Baltic J. Modern Computing, Vol. 2 (2014), No. 3, 117-131

A Methodology for Capturing and Managing Non-Functional Requirements for Enterprise Service-Oriented Systems

Sandra SVANIDZAITĖ

Department of Software Engineering
Vilnius University Institute of Mathematics and Informatics
Akademijos 4, LT-01108 Vilnius, Lithuania

sandra.svanidzaite@gmail.com

Abstract. Service orientation is a new software development paradigm. It inherits a number of concepts and principles from earlier paradigms but differs from these paradigms in the manner in which the separation of concerns in the software system is done. In addition, it provides an additional software system abstraction layer – business logic layer. Service oriented architecture (SOA) is an architectural style that implements service-orientation approach. SOA raises new problems in software requirements engineering. As a result, a new requirements engineering sub discipline – service-oriented requirements engineering (SORE) – emerges. SORE faces with such SOA requirements engineering issues and challenges: 1) Service Specification issues, 2) Service Discovery issues, 3) Service Knowledge Management issues, 4) Service Composition issues. This paper contributes to solving SORE service specification issues and challenges as it provides a methodology for capturing and managing non-functional requirements for ESOA systems.

Keywords: service-oriented architecture, enterprise service-oriented architecture, service-oriented requirement engineering.

1. Introduction

Service orientation is a new software development paradigm suggesting that business applications should be implemented in the form of services. It inherits a number of concepts and principles from earlier paradigms, first of all, from object-orientation, component-based software engineering (CBSE) and open distributed processing (ODP). The most important innovation of service orientation is the manner in which the separation of concerns is done. A service-oriented architecture (SOA) is an architectural style that implements service-orientation approach.

According to (Newcomer and Lomow, 2004), SOA is "*a style of design that guides all aspects of creating and using business services throughout their lifecycle (from conception to retirement)*". One of sub styles of SOA is an Enterprise SOA (ESOA), to use the term coined by the SAP Corporation (SAP, 2008). This sub-style provides guidelines how to develop and to use service-oriented applications in Enterprise Systems (a.k.a. Systems of Systems). It is a business-driven style, that is, it must support enterprise's business strategy and objectives. This means that business processes in ESOA must be designed keeping in mind this goal. On the other hand, business

processes should be translated into abstracted and normalized Enterprise Business Services (EBSs) drawing on global data types. Normalization means that EBS should be designed with the intent to avoid functional overlaps and to reduce the redundancy (i.e. similar or duplicate bodies of service logic). Global data types are enterprise-wide defined data types based on international standards (Sambeth, 2006). To simplify enterprise-wide service integration and communication, ESOA provides typically one additional architectural element referred to as enterprise service bus. According to (Bichler and Lin, 2006), ESOA allows enterprise to use "plug-and-play interoperability to compose business processes and integrate different information systems on the fly to enable ad hoc cooperation between new partners." It creates business services networks, also known as service supply chains that "raise many new questions about how to foster collaboration and orchestrate processes among partners". So ESOA in many aspects differs from SOA. Although some service providers in ESOA can reside in outside of enterprise and, vice versa, some service consumers also can reside in outside of enterprise, they must obey enterprise's standards and all are designed keeping in mind these standards. In this sense ESOA system is operating rather in a less open environment than ordinary SOA.

As it is described in (Svanidzaite, 2014a), service-oriented paradigm inherits a number of issues and challenges from traditional and component based software development, as well as, adds new ones. As a result, a new requirements engineering sub-discipline – service-oriented requirements engineering (SORE) – emerges.

Bano and Irkam (Bano and Irkram, 2010) suggest that issues and challenges of SORE can be grouped into four categories: *1) Service Specification issues* deal with requirements elicitation and documentation for service-oriented system. *2) Service Discovery issues* deal with the searching for services after their specifications are prepared and finding out which of the services actually meet the functional and non-functional requirements *3) Service Knowledge Management issues* deal with the knowledge management of service compositions - functionality that would aid for service specification and discovery. *4) Service Composition issues* deal with the investigation challenges deciding whether the integrated service-oriented system meets the original requirements defined for each service separately.

Service Specification issues are the ones of great importance for ESOA. As a result, our research concerns in the resolution of the following issue: *Capturing and managing non-functional requirements, finding conflicting requirements and proposing an approach how to resolve conflicts.*

This paper contributes to defining a methodology for capturing and managing non-functional requirements for Enterprise Service-Oriented systems, as it provides: 1) a definition of SOA/ESOA and highlights their differences, 2) describes EA frameworks, EA standards that can be used as an input for ESOA viewpoints and quality attributes derivation, 3) analyses possible stakeholders and quality attributes (NFRs – non-functional requirements), 4) proposes viewpoints for ESOA systems development, 5) defines a methodology for capturing and managing ESOA NFRs.

The remainder of the paper is organized as follows. Section 2 reviews related work. Section 3 defines how non-functional requirements for ESOA systems can be captured using viewpoints, provides a short overview of EA frameworks and standards and discusses how they can be used to define ESOA viewpoints and lastly defines possible ESOA stakeholders and non-functional requirements. Section 4 defines ESOA viewpoints. Section 5 proposes a methodology for capturing and managing ESOA NFRs. Section 6 concludes the paper.

2. Related Work

SORE like traditional requirement engineering, concerns with specification and analysis of system requirements and constraints but its focus is on identification of services and workflows used to modelling applications and on their reuse. Service-oriented software engineering (SOSE) is relatively new and still rapidly growing research and development area. This discipline emerged in the last decade of previous century (Arsanjani, 1999), (Layzell et al., 2000), as a response to the challenges of integration of heterogeneous applications, including legacy ones, cross-platform interoperability and bridging the gap between business models and software architectures. In its initial stages SORE was concerned mostly with the service-oriented software process considering it as an extension and improvement of the Rational Unified Process (Rational Software, 1998) or IBM's Global Service's Method (Arsanjani, 2001). SORE as an integral part of SOSE emerged in the first quinquennium of the 21st century. First publications on this topic discussed the nature of this discipline, its differences between SORE and traditional RE, the structure of service-oriented requirements lifecycle, and possible approaches to address the identification and handling of functional and non-functional requirements for service-oriented systems (Van Eck and Wieringa, 2003), (Trienekens, et at., 2004).

In addition to this, several Service-Oriented System Development methodologies and approaches such as (Svanidzaite, 2014a): IBM RUP/SOMA, SOAF, SOUP, methodology by Tomas Erl and methodology by Michael Papazoglou have been proposed to ensure successful Service-Oriented systems development by providing process guidance and proven best practices from already accomplished SOA projects. SOA development lifecycle in these methodologies is divided into nine phases: Service-oriented planning/inception, Service-oriented analysis, Service-oriented design, Service Construction, Service Testing, Service Provisioning, Service Deployment, Service Execution and Service Monitoring. Although, these methodologies help to structure Service-Oriented systems development processes, they are not aimed at defining SORE process and do not provide any approach to requirement conflicts resolution. As a result, further research is required.

Furthermore, many large software projects are ill-defined as a result of the high level of complexity. It becomes difficult not only to fully specify system requirements but even to understand all aspects of the system. (Leite and Freeman, 1991), (Sommerville and Sawyer, 1997), (Russo et al., 1999), (Nuseibeh and Easterbrook, 2000) suggest that viewpoints can be used to improve system requirements gathering, managing and conflict resolution process. System requirements should be elicited and defined from different viewpoints. For any given viewpoint of the system many aspects will be hidden and only ones actual to the viewpoint will be depicted in details. As a result, multiple viewpoints need to be considered in order to fully understand and specify the system-of-interest.

i* (pronounced "i star") or i* framework (Yu, 2009) is a modelling language suitable for an early phase of system modelling in order to understand the problem domain. i* modelling language allows to model both *as-is* and *to-be* business models. The name i* refers to the notion of distributed intentionality which underlines the framework. This approach is originally developed for modelling and reasoning about organizational environments and their information systems composed of heterogeneous actors with different, often competing, goals that depend on each other to undertake their tasks and achieve these goals. It covers both actor-oriented and goal modelling. i* models answer

the question *who* and *why*, not *what*. i* framework is a part of an User Requirements Notation (URN) international standard. Standard combines two sub-languages (Amyot and Mussbacher, 2011): Goal-oriented Requirement Language (GRL), and Use Case Maps (UCM) notation. URN is the first international standard that addresses business goals and scenarios and links between them in a graphical way. GRL is a visual notation (based on i* notation) used for modelling different stakeholders business goals, NFRs and their alternatives that have to be considered. GRL supports reasoning about goals and NFRs, as it shows the impact of often conflicting goals and various global alternative solutions proposed to achieve goals. The UCM visual scenario notation focuses on the causal flow of behaviour optionally superimposed on a structure of components. UCM supports the definition of scenarios that describe a specific path through the UCM model where only one alternative at any choice point can be taken.

According to (WEB, h), (Svanidzaite, 2014b) SOA projects potentially (the same problems also apply to ESOA projects) suffer from one or more of the following problems:

- SOA projects are significantly more complex than typical software projects, because they require a larger, cross-functional team along with correspondingly more complex inter-team communication and logistics.
- Usually it is hard to define the scope and boundaries of a SOA project. As a result, the vision for the final result is often not clear at the project's inception.
- SOA can have a very positive impact on an organization, but, on the other hand, SOA development and replacement of legacy systems can be very expensive.
- SOA project has a higher risk of failure than other traditional software development projects.

Despite these problems, ESOA/SOA approaches are gaining popularity and are used for more and more complex systems. Having this in mind, ESOA projects require much more sophisticated requirement gathering and management techniques. As SORE emerged recently, there are no works that deal with Service Specification issues employing viewpoints and User Requirements Notation – URN standard languages directly. As a consequence, further research is required.

3. Capturing Non-Functional Requirements for ESOA Systems Using Viewpoints

For technical and human reasons system requirements specifications will always be imperfect. It has been recognized for many years that problems with specifications are probably the principal reason for project failure. Improving the quality of specifications can be achieved in two ways (Sommerville and Sawyer, 1997): *"1) by improving the requirements engineering process so that errors are not introduced into the specification, 2) by improving the organization and presentation of the specification itself so that it is more amenable to validation."*

We propose a methodology for ESOA non-functional requirements capturing and managing which addresses both of these improvement dimensions. It is based on collecting and analyzing the non-functional requirements for ESOA systems from different viewpoints. Viewpoints are entities that are widely used in traditional software systems architecture descriptions and they can be used also to structure ESOA system requirements (functional and non-functional) elicitation and specification.

The organization of system architecture description into views using viewpoints provides a mechanism for separation of concerns among stakeholders, while providing the view of the whole system that is fundamental to the notion of architecture (Sommerville and Sawyer, 1997), (ISO/IEC/IEEE 42010:2011). An architecture description (see Figure 1) includes one or more architecture views. An architecture view addresses one or more of concerns held by the system's stakeholders. An architecture view expresses the architecture of the system-of-interest in accordance with an architecture viewpoint. There are two aspects of a viewpoint: the concerns it frames for stakeholders and the conventions it establishes on views. An architecture viewpoint frames one or more concerns. A concern can be framed by more than one viewpoint. A view is governed by its' viewpoint: the viewpoint establishes the conventions for constructing, interpreting and analyzing the view to address concerns framed by that viewpoint. Viewpoint conventions can include languages, notations, model kinds, design rules, and/or modeling methods, analysis techniques and other operations on views.

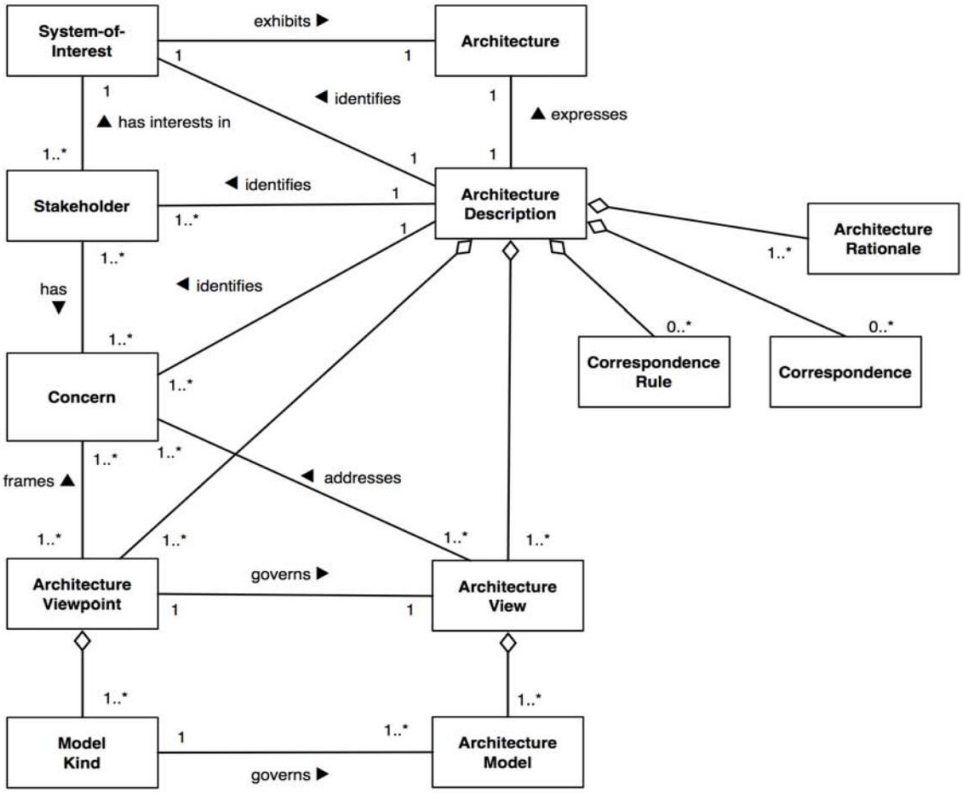

Figure 1. Conceptual model of an architecture description (ISO/IEC/IEEE 42010:2011)

According to the "Systems and software engineering — Architecture description" (ISO/IEC/IEEE 42010:2011) an architecture viewpoint shall specify:

- one or more concerns framed by this viewpoint;
- typical stakeholders for concerns framed by this viewpoint;
- one or more model kinds used in this viewpoint;

- for each model kind, languages, notations, conventions, modelling techniques, analytical methods and/or other operations to be used on models of this kind;
- references to its sources.

Usually stakeholders have different expectations and non-functional requirements (in our methodology non-functional requirements are treated as concerns that are framed by one or more architecture viewpoint) may differ from one viewpoint to another and part of the requirement analysis and negotiation process is to detect and resolve such conflicts.

A viewpoint-based approach to requirements engineering recognizes that all information about the system requirements cannot be discovered by considering the system from a single perspective (Sommerville and Sawyer, 1997). Rather, we need to collect and organize requirements from a number of different viewpoints. A viewpoint is an encapsulation of partial information about system's requirements. Information from different viewpoints must be integrated to form the final system specification.

3.1. Enterprise Architecture Frameworks and Standards

As described in (IEEE Std 1471:2000), architecture is "the fundamental organization of a system, embodied in its components, their relationships to each other and the environment, and the principles governing its design and evolution."

Architecture is important for at least three reasons. It: "1) enables communication among stakeholders, 2) facilitates early design decisions, and 3) creates a transferable abstraction of a system/environment description" (Fernandez-Martinez, Lemus-Olalde, 2004). Enterprise architecture work provides a systematic assessment and description of how the business function operates at the current time; it provides a "blueprint" of how it should operate in the future, and, it provides a roadmap for getting to the target state.

EA layered frameworks and models that employ viewpoints have been proved useful because they provide an advantage for defining contained, non-overlapping partitions of an environment and allow analyzing system architecture from different views. There are a number of frameworks and models that meet this description, for example, The Open Group Architecture Framework (WEB, c), OASIS Reference Architecture Foundation for SOA (WEB, a), Extended Enterprise Architecture Framework (WEB, d), Zachman Enterprise Architecture Framework (WEB, b), DoD Architecture Framework (WEB, e), Kruchten's "4+1" view model/ RUP's 4 + 1 Views (Kruchten, 1995), Siemens' 4 views method (Hofmeister, et al., 2000), Reference Model for Open Distributed Processing (WEB, f). An important recent development in IT architecture practice has been the emergence of standards for architecture description - IEEE 1471-2000 Recommended Practice for Architectural Description (IEEE Std 1471:2000) and its' updated version named - ISO/IEC/IEEE 42010 Systems and software engineering — Architecture description (ISO/IEC/IEEE 42010:2011). These standards aim to promote a more consistent, systematic approach to the creation of architectural views and viewpoints.

3.2. Stakeholders of ESOA Systems

In software systems, stakeholders are the persons or groups of people who are supposed to influence and bring the benefit to the development of the system. In order to address business requirements efficiently in ESOA development, all the key roles must be

identified as stakeholders from the beginning of the project. ESOA stakeholders differ from traditional software systems stakeholders in a number of ways. Firstly, new service-oriented roles, tasks and responsibilities are introduced. Secondly, ESOA projects require more governance as ESOA initiative usually encompasses all enterprise and is not limited to a specific project. For example, (WEB, g) suggests service-oriented governance stakeholder groups such as: ESOA Steering Board, ESOA Governance Board in addition to Business/IT Steering Group and EA Governance Board stakeholder groups which are usually established in traditional software development projects. Moreover, ESOA initiatives require more experience and supervision. As a result, ESOA Center of Excellence stakeholder group is introduced by (WEB, g). Furthermore, a separate Service Development Team is proposed to be able to develop services by one team and integrate (compose) them by another – Solutions Development Team.

Stakeholders of a system have concerns with respect to the system-of-interest considered in relation to its environment. Quality attributes of ESOA system in viewpoints can be reflected as concerns. A concern can be held by one or more stakeholders. Concerns arise throughout the life cycle: from system needs and requirements, from design choices and from implementation and operating considerations. The role of an architect is to address these concerns, by identifying and refining requirements that stakeholders have, developing viewpoints of an architecture that show how concerns and requirements are going to be addressed, and by showing the trade-offs that are going to be made in reconciling the potentially conflicting concerns of different stakeholders (Sommerville and Sawyer, 1997).

The following stakeholder groups should be considered and when applicable, identified in the architecture description (ISO/IEC/IEEE 42010:2011): users of the system, operators of the system, acquirers of the system, owners of the system, suppliers of the system, developers of the system, builders of the system, maintainers of the system.

The list of stakeholder groups for ESOA systems can include some or all of the groups as follows (WEB, g):

Business/IT Steering Group *(Sponsorship of all IT Solutions and Services)*: CIO – Chief Information Officer, CTO or Chief IT Strategist, Chief Architect, Business Domain Owners

ESOA Steering Board *(Sponsorship of ESOA Program and Leadership):* ESOA Chief Architect, ESOA Program Director, ESOA Business Sponsor

EA Governance Board*:* Chief Enterprise Architect, Enterprise Architects, Chief ESOA Architect

ESOA Center of Excellence *(Definition and Development):* Business Champion, Chief ESOA Solution Architect, Organizational Change Consultant, Test Strategist, Tool strategist

Business Domain Representatives *(Scope and Delivery Management):* Program Manager, Business Architect, Process Engineer, Business Subject-matter Expert

ESOA Governance Board *(Informing and Monitoring):* ESOA Chief Architect, Business Architects

Solution Development Team *(Execution and Delivery):* Project Manager Business Analysts, Solution Architects, Integration Specialist, Operations Architect, Developers, Testers, Security Architect

Service Development Team *(Execution and Delivery):* Project Manager Business Analysts, Service Architects, Integration Specialist, Operations Architect, Developers, Testers, Security Architects

IT Operations *(Execution and Delivery):* Database Administrator, Network Infrastructure Architect, System Administrator, Service Operations Manager

ESOA Consumers *(Production):* Users that directly interact with ESOA, external systems, applications, services.

It is up for the enterprise to identify which of the above described ESOA stakeholder groups will exist on its' ESOA initiative. Every identified stakeholder group will have its' own concerns regarding non-functional requirements of ESOA that will be framed by ESOA viewpoint. The composition of ESOA viewpoints is discussed in section 4 of this paper.

3.3. Non-Functional Requirements for ESOA Systems

Non-functional requirements for ESOA systems are inherited from traditional software systems. The difference lies only in the definition of quality attribute and its' metrics. In traditional software systems requirements engineering quality attributes are defined in a more generic way and usually concern about the characteristics of the whole system. For example, availability quality attribute in (ISO/IEC 25010:2011) is defined as a *"degree to which a system, product or component is operational and accessible when required for use".* Some of the metrics for this attribute are: mean time between failure (MTBF) and mean time to recover (MTTR). On the contrary, in ESOA world quality attributes can be defined in a more specific way and are limited to measuring the characteristic of a specific service. The same availability attribute in (O'Brien et al., 2005), (Choi et al., 2007) is defined as a *"quality attribute that measures the degree to which a service is accessible and operational when service consumer requests for use"*. The metrics for this attribute in ESOA suggested by (Choi et al., 2007) are: availability of business process (ABP) and availability of web service (AWS). Generally, in ESOA quality of service is hidden from service consumers due to the black-box nature of ESOA. In a service composition, low quality of an atomic service may cause the quality degradation of all its successors in a service composition.

The choice to use an ESOA approach depends on several factors including the architecture's ultimate ability to meet functional and non-functional requirements. Usually, architecture needs to satisfy many non-functional requirements in order to achieve the organization's business goals. (O'Brien et al., 2005), (Choi et al., 2007) suggest defining ESOA non-functional requirements by identifying unique features (principles) of ESOA such as: *loose coupling, well-defined service contract, standard based, abstraction, reusability, discoverability, composability, adaptability, service interface level abstraction* and mapping those features to quality attributes. In almost all cases, tradeoffs have to be made between these requirements. As a consequence, each of the ESOA stakeholder group (defined in section above) will be concerned in one or more quality attributes provided below (O'Brien et al., 2005), (Choi et al., 2007):

Availability - this quality attribute measures the degree to which a service is accessible and operational when service consumer requests for use.

Performance - this quality attribute measures the capability of the service to provide appropriate response and processing times and throughput rates when performing its function, under stated conditions

Reliability - this quality attribute measures the ability of a service to keep operating with specified level of performance over time

Usability - this quality attribute measures the capability of a service to be effectively understood, learned and used by the service consumer

Discoverability- this quality attribute measures the capability of the service to be easily, accurately, and suitably found at both design time and runtime for the required service specification

Adaptability - this quality attribute measures the capability of the service to be feasibly adapted at both design time and runtime for different consumer's preference and service context information

Composability - this quality attribute measures the capability of a service to be well composed to other services or a service composition to operate successfully by composing atomic services

Interoperability – this quality attribute refers to the ability of a collection of communicating entities to share specific information and operate on it according to an agreed-upon operational semantics

Security - this quality attribute denotes different things with respect to software systems, in general, it is associated with four principles: confidentiality, authenticity, integrity, availability

Scalability – this quality attribute refers to the ability of an SOA to function well (without degradation of other quality attributes) when the system is changed in size or in volume in order to meet users' needs

Extensibility is the ease with which the services' capabilities can be extended without affecting other services or parts of the system

Testability is the degree to which a system or service facilitates the establishment of test criteria and the performance of tests to determine whether those criteria have been met

Auditability is the quality factor representing the degree to which an application or component keeps sufficiently adequate records to support one or more specified financial or legal audits

Modifiability is the ability to make changes to a system quickly and cost-effectively.

4. Composition of ESOA Viewpoints

Viewpoints that we developed for ESOA systems modelling are based on: Service-oriented Architecture layers proposed by (WEB, a), on EA standards (ISO/IEC/IEEE 42010:2011), (IEEE Std 1471:2000) and organizational and domain knowledge that according (Sommerville and Sawyer, 1997) is "*knowledge which constrains the system requirements. The constraints may be physical (e.g., network performance), organizational (e.g., incompatible hardware used in different divisions of a company), human (e.g., average operator error rate) or may reflect local, national or international laws, regulations and standards. This type of viewpoint cannot be associated with a single class of stakeholder but includes information collected from many different sources (people, documents, other systems, etc.)*". The viewpoints that we designed include one business process viewpoint - *Organization Business Processes Viewpoint* and nine ESOA system *architectural viewpoints*. Such a composition of viewpoints provides a holistic view about the system-of-interest starting from high level business process description and requirements and transforming these business process requirements into enterprise service-oriented system requirements by providing more detailed system non-functional requirements on each of the nine ESOA system architectural viewpoints.

The following list of ESOA viewpoints is suggested:

Organization Business Processes Viewpoint – viewpoint that displays all enterprise business processes (business model) and their interconnections without alignning them to software systems. Stakeholder groups that are interested in this viewpoint are the following: Business/IT Steering Group. Viewpoint has none of ESOA non-functional requirements in its concerns.

Consumer Viewpoint is the viewpoint where consumers interact with the ESOA. It enables an ESOA to support a client-independent, channel-agnostic set of functionality, which is separately consumed and rendered through one or more channels (client platforms and devices). Thus, it is the point of entry for consumers (humans and other applications/systems) and services from external sources (e.g., Business-to-Business (B2B) scenarios) to interact with system. Stakeholder groups that are interested in this viewpoint are the following: ESOA Consumers, Business Domain Representatives. Viewpoint frames the following quality attributes: availability, performance, usability, reliability, security, scalability, auditability.

Business Process Viewpoint supports and manages business processes and enables the ESOA to choreograph or orchestrate services to realize business processes. Stakeholder groups that are interested in this viewpoint are the following: Business Domain Representatives, ESOA Center of Excellence, ESOA Governance Board, Solution Development Team. Viewpoint frames the following quality attributes: discoverability, adaptability, composability, interoperability.

Service Viewpoint consists of all the services defined within the ESOA. This viewpoint can be thought of as containing the service descriptions for business capabilities and services with their IT manifestation during design time, as well as service contract and descriptions that will be used at runtime. Stakeholder groups that are interested in this viewpoint are the following: Business Domain Representatives, ESOA Center of Excellence, ESOA Governance Board, Service Development Team. Viewpoint frames the following quality attributes: discoverability, adaptability, composability, availability, performance, usability, reliability, extensibility, testability, modifiability.

Service Components Viewpoint - contains software components, each of which provides the implementation or "realization" for services and their operations, hence the name "Service Component". Viewpoint also contains the Functional and Technical Components that facilitate a Service Component to realize one or more services. Stakeholder groups that are interested in this viewpoint are the following: ESOA Center of Excellence, ESOA Governance Board, Service Development Team. Viewpoint frames the following quality attributes: adaptability, composability, availability, performance, reliability, extensibility, modifiability.

Operational Service Viewpoint - All runtime elements of architecture reside in this viewpoint. Effectively, this viewpoint can conceptually be thought of as the runtime or deployment time of the solution. This viewpoint describes the runtime and deployment infrastructure; the programs, platforms, application servers, containers, runtime environments, packaged applications, virtual machines, etc. that are on the hardware and are needed to support the ESOA solution. Stakeholder groups that are interested in this viewpoint are the following: ESOA Center of Excellence, ESOA Governance Board, IT Operations. Viewpoint frames the following quality attributes: availability, performance, reliability, security, scalability, auditability.

Integration Viewpoint - is a key enabler for an ESOA as it provides the capability to mediate which includes transformation, routing, and protocol conversion to transport service requests from the service requester to the correct service provider. Thus, it supports the capabilities required for enabling ESOA such as routing, protocol support

and conversion, messaging/interaction style, support for heterogeneous environment, adapters, service interaction, service enablement, service virtualization, service messaging, message processing, and transformation. Stakeholder groups that are interested in this viewpoint are the following: ESOA Center of Excellence, ESOA Governance Board, IT Operations, Solution Development Team and Service Development Team. Viewpoint frames the following quality attributes: availability, performance, reliability, discoverability, adaptability, composability, interoperability, scalability.

Quality of Service/Solution Viewpoint provides solution QoS management of various aspects, such as availability, reliability, security as well as mechanisms to support, track, monitor, and manage solution QoS control. Viewpoint provides the service and ESOA solution lifecycle processes with the capabilities required to ensure that the defined policies, Non-Functional Requirements (NFRs), and governance regimens are adhered to. Stakeholder groups that are interested in this viewpoint are the following: ESOA Center of Excellence, ESOA Governance Board, EA Governance Board.

Information Architecture Viewpoint is responsible for manifesting a unified representation of the information aspect of an enterprise as provided by its IT services, applications, and systems enabling business needs and processes and aligned with the business vocabulary – glossary and terms. This viewpoint includes information architecture, business analytics and intelligence, metadata considerations, and ensures the inclusion of key considerations pertaining to information architectures that can also be used as the basis for the creation of business analytics and business intelligence through data marts and data warehouses. Stakeholder groups that are interested in this viewpoint are the following: ESOA Center of Excellence, ESOA Governance Board, EA Governance Board. Viewpoint has none of ESOA non-functional requirements in its concerns.

ESOA Governance Viewpoint ensures that the services and ESOA solutions within an organization are adhering to the defined policies, guidelines, and standards that are defined as a function of the objectives, strategies, and regulations applied in the organization and that the SOA solutions are providing the desired business value. Stakeholder groups that are interested in this viewpoint are the following: ESOA Steering Board, ESOA Center of Excellence, ESOA Governance Board, EA Governance Board. Viewpoint has none of ESOA non-functional requirements in its concerns.

5. Methodology for ESOA NFRs Capturing and Management

We propose a methodology for ESOA NFRs capturing and management that is based on the aim of service-orientation - to develop systems that support enterprise business strategy, objectives and goals and, as a result, is primarily concerned with exposing "*why*" (by modeling business goals) certain NFRs are more important than the others. In this paper, we introduce a requirement negotiation spiral model which is based on a requirements negotiation model described in (Ahmad, 2008). This model (see Figure 2) is designed to benefit from the iterative requirement negotiation process and allows renegotiation. Requirement negotiation process is based on a spiral model to accommodate the dynamic requirements engineering. Each round of the cycle resolves more conflicted requirements and achieves better resolution.

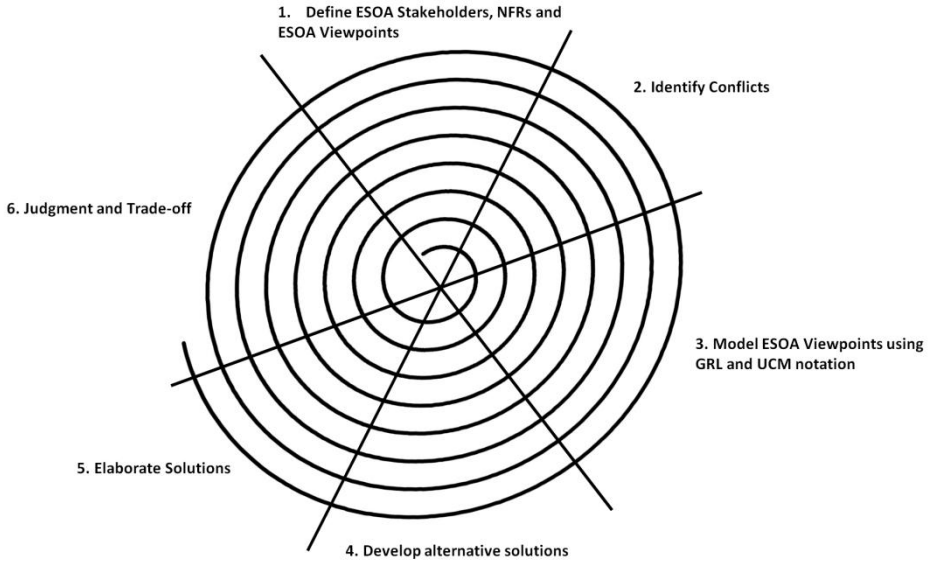

Figure 2. ESOA NFRs Negotiation Spiral Model

Methodology starts with "Define ESOA Stakeholders, NFRs and ESOA Viewpoints" activity. The input of this activity is the list of stakeholders, their concerns regarding ESOA system quality attributes and a list of possible ESOA viewpoints. This activity results in defining ESOA viewpoints that clearly state ESOA system stakeholders and their concerns for NFRs. Next step is to identify conflicts. If a viewpoint has more than one stakeholder group, we search for conflicting NFRs in it by employing a simple tabular method similar to the Quality Function Deployment (QFD) method (Sommerville and Sawyer, 1997), (Errikson and McFadden, 1993) where two stakeholder groups NFRs are checked for mutual consistency. NFRs of one stakeholder group is displayed as rows, NFRs of another stakeholder group are displayed as columns. Where they intersect, we examine them to assess whether they are overlapping, conflicting or independent. If some overlapping or conflicting NFRs are found they are further analyzed and discussed using GRL and UCM diagrams so that requirement overlaps and conflicts would be resolved.

	Stakeholder Group 1			
		REQ1	REQ2	REQ3
Stakeholder Group 2	REQ1	0	10	1
	REQ2	1	0	0
	REQ3	0	1	10

Table 1. Tabular method to check NFRs for mutual consistency (independent requirements are marked with "0", overlapping – "10", conflicting – "1")

Secondly, after NFRs are checked in the limits of one viewpoint, we start looking for conflicting NFRs among different viewpoints. Each pair of viewpoints with intersecting focus is checked for mutual consistency. The same tabular method is used as a checklist

of requirements compliance where two viewpoints named VP1 and VP2 are displayed. VP1's NFRs are represented as rows and VP2's NFRs are represented as columns. Where they intersect, we examine them to assess whether they are overlapping, conflicting or independent. If some overlapping or conflicting NFRs are found they are further analyzed employing GRL and UCM diagrams. After the impact of conflicting NRFs to business goals is elicited, stakeholders need to develop alternative solutions. The solution alternatives are then further elaborated to promote a better understanding among stakeholders. Lastly, judgment and trade off takes place based on the judgment criteria (for example: schedule, cost, functionality and technology capability) and resolution strategy. As an example, if stakeholders choose a collaborative strategy that means that they are focused on satisfying the concerns of all stakeholders. As a result, they may come out with a solution that satisfies a minimum number of concerns of all the stakeholders. The agreed requirements are then evaluated and analyzed. If requirements re-negotiation is required, it has to go into another spiral.

6. Conclusions and Future Research

The aim of this paper was to propose a methodology for capturing and managing non-functional requirements for enterprise service-oriented systems that would help to solve service specification issues and challenges which are encountered in Service-Oriented Requirement Engineering – SORE. Methodology employs viewpoint concept that is widely used in Enterprise Architecture standards and frameworks. It defines possible stakeholder groups, non-functional requirements, describes a method how to find conflicting requirements and proposes a requirements negotiation process for conflict resolution. Requirement negotiation process suggests using User Requirements Notation (URN) standard languages: Goal-oriented Requirement Language (GRL) and Use Case Maps (UCM) notation to model viewpoints that contain conflicting and overlapping non-functional requirements. These languages are designed to model system requirements by showing how they affect high level business goals and business strategy.

Our proposed methodology is the first one that defines viewpoints for enterprise service-oriented systems. The one, that proposes to solve non-functional requirements conflicts by modelling viewpoints using GRL and UCM languages. This methodology can also be used for modelling traditional software systems non-functional requirements. Light adjustments – ESOA Viewpoints (including stakeholders and non-functional requirements) need to be redesigned to remove service-orientation principle. In addition to this, methodology can also be used for modelling functional requirements. This time, non-functional requirements (treated as concerns in viewpoints) have to be swapped with functional requirements. Moreover, Use Cases for the system-of-interest (prepared before the application of this methodology) can be of a great help. Use Cases can be easily transformed to Use Case Maps by transforming each activity on a Use Case to Use Case Map component responsibility and by providing execution scenarios that include more details (conditions, possible values, loops, waiting places, timers, timeout paths) about the way these activities will executed in real life on a system. On the other hand, this methodology can be very hard to apply it practically if no direct mapping between business goals (that are modelled with GRL) and system functions (modelled with UCM) exists, as the main aim of this methodology is to help to choose such system functions with such quality characteristics that help to achieve business goals the best.

Our future plan is to test this methodology on Insurance domain by modelling an Enterprise Insurance System which provides personal insurance services.

Acknowledgement

Thank you to my supervisor, Prof. Dr. Albertas Čaplinskas for his endless support, stimulating suggestions and encouragement throughout the process of writing this paper.

References

Ahmad, S. (2008). Negotiation in the Requirements Elicitation and Analysis Process. 19th Australian Conference on Software Engineering. *IEEE Computer Society, 683-689*

Amyot, D., Mussbacher, G. (2011). User Requirements Notation: The First Ten Years, The Next Ten Years. *Journal of Software Vol. 6, No. 5*

Arsanjani, A. (1999). Service Provider: A Meta-Domain Pattern and its Business Framework Implementation, *In Online Proceedings of the Pattern Languages in Programming Conference*

Arsanjani, A. (2001). Enterprise Component: A compound pattern for building component architectures, *IEEE Computer Society*

Bano, M., Irkram, N. (2010). Issues and challenges of Requirement Engineering in Service Oriented Software Development, *IEEE Computer Society*, 64-69 .

Bichler, M., Lin, K-J. (2006). Service-Oriented Computing. *Computer* (39-3), 99-101

Choi, S. W., Her, J. S., Kim, S. D. (2007). Modeling QoS Attributes and Metrics for Evaluating Services in SOA Considering Consumers' Perspective as the First Class Requirement. *APSCC, IEEE*, 398-405

Errikson, I., McFadden, F. (1993). Quality Function Deployment: A Tool to Improve Software Quality. *Information and Software Technology 35, 9*

Fernandez-Martinez, L. F., Lemus-Olalde, C. (2004). Improving the IEEE std 1471-2000 for Communication among Stakeholders and Early Design Decisions, *Proceeding (418) Software Engineering*.

Hofmeister, C., Nord, R., Soni, D. (2000). *Applied Software Architecture*. Addison-Wesley, Boston

IEEE Std 1471:2000. *Recommended Practice for Architectural Description of Software-intensive Systems*

ISO/IEC/IEEE 42010:2011. *Systems and software engineering - Architecture description.*

ISO/IEC 25010:2011, *Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models*

Yu, E. (2009). *Social Modeling and i**. In Conceptual Modeling: Foundations and Applications, Lecture Notes in Computer Science Volume 5600, 99-121

Kruchten, P. (1995). Architectural Blueprints — The "4+1" View Model of Software Architecture. *IEEE Software* 12 (6), 42-50

Layzell, P. et al. (2000). Service-based Software: The Future for Flexible Software, *In Proceedings of Asia-Pacific Software Engineering Conference*, 5-8 December, Singapore. IEEE Computer Society

Leite, P., Freeman, P. (1991). Requirements Validation Through Viewpoint Resolution. *IEEE Trans. Softw. Eng. 17, 12* .

Newcomer, E., Lomow, G. (2004). *Understanding SOA with Web services (Independent Technology Guides)*. Addison-Wesley

Nuseibeh, B., Easterbrook, S. (2000). Requirements engineering: a roadmap. *Proceedings of the Conference on The Future of Software Engineering (ICSE '00).*

O'Brien, Liam., Bass, Len., Merson, P. F. (2005). Quality Attributes and Service-Oriented Architectures. *Software Engineering Institute. Paper 449*

Rational Software. (1998). *Rational Unified Process: Best Practices for Software Development Teams*,

Russo, A., Nusbeibeh, B., Kramer, J. (1999). Restructuring Requirements Specifications. *IEEE Proceedings*

Sambeth, M. (2006) *Enterprise SOA. Mastering Future Business*. Presentation slides, SAP AG

SAP (2008). *Enterprise SOA Development Handbook 1.1*, available at http://www.sdn.sap.com/irj/scn/go/portal/prtroot/docs/library/uuid/40db4735-02f9-2a10-b198-a888a056bb67?overridelayout=true

Sommerville, I., Sawyer, P. (1997). Viewpoints: principles, problems and a practical approach to requirements engineering. *Annals of Software Engineering*, Volume 3, 101–130

Svanidzaite, S. (2014a). Towards Service-oriented Requirement Engineering. *Proceedings of IVUS 2014*, XIX Interuniversity Confer. Information Society and University Studies 2014, 15-20

Svanidzaite, S. (2014b). An Approach to SOA Methodology: SOUP Comparison with RUP and XP, Computational Science and Techniques 2,1, 238-252

Trienekens, J., Bouman, J.J., van der Zwan, M. (2004). Specification of Service Level Agreements: Problems, Principles and Practices, *Software Quality Journal, 12(1),* 43-57

Van Eck, P., Wieringa, R. (2003). Requirements Engineering for Service-Oriented Computing: a Position Paper, *Proceedings of First International E-Services Workshop, ICEC 03*, Pittsburgh, USA, 23-28

WEB (a). *The Open Group SOA Reference Architecture (SOA-RAF)*. http://www.opengroup.org/soa/source-book/soa_refarch/index.htm

WEB (b). *Zachman Institute for Framework Advancement (ZIFA)*. http://www.zifa.com/

WEB (c). *TOGAF®, an Open Group standard*, http://www.opengroup.org/subjectareas/enterprise/togaf

WEB (d). *Extended Enterprise Architecture Framework Essentials Guide v1.5. Institute For Enterprise Architecture Developments,* http://www.enterprise-architecture.info/Images/E2AF/Extended%20Enterprise%20Architecture%20Framework%20Essentials%20Guide%20v1.5.pdf

WEB (e). *DoDAF Architecture Framework Version 2.02*, http://dodcio.defense.gov/dodaf20.aspx

WEB (f). *Reference Model of Open Distributed Processing (RM-ODP)*, http://www.rm-odp.net/

WEB (g). *SOA Governance Technical Standard: SOA Governance Reference Model (SGRM)*, http://www.opengroup.org/soa/source-book/gov/sgrm.htm

WEB (h). Mittal, K. Service Oriented Unified Process (SOUP), http://www.kunalmittal.com/html/soup.html

Received August 9, 2014, revised August 31, 2014, accepted September 4, 2014.