Using Systematic Review to Elicit Requirements of Reference Architectures

Elisa Yumi Nakagawa and Lucas Bueno Ruas Oliveira

Dept. of Computer Systems
University of São Paulo - USP
PO Box 668, 13560-970, São Carlos, SP, Brazil
{elisa,buenolro}@icmc.usp.br

Abstract. Software architectures have played a significant role in determining the success of software systems. In this context, reference architectures have emerged as a special type of architecture that contains knowledge of a specific domain, aiming at improving reuse and productivity, facilitating the development of systems of that domain. Considering their relevance, the establishment of reference architectures is very interesting; however, it is not a trivial task, mainly due to the difficulty to identify their requirements, since a range of knowledge is required. Thus, the main objetive of this paper is to propose the use of Systematic Review as a technique to elicit requirements of reference architectures. In order to show the viability of our idea, we present a case study on the software testing domain. Results have pointed that Systematic Review could be considered an efficient and additional technique to gather spread domain knowledge, essential task to the establishment of a more complete reference architecture.

1 Introduction

Software architectures have received increasing attention as an important subfield of Software Engineering [25]. Besides, software architectures play a major role in determining system quality, since they form the backbone to any successful software-intensive system. Decisions made at the architectural level directly enable, facilitate, hamper or interfere with achieving business goals as well as meeting functional and quality requirements. In this perspective, in this work, we adopt software architecture as a structure (or a set of structures) of the system which comprises software elements, the externally visible properties of those elements, and the relationships among them [4]. In this context, reference architectures have emerged as an element that aggregates knowledge of a specific domain by means of activities and their relations. They promote reuse of design expertise by achieving solid, well-recognized understanding of a specific domain. Considering their relevance, reference architectures for different domains have been increasingly proposed and used as basis to development of software systems [1].

Considering the impact of reference architectures, initiatives that propose approaches to establish these architectures can be found [5,17,19]. Therefore,

these initiatives do not propose techniques that deal with specifically eliciting architectural requirements, i.e., requirements of a reference architecture that describes common functionalities and configurations presented in systems of a given domain. Since requirement elicitation is an essential activity to obtaining requirements of software systems, as widely discussed in the literature [26], we believe that it could be also explored in the reference architecture context, aiming at obtaining requirements of reference architectures. This activity could result in more complete reference architectures regarding knowledge they contain and that in fact they could collaborate to the software development.

In another perspective, in recent years Evidence-Based Software Engineering (EBSE) has attracted much attention of the software engineering community [8,12], aiming at providing knowledge about when, how, and in what context technologies, processes, methods or tools are more appropriate for software engineering practices. EBSE has proposed Systematic Review as a technique to provide a complete and fair evaluation of evidences related to a topic of interest [12]. In other words, it makes possible to systematically obtain literature review and it is used to summarizing, assessing and interpreting the relevance of all evidences related to a specific question, topic area, or phenomenon of interest. Thus, Systematic Review could be used in order to gather information and knowledge about a specific domain. Considering its advantages, in the last years, events, such as the International Conference on Evaluation and Assessment in Software Engineering and Experimental Software Engineering Latin American Workshop, have widely discussed the application of Systematic Review.

Since reference architectures intend to contain all knowledge of a specific domain, the use of a technique that makes possible to identify this range of knowledge seem to be very interesting. Thus, the main objective of this paper is to present SyRRA (Systematic Review for the Reference Architecture context), a process of Systematic Review to specifically elicit requirements containing information and knowledge of a domain, aiming at establishing reference architectures. In order to show the viability of our idea, we have presented a case study on the software testing domain. Results have pointed out that Systematic Review is an efficient technique to gather mainly spread knowledge that is essential to the establishment of a more complete reference architecture.

This paper is organized as follows. In Section 2, we present the background on reference architecture and Systematic Review, as well as the related work. In Section 3, we present our approach to the requirement elicitation using Systematic Review. In Section 4, we present the case study. Finally, in Section 5, we present our conclusions and future work.

2 Background

Given that reference architectures and systematic review technique are basis of this work, in this section, we present a brief overview about them, as well as the related work.

2.1 Reference Architecture

A reference architecture plays a dual role with regard to specific target software architectures [1]: it generalizes and extracts common functions and configurations; and it provides a base for instantiating target systems. In other words, they can be seen as a knowledge repository of a given domain, contributing to the software development, since reuse of knowledge and improvement in the productivity are promoted. A diversity of reference architectures for different domains can be found [1]. Therefore, these architectures have been sometimes designed without using a more systematized process. Regarding reference architecture design, Muller [17] has proposed recommendations in order to create and maintain reference architectures; basically, references architecture must be understandable, up-to-date and maintainable. In the context of product line development, Bayer [5] presents PuLSE-DSSA (Product Line Software Engineering - Domain-Specific Software Architecture), a systematic approach to define reference architectures capturing knowledge from existing system architectures. Other work have pointed out the need of formalizing processes to design reference architectures [10], since informal processes have still been used. Furthermore, in a previous work, we have proposed ProSA-RA [19]. In order to obtain knowledge about the domain, ProSA-RA suggests to use documents (user manual, for instance), software systems, domain experts and even an ontology as information sources to identify requirements of the reference architecture. In spite of these initiatives, we have observed that there is a lack of techniques that systematically make possible to conduct the requirement elicitation activity in the context of reference architecture.

2.2 Systematic Review

Systematic review is a technique that provides a comprehensive and systematic evaluation of research using a predefined strategy of search aiming at minimizing bias [13]. In other words, it makes possible to systematically obtain literature review and it is used to summarizing, assessing and interpreting the relevance of all evidence related to a specific question, topic area, or phenomenon of interest. In this context, an individual evidence (for instance, a case study or an experimental study divulged in a publication/paper) which contributes to a systematic review is called *primary study*. In order to conduct the systematic review, Kitchenham et al. [12] has proposed a process. In short, this process has three main phases: (i) Planning: the research objectives and the review protocol are defined. The protocol constitutes a pre-determined plan that describes the research questions and how the systematic review will be conducted; (ii) Conduction: during this phase, the primary studies are identified, selected and evaluated according to the inclusion and exclusion criteria established previously. For each selected study, data are extracted and synthesized; and (iii) Reporting: a final report is formatted and presented.

Considering its relevance, systematic review has been applied successfully in different topics of interest. For instance, in the Software Architecture area, systematic reviews can be also found [11,24]. In particular, Farenhorst and Boer [11]

apply systematic review to specifically understand how the term "architectural knowledge" has been approached by community and what is it related to.

In summary, Systematic Review has been initially proposed as a technique to be applied for different topics of interest, in order to identify knowledge about such topics. However, we have proposed to use it as a requirement elicitation technique when developing reference architectures, since it seems to be interesting and viable.

3 An Approach to Elicit Requirements of Reference Architectures

As stated before, reference architectures have emerged as a special type of architecture, providing more precise directions to specify concrete architectures of a set of systems of a given application domain. Therefore, there are several characteristics inherent of this type of architecture that must be considered during the requirement elicitation time. In this perspective, we will first discuss requirements of reference architectures and concrete architectures.

3.1 Requirements of Concrete Architectures and of Reference Architectures

There is a number of differences between concrete architectures (architecture of a given system) and reference architectures [4]. Differently from concrete architectures, in general, reference architectures have a general nature and are designed to meet the attributes of all stakeholders and business context from a specific domain. Furthermore, according to Angelov et al. [2], there are important characteristics that make concrete architectures and reference architectures different: (i) Reference architectures are defined on a high level of abstraction if compared with concrete architectures. This fact is due to the more general nature of reference architecture. Sometimes, these architectures must represent the whole domain for which they were designed; (ii) While the stakeholder group for concrete architectures is more defined; there is not sometimes a clear group of stakeholders of a reference architecture. Moreover, because of business concerns, the composition of a heterogeneous stakeholder group, evolving different companies, is in general infeasible; and (iii) A reference architecture has to address more architectural qualities than a concrete architecture. These additional architectural qualities are due to the their wider audience. There are also non-functional requirements that are important only to reference architectures, such as applicability or architectural feasibility.

As a consequence of these differences, the requirements of concrete architectures are certainly different from requirements of reference architectures. In particular, it is observed that requirements of reference architectures must be obtained considering more diverse information sources. Furthermore, there is an inherent difficulty in order to obtain requirements that adequately represent the entire domain. Therefore, the ways and techniques to obtain requirements of

a reference architecture are also different if compared with those used to elicit requirements of a concrete architecture, i.e., requirements related to a given software system. In this perspective, Systematic Reviews could be applied as a means of evaluating and interpreting all available relevant research to a particular application domain, using a trustworthy, rigorous, and auditable methodology.

3.2 Systematic Review Adapted to the Reference Architecture Context

In order to apply Systematic Review to the reference architecture context, we have adapted the process of Systematic Review initially proposed by Kitchenham et al. [12]. Thus, we have established SyRRA, a process that supports the conduction of systematic review to specifically elicit requirements of reference architectures. As presented in Figure 1, SyRRA presents three main phases: planning, conduction and requirement establishment. Following, these phases are discussed in more details:

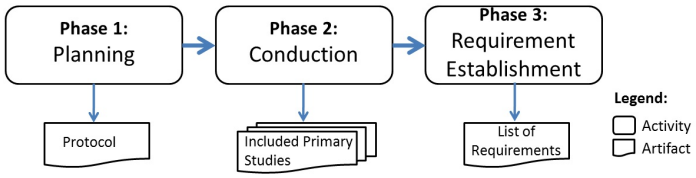

Fig. 1. SyRRA: Systematic review process to the reference architecture context

– **Phase 1 - Planning:** In this phase, the review protocol is defined. It is important to highlight that the appropriate definition of the protocol is essential to correctly identify the requirements of a reference architecture. This protocol must contain: the research questions, search strategy, search sources and inclusion/exclusion criteria.
The research questions aim at finding all information (i.e. the primary studies) necessary to provide evidences about the domain in which the reference architecture will be proposed. In other words, these questions should be formulated to include primary studies involving four important points to establish the requirements of a reference architecture. These points are: software systems of a given domain, architectures of these software systems and functionalities of these software systems; when available, they must also include previous reference architectures/models. For instance, if it is intended to establish a reference architecture for e-commerce systems, i.e. an architecture for e-commerce domain, the research questions must be formulated to find primary studies that envolve e-commerce systems, architectures of e-commerce systems, as well as other reference architecture of e-commerce

systems, and functionalities of e-commerce systems. Examples of research questions are: "Which are the architectures of e-commerce systems that are available?" and "Which are the functionalities presented by e-commerce systems?".

In order to establish the search strategy, considering the research questions, the main keywords and its related terms must be defined. It is suggested the use of boolean OR operator to link the main keywords and their related terms and, then, all these terms combined using the boolean AND operator [12]. Considering the reference architecture for e-commerce systems, an example of main keyword could be "e-commerce" and its related term could be "electronic commerce". The search string should be structured in order to be simple enough to bring many results and, at the same time, rigorous enough to cover only the desired research topic of domain.

In addition to the research questions and search strategy, it is necessary establish which search sources (i.e., publication databases) would be used to find the primary studies. These criteria are deeply discussed in [6]. An appropriate set of database to conduct systematic review in software engineering has been defined by Dybå et al. [8]. Besides these, Kitchenham et al. [13] advocate that the *Scopus* must be included, since it is considered the largest database of abstracts and citations.

Another important element of the systematic review planning is to define the inclusion criteria and exclusion criteria. These criteria make possible to include primary studies that are relevant to answer the research questions and exclude studies that do not answer them. Thus, it is quite important at this time a clear definition about which primary studies are relevant to the establishment of the requirements of the intended reference architecture.

- **Phase 2 - Conduction:** In this phase, the search by primary studies is conducted according to previously established plan. This identification is done by looking for all primary studies that match with the search string in the search sources. This could be automatically conducted, since these sources provide sometimes a search engine. Following, the primary studies are selected and evaluated through reading of titles and abstracts of the primary studies and application of the inclusion and exclusion criteria. Besides that, the primary studies are read in full and inclusion and exclusion criteria are again applied. Again, considering the reference architecture for e-commerce systems, for instance, if the primary study presents an architecture of an e-commerce system, this study must be included. Otherwise, if the study presents, for example, usability evaluation of e-commerce systems, it must not be considered. Thus, a set of selected primary studies are obtained and it can be considered as the most relevant to our interest.
- **Phase 3 - Requirement Establishment:** In this phase, based on the selected primary studies, information must be carefully extracted and synthesized, resulting in a set of requirements of the reference architecture and in a set of related concepts that must be considered in the architecture. For this, three main tasks are conducted: (i) requirements of systems or of other reference architectures of the domain are identified. These require-

ments must reflect the processes/activities/tasks that must be automated by systems of that domain; (ii) based on those system/reference architecture requirements, a unique set of architectural requirements is established. Probably more than one requirement will be aggregated in an architectural requirement. It is observed that architectural requirements are more comprehensive than the system requirements, since they describe the requirements of a set of systems of the domain; (iii) architectural requirements are mapped in concepts, aiming at facilitating the design of the reference architecture.

4 Case Study

To show the viability of SyRRA, we have presented a case study in order to establish the set of requirements of a reference architecture for the software testing domain. In short, software testing is one of the most important activities to guarantee the quality and the reliability of the software under development [18]. In this context, the availability of testing tools has made also the testing a more systematic activity, minimizing cost, time consumed, as well as errors caused by human intervention. Testing automation is therefore an important issue related to the quality and productivity of the testing processes and, as a consequence, of the software processes. A myriad of testing tools — commercial, academic, and open source — automating software testing tasks can be found, such as [15,16]. However, these tools have almost always been implemented individually and independently, presenting its own architectures and data structures. As a consequence, difficulty of integration, evolution, maintenance, and reuse of these tools is very common. In this context, reference architectures can be used as basis to develop testing tools. Thus, for that domain, reference architectures can be found [9,19]. However, these architectures does not support the development of software testing tools that are based on SOA (Service-Oriented Architecture) [3,21], i.e., testing tools organized as services. Thus, in this case study we intend to establish a SOA-based reference architecture.

SOA has arisen as a new architectural style to develop software systems. It has been recently focus of considerable attention of the academy and industry. In SOA, software functionalities are packaged in independent, self-contained and well-defined modules, called services, that are the basis to compose more complex service-oriented systems. SOA intends to contribute with low coupling systems and, as a consequence, it can promote reuse and productivity in software development [23]. However, in spite of the relevance of SOA, there are still challenges to create efficient solutions using this architectural style [3].

In order to support design the service-oriented reference architecture for the software testing domain, we have applied SyRRA, aiming at systematizing the tasks required to obtain requirements to this architecture. It is worth to highlight that in this case study we are specifically interested in using SyRRA to identify requirements related to SOA, since requirements related to testing domain will be reused of RefTEST.

4.1 Applying SyRRA

We have applied the three phases of SyRRA in order to obtain requirement to our reference architecture. We have carefully planned, conducted and established the set of requirements. Following, these phases are discussed in more details:

- **Phase 1 - Planning:** In this phase, we have prepared the review protocol. For this, we have established the research questions, search strategy, search sources, inclusion and exclusion criteria.
  Adequate research questions are essential to successfully establish the set of requirements. In our case, we are interested in identifying: (i) guidelines to the service-oriented system development; and (ii) service-oriented reference architectures for different domains. We believe that these elements are sufficient to be basis to the set of requirements of our reference architecture. The research question must reflect therefore these elements. Thus, our main research questions are: "Which SOA characteristics have been considered during the design and development of reference models and reference architectures? " and "What are the "inputs" (set of information) that support the development of service-oriented reference architectures and service-oriented reference models?".
  Considering these research questions, we identified the main keywords: "Reference Architecture" and "Service Oriented Architecture". Following, we found related terms for these keywords: "Reference Model", "Service based", "Service Oriented", and "SOA". The chosen keywords must be simple enough to bring many results and, at the same time, rigorous enough to cover only the desired research topic. We used the boolean OR operator to link the main terms and their related terms. Eventually, all these terms were combined using the boolean AND operator. The final search string was: *(("Reference Architecture" OR "Reference Model") AND ("Service Oriented Architecture" OR "Service based" OR "Service Oriented" OR SOA))*. The search sources of our systematic review were: *ACM Digital Library*, *IEEE Xplore*, *ScienceDirect*, *Scopus*, *Springer*, *Web of Science*.
  Another important task is to define the Inclusion Criteria (IC) and Exclusion Criteria (EC). These criteria make possible to include primary studies that are relevant to answer the research questions and exclude studies that do not answer them. Thus, the inclusion criteria are: (i) The primary study presents a service-oriented reference architecture or a service-oriented reference model; and (ii) The primary study presents some experience involving a service-oriented reference architecture or service-oriented reference model. Otherwise, the exclusion criteria are: (i) The primary study presents a reference architecture or reference model; however, it involves a specific characteristic or a part of SOA (for instance, reference architecture for systems that support *Enterprise Service Bus* (ESB) or systems that manage *Service Level Agreement* (SLA)); (ii) The primary study presents a reference architecture or a reference model to other types of systems that do not contain features related to service; and (iii) The primary study is related to SOA,

but it does not propose or discuss reference architectures or reference models. Thus, considering the review protocol, the conduction of the systematic review was conducted in the next phase.

– **Phase 2 - Conduction:** In this phase, the systematic review was then conducted according to the review protocol. The search by primary studies was conducted by looking for all primary studies that match with the search string in the search sources. As result, 181 primary studies were identified. After that, the primary studies was selected, through reading of titles and abstracts and application of the inclusion and exclusion criteria. Thus, a total of 38 primary studies were read in full and inclusion and exclusion criteria were again applied. It is worth to highlight that during the selection task, it is important to bear in mind that only primary studies that could contribute to obtain requirements of the reference architecture must be considered. Finally, six studies, as showed in Table 1, were considered the most relevant to the requirement elicitation. This table presents the authors, publication year and a brief description of the primary studies. Based on these primary studies, in the next phase, we have extracted the requirements of our reference architecture.

Table 1. Selected primary studies

N.	Authors	Year	Brief Description
S1	Arsanjani, A. et al. [3]	2007	A detailed definition of reference architecture based on SOA.
S2	Dillon, T. et al. [7]	2008	Discusses evolution of a reference architecture of the multimedia domain in the SOA context.
S3	Lan, J. et al. [14]	2008	Practices and guidance for the development process of SOA systems.
S4	OASIS [21]	2006	Definition of the essence of SOA, the vocabulary and the common understanding of SOA.
S5	OASIS [22]	2008	A technology-independent abstraction of SOA that describes the relationship, use and ownership of SOA-based systems.
S6	Zimmermann, O. et al. [27]	2009	Discussion about architectural knowledge to create reference architecture based on SOA.

– **Phase 3 - Requirement Establishment:** In order to establish the set of architectural requirements of our service-oriented reference architecture for the testing domain, we have again conducted a full reading of each primary study (listed in Table 1), aiming at extracting requirements, and written down a set of requirements provided by each study. Following, we have carefully analyzed each set of requirements and synthesized them in a unique set of requirements. At the end, 39 requirements were identified. In Table 2, it is listed part of these requirements. Column 2 refers to the requirements and column 3 is related to the sources (the primary studies) that contributed to establish that requirement. For instance, requirement AR1 (Architectural

Requirement 1) has three sources: primary studies S1, S4 and S5. Besides that, in order to facilitate the posterior design of the reference architecture, column 4 presents the concept related to each requirement. Thus, the concepts found in our case study are: Service Description (SD), Service Interaction (SI), Service Publication (SP), Quality of Service (QoS), Service Composition (SC), Polices (P) and Governance (G). Considering AR1, this requirement is related to concept SD.

Table 2. Part of Requirements of the Intended Reference Architecture

N.	Requirement	Source	Concept
AR1	The reference architecture must make possible the development of testing tools that support the storage, referencing, and access of normative descriptions that define how to interact with a service.	S1, S4, S5	SD
AR2	The reference architecture must make possible the development of testing tools that support the storage, referencing, and access of semantic models that enable categorization of services.	S1, S4, S5	SD
AR3	The reference architecture must make possible the development of testing tools that support the publication of service description directly to consumers.	S4, S5	SP
AR4	The reference architecture must make possible the development of testing tools that support the publication of service description through mediators.	S1, S2, S4, S5	SP
AR5	The reference architecture must make possible the development of testing tools that provide a mechanism to organize the collaboration among services using orchestration and/or choreography.	S1, S4, S5	SC
AR6	The reference architecture must make possible the development of testing tools that support to capturing, monitoring, logging, and signaling non-compliance with nonfunctional requirements	S1, S5, S6	QoS
...	...	...	...

These architectural requirements related to SOA and the architectural requirements related to testing domain and provided by RefTEST [20] have composed therefore the architectural requirements of our service-oriented reference architecture for testing domain. These requirements are an important and essential input to the reference architecture design phase. In our work, we have used ProSA-RA in order to design and evaluate our reference architecture.

4.2 Discussions

The establishment and use of SyRRA have provided us with feedback about how we can benefit from Systematic Review technique in order to elicit requirement of reference architectures. Case studies have showed that it is specially important when the intended reference architecture must encompass a whole knowledge of a domain. SyRRA provides means to possibly find all published works related

to that domain; therefore, it makes possible to cover all published knowledge that is available at least in the search sources (i.e., publication databases), such as *ACM Digital Library*. However, reference architectures for a specific company could have benefits using our approach only if it is intended to have a more general architecture. Furthermore, it is known that it is not trivial to adequately apply Systematic Review, since it may require considerable manual efforts and full reading of works. Moreover, in spite of positive results presented by SyRRA, it must be conducted together other requirement elicitation techniques, such as interviews, documentation review and brainstorming. Therefore, SyRRA intends to be an additional technique, aiming at gathering the knowledge that is sometimes spread in different publication databases.

5 Conclusion

The establishment of reference architectures can bring a significant contribution to areas in which software systems need to be developed. Thus, approaches that contribute to establish these architectures are very relevant. In this context, the main contribution of this paper is to propose the use of Systematic Review technique to elicit requirements of reference architectures, consolidated in a process named SyRRA. Qualitative results of our case studies have pointed out that SyRRA is an effective process to elicit requirements of reference architectures, since these requirements must represent a range of domain knowledge and SyRRA has the ability to cover this need. However, more studies must be conducted yet, aiming at refining our approach and having quantitative evidences about Systematic Review as an effective technique to the requirement elicitation.

Motivated by the promising results of our approach, as future work, we intend to apply SyRRA for other reference architectures, aiming at evolving SyRRA and contributing to development of reference architectures for different domains and, as a consequence, promoting reuse and productivity in the software development.

Acknowledgments: This work is supported by Brazilian funding agencies (FAPESP, CAPES, and CNPq).

References

1. S. Angelov, P. W. P. J. Grefen, and D. Greefhorst. A classification of software reference architectures: Analyzing their success and effectiveness. In *WICSA'09*, pages 141–150, Cambridge, UK, Sep 2009.
2. S. Angelov, J. J. Trienekens, and P. Grefen. Towards a method for the evaluation of reference architectures: Experiences from a case. In *ECSA'08*, pages 225–240 (LNCS .5292), Paphos, Cyprus, 2008. Springer-Verlag.
3. A. Arsanjani, L.-J. Zhang, M. Ellis, A. Allam, and K. Channabasavaiah. S3: A service-oriented reference architecture. *IT Professional*, 9(3):10–17, 2007.
4. L. Bass, P. Clements, and R. Kazman. *Software Architecture in Practice*. Addison-Wesley, 2003.
5. J. Bayer, T. Forster, D. Ganesan, J. F. Girard, I. John, J. Knodel, R. Kolb, and D. Muthig. Definition of reference architectures based on existing systems. Technical Report 034.04/E, Fraunhofer IESE, 2004.

6. O. Dieste, A. Grimán, and N. Juristo. Developing search strategies for detecting relevant experiments. *Empirical Software Engineering*, 14(5):513–539, 2009.
7. T. S. Dillon, C. Wu, and E. Chang. Reference architectural styles for service-oriented computing. In *IFIP/NPC'07)*, pages 543–555 (LNCS v.4672), Dalian, China, Sep 2008. Springer-Verlag.
8. T. Dybå, T. Dingsoyr, and G. K. Hanssen. Applying systematic reviews to diverse study types: An experience report. In *ESEM'07*, pages 225–234, Los Alamitos, USA, 2007. IEEE Computer Society.
9. N. S. Eickelmann and D. J. Richardson. An evaluation of software test environment architectures. In *ICSE'96*, pages 353–364, Berlin, Germany, 1996.
10. U. Eklund, Örjan Askerdal, J. Granholm, A. Alminger, and J. Axelsson. Experience of introducing reference architectures in the development of automotive electronic systems. *SIGSOFT Softw. Eng. Notes*, 30(4):1–6, 2005.
11. R. Farenhorst and R. C. Boer. *Software Architecture Knowledge Management*, chapter Knowledge Management in Software Architecture: State of the Art, pages 21–38. 2009.
12. B. Kitchenham. Procedures for performing systematic reviews. Technical Report TR/SE-0401 and NICTA Technical Report 0400011T.1, Keele University and National ICT Australia Ltd, Jul 2004.
13. B. Kitchenham and S. Charters. Guidelines for performing systematic literature reviews in software engineering. Technical Report EBSE 2007-001, Keele University and Durham University Joint Report, 2007.
14. J. Lan, Y. Liu, and Y. Chai. A solution model for service-oriented architecture. In *WCICA'08*, pages 4184–4189, Chongqing, China, Jun 2008.
15. Y.-S. Ma, J. Offutt, and Y.-R. Kwon. MuJava: a mutation system for Java. In *ICSE'06*, pages 827–830, New York, NY, 2006.
16. J. Misurda, J. Clause, J. L. Reed, B. R. Childers, and M. L. Soffa. Demand-driven structural testing with dynamic instrumentation. In *ICSE'05*, Missouri, USA, 2005.
17. G. Muller. A reference architecture Primer. [*On-line*], *World Wide Web*, 2008. Available: `http://www.gaudisite.nl/` (Access: 11/14/2010).
18. G. J. Myers, C. Sandler, T. Badgett, and T. M. Thomas. *The Art of Software Testing*. John Wiley & Sons, 2nd. edition, 2004.
19. E. Y. Nakagawa, R. Martins, K. Felizardo, and J. C. Maldonado. Towards a process to design aspect-oriented reference architectures. In *CLEI'2009*, pages 1–10, Pelotas, Brazil, Sept. 2009.
20. E. Y. Nakagawa, A. S. Simão, F. Ferrari, and J. C. Maldonado. Towards a reference architecture for software testing tools. In *SEKE'07*, pages 1–6, Boston, USA, 2007.
21. OASIS. Reference model for service oriented architecture 1.0. Technical report, OASIS Standard, Oct 2006.
22. OASIS. Reference architecture for service oriented architecture version 1.0. Technical report, OASIS Standard, Abr. 2008.
23. M. P. Papazoglou, P. Traverso, S. Dustdar, and F. Leymann. Service-oriented computing: a research roadmap. *Int. Journal of Cooperative Information Systems*, 17(2):223–255, 2008.
24. H. Pei-Breivold and I. Crnkovic. A systematic review on architecting for software evolvability. In *ASWEC'10*, pages 13–22, Auckland, New Zealand, apr 2010.
25. M. Shaw and P. Clements. The golden age of software architecture. *IEEE Software*, 23(2):31–39, Mar/Apr 2006.
26. I. Sommerville and P. Sawyer. *Requirements Engineering: A good practice guide*. John Wiley and Sons, 1997.
27. O. Zimmermann, P. Kopp, and S. Pappe. Architectural knowledge in an SOA infrastructure reference architecture. In *Software Architecture Knowledge Management*, pages 217–241. Springer-Verlag, 2009.