rather monolithic architecture).

So where ever there is a service oriented systems quality assurance is required there. Organizations implementing service-oriented systems must develop a comprehensive SOA assurance strategy that accounts for all the appropriate roles and perspectives and for the relevant elements of the service-oriented environment: infrastructure, individual and composite services, and end-to end threads. Strategies for assuring functional, quality of service, and conformance testing should be identified and integrated into the organizational development and governance processes. Given the continuously and rapidly evolving nature of service-oriented standards and technologies, it essential for organizations to track and adapt accordingly.

Let's take an example: Simulated service oriented consumer devices execute an application for navigation which is location-based. It includes information of location, amount of traffic, weather and map. Consumers of this application need a navigation device which is stimulated with need based requirements. (E.g. Mobile phone or internet tablet). Different providers are needed for these different conditions. So their services may have differences in terms of QOS and Cost. So to provide a better quality assured system there is a need to follow some proper framework of quality assurance which governs not only the individual service but also their composition problems.

6. Current Status

As service-oriented systems become widely accepted, more focus has been given to assurance issues. Characteristics of service-oriented systems (distributed, loosely coupled, etc.) pose serious challenges from an assurance perspective. The more distributed and loosely coupled a service-oriented system is, the harder it is to provide assurance, because of decreasing control. We have found out at start of paper that quality has been variously defined. But due to dynamic and adaptable nature of service oriented systems consumers are mostly concerned with cost and providers reputation.

Present quality assurance schemes are mostly concerned with forecasting systems features only on static properties of its modules. System composed from many services can't be satisfied by the static properties it needs some dynamic runtime method. The method should be able to find and solve the runtime problems not only for the service system but also for its interdependencies.

Secondly, present quality frameworks offer the user only limited control over the quality of a service and therefore the service oriented system. As there is composition of different services so third-party software services are used. So out-side services means that a consumer has tiny authority over the quality of services outside the static service level agreement (SLA).

Current systems are not best for the runtime quality. There are frameworks for monitoring and controlling but active negotiation is missing most of the time. The systems present are restricted to only a limited domain of some famous quality needs. There is a need for assurance of variable qualities.

Sometimes the environment for a service oriented system is limited so quality assurance needs inspection according to the given environment. Methods for this purpose are poor to check for flexibility of the ratio of environment provided to the required service.

We have emphasized earlier that significant quality models are essential to develop high-quality systems. Our basic intention for this work is due to our observation that the available quality models alone cannot be used to meet all the considerations when discussing about quality.
They are good in a limited area but for a large domain there is still a need of a better quality assurance technique.

Current needs require not only a model which allows priority preferences but also dynamic conditional preferences. Thought some work is also done for conditional preferences but that is also limited. Run time quality check is most difficult of all assurance strategies. There is a need of efficient emergency check mechanisms at consumer side. We cannot expect to engineer high quality systems if we cannot adapt quality-related requirements that come certainly to engineers and future users.

8. Conclusion

Assurance should be a central interest from the start, when developing a service-oriented system. Beginning assurance late in the cycle can be costly. Mostly a blend of corresponding assurance strategies will be required to attain satisfactory assurance in service-oriented environments. No single assurance framework is enough up till now. Present assurance practices are effective underneath service level. The service-oriented assurance field is growing, with many open problems that are still in research. Supplementary quality assurance frameworks are required to let the users identify their expectations for quality. We have noticed many similarities between the two quality models but there are some deficiencies in both of them. Both complement some of the advantages and disadvantages of each other. So we have decided to integrate them for assurance in service-oriented systems, moreover we will try to implement active monitoring system to find out and replace any fall back of service. We will advance current research by incorporating few models for dependency and priority information and integrate with these two.

Our intention for the derived model is to serve as a recommended base for further research in quality assurance for service-oriented systems.

References

[1] B. W. Boehm, J. R. Brown, H. Kaspar, M. Lipow, G. J. Macleod, and M. J. Merrit. "Characteristics of Software Quality". North-Holland, Amsterdam, 1978.

[2] T. Erl. Service-Oriented Architecture: "Concepts, Technology, and Design". Prentice Hall PTR, Upper Saddle River, NJ, USA, 2005.

[3] M. Turner, D. Budgen, and P. Brereton. "Turning software into a service". Computer, 36(10):38–44, 2003.

[4] I. Sommerville. Software Engineering (8th Edition), chapter 31. Addison Wesley, 2006.

[5] A. Barbir, C. Hobbs, E. Bertino, F. Hirsch, and L. Martino, "Challenges of Testing Web Services and Security in SOA Implementations " in Test and Analysis of Web Services: Springer Berlin Heidelberg, 2007, pp. 395-440.

[6] G. Canfora and M. Di Penta, "Testing services and servicecentric systems: challenges and opportunities," IT Professional, vol. 8, pp. 10-17, 2006.

[7] Y. Le Blevec, C. Ghedira, D. Benslimane, X. Delatte, and Z. Jarir, "Exposing Web Services to Business Partners: Security and Quality of Service Issue," in First International Conference on Digital Information Management: IEEE, 2006, pp. 69-74.

[8] Daniel Robinson, G. K. "A Self-Managing Brokerage Model for Quality Assurance in Service-Oriented". (2008).

[9] Soumya Simanta, Edwin Morris, Grace A. Lewis, Dennis B. Smith "A Framework for Assurance in Service-Oriented Environments"**.** (2010).

[10] Jaroslav Král and Michal Žemlička. "Service Orientation in Environmental Information Systems" Informatics for Environmental Protection - Networking Environmental Information.