Towards the Open Source Reference Architectures

Elisa Yumi Nakagawa and José Carlos Maldonado
Department of Computer Systems
University of São Paulo - USP
PO Box 668, 13560-970
São Carlos, SP, Brazil
E-mail: {elisa, jcmaldon}@icmc.usp.br

Abstract—Software architectures have received increasing attention by playing a significant role in determining the success and quality of software systems. In particular, reference architecture is a special type of architecture that captures the essence of software systems of a specific domain, achieving therefore well-recognized understanding of that domain. In spite of this, it is not observed concern in order to widely disseminate reference architectures and, as a consequence, the knowledge encompassed by these architectures. At the same time, Open Source Software (OSS) has been largely developed and used in both academy and industry. A diversity of OSS has been made available and has contributed to the software development through dissemination of knowledge that is encompassed mainly in the source code. Specifically, its success is due to OSS licences that have adequately supported its development and evolution. Thus, applying this same idea in order to disseminate reference architectures seems to be very interesting. The main contribution of this paper is to propose Open Source Reference Architecture (OSRA) that, based on principles of OSS, aims at promoting dissemination and evolution of reference architectures, intending to contribute to a more effective software development.

Keywords: Reference Architecture, Open Source, Open Source License

I. INTRODUCTION

Software architecture has received increasing attention as an important subfield of the Software Engineering [31], [53]. The software architecture of a computing system is the set of structures needed to reason about the system, which comprises software elements, relations among them, and properties of both [9]. Software architectures play a major role in determining system quality — for instance, performance, maintainability, and reliability — since they form the backbone of any successful software-intensive system [57]. 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 can also promote reuse of design expertise by achieving solid, well-recognized understanding of a domain. Considering their significant relevance, diverse application domains have proposed and used these architectures. Examples of domains with consolidated reference architectures are embedded software [7], [15], [44], software engineering tools [14], [38], [41], web browsers [27] and computer games [12]. Other examples of reference architectures are found in [2], [45]. Specifically, two good examples are UniversAAL [56], for Ambient Assisted Living (AAL), and AUTOSAR [4], for the automotive domain, both developed by consortiums that involve major industrial players (such as, manufacturers and suppliers) and researchers. Thus, it is noticed that different domains have already understood the need of encompassing knowledge in reference architectures, aiming at disseminating and reusing this knowledge and standardizing the systems as well. In this perspective, they can be seen actually as a knowledge repository that should be disseminated and reused.

At the same time, Open Source Software (OSS), or Free and Open Source Software (FOSS), has arisen to great prominence within the last years, mainly due to the success of quite known projects, such as the Apache\(^1\) web server and the GNU/Linux\(^2\) operating system. In particular, in order to understand the success of these projects, studies focusing on the OSS development process have also been conducted [1], [33], [52]. In general, this process presents an openly cooperative, geographically distributed effort involving sometimes several stakeholders, such as developers, testers, and users. In general, improvement in the software quality, satisfaction of the user’s requirements, and lower costs can be considered as the main benefits provided by the OSS process. In this context, software licenses have been a key element to the success of OSS. Essentially, a software license corresponds to a legal instrument governing the usage or redistribution of software systems and establishing their copyright. In particular, OSS licenses aim at preserving the openness of the software itself, giving freedom to users to mainly access the source code and to get knowledge freely available in their code, regarding, for instance, how to structure a system, how to use some technology, and so on. Considering its importance, several OSS licenses can be found [23], [47]. In particular, GPL

\(^1\)http://www.apache.org/
\(^2\)http://www.gnu.org/
widely used OSS license. For instance, most of the OSS available in SourceForge\(^3\), one of the most well-known OSS repositories, are licensed under GPL.

The idea under OSS was also extended to documents (for instance, software manuals, textbooks, and dictionaries) and licenses have been proposed to them. Two well-known examples of license are GNU FDL (Free Documentation License) [22] and Creative Commons copyright licenses (or simply CC licenses) [10]. Therefore, source code as well as documents have been made freely available, aiming at disseminating them and, consequently, the knowledge encompassed by them.

In this context, despite some initiatives that have investigated the role of software architectures in the OS context, i.e., relationship between software architecture and OSS [3], [20], [32], [42]; they have not addressed the relationship between reference architecture and OSS. There is also a lack of work that propose dissemination and free distribution of reference architectures.

The main objective of this paper is to propose the Open Source Reference Architecture (OSRA), a means by which reference architectures can be freely used, distributed, and evolved. In other words, we propose means for knowledge in reference architectures to be disseminated and continue freely available for all that intend to use these architectures. We present also initiatives of making our reference architectures as OSRA. As main result, we have observed that there are advantages of adopting OSS philosophy in reference architectures and we hope that the Software Engineering community, including software system developers, could take advantages of OSRA.

This paper is organized as follows. In Section II background about reference architecture and OSS are presented. In Section III we present OSRA. In Section IV we present our initiatives of OSRA. In Section V we present a brief discussion about our work. Finally, in Section VI we summarize our conclusions, contributions, and future perspectives.

II. BACKGROUND

From the first work of Kruchten on iterative software development with a focus on software architecture [29], a number of work has recognized the value of considering explicitly software architectures in system development processes [6], [30], [58]. Software Architecture research area has grown up and accumulated important knowledge, including contribution to the software quality. Considering the relevance of software architecture, research in several directions has been conducted. One of these directions is the proposal, representation, and use of reference architectures.

A reference architecture plays a dual role with regard to specific target software architectures [25]: it generalizes and extracts common functions and configurations; and it provides a base for instantiating target systems that use that common basis. It is important to highlight that, in a previous work [37], we have gone in the direction of establishing a first definition of reference architecture, since the Software Architecture community did not have this definition. Thus, in this work, we will adopt this definition that considers reference architecture as an element that encompasses the knowledge about how to design concrete architectures of systems of a given application domain; therefore, it must address the business rules, architectural styles (sometimes defined also as architectural patterns\(^4\)), best practices of software development (for instance, architectural decisions, domain constraints, regulations, and standards), software elements that support development of systems for that domain, and all supported by a unified, unambiguous, and widely understood domain terminology [37].

It is worth highlighting that the Software Product Line (SPL) community has also sometimes adopted the term reference architecture to refer to product line architecture, a special type of software architecture used to build a product line, describing explicitly commonality and variability (i.e., information about how the individual product varies in a SPL), being the basis to the architectures of all product line members [49]. Considering the lack of work that differentiates reference architecture and product line architecture [37], we have considered that reference architecture, as adopted in this work, does not comprise the variabilities and it can be, in general, considered as an architecture in a higher abstraction level if compared with product line architecture and, therefore, they could be the basis for product line architectures. In other words, a reference architecture can result in different specializations (i.e., software product architectures), each one for a specific SPL.

In another perspective, the Open Source (OS) community has made available a substantial number of OSS, mainly through OSS repositories, such as SourceForge, Freshmeat\(^5\), CodigoLivre\(^6\), and Google Code\(^7\). These repositories have as main goal to publish, find, and promote development and evolution of OSS using web platform and its facilities. Besides that, this community has also published results of researches on OSS and its development process [1], [33], [50]. It has also conducted studies focusing the investigation of quite known OS projects, such as Linux, Mozilla, and

\(^3\)http://sourceforge.net

\(^4\)We have considered an architectural pattern as a type of software pattern (in the same direction of GoF design patterns [26]) that addresses quality attributes into the reference architecture.

\(^5\)http://freshmeat.net/

\(^6\)http://codigolivre.org.br/

\(^7\)http://code.google.com/
Apache [33], [54], [55], in order to understand how these projects have achieved success. In this context, the single best description and widely referenced OSS development process is “The Cathedral and the Bazaar” [50]. While Cathedral represents conventional commercial practices, where developers work using a relatively closed and centralized methodology, Bazaar represents the openly cooperative effort of OSS development. A set of main OS practices that characterize this process can be identified [52]: (i) requirement identification activity is not conducted; (ii) software implementation is logically centralized, while being physically distributed in an autonomous and decentralized manner; (iii) developers have usually good experience in coding and ability of cooperative distributed work; and (iv) Internet resources are intensively used. Besides these practices, it is also known that there are other important issues related to OS processes, such as a new social-technical network to software development. In short, the means by which OSS have been developed has certainly contributed to the development of a considerable number of OSS and has pointed out also to the relevance of this means to develop and evolve OSS.

In parallel, since documentation is an essential part of any software system and the lack of good free manuals to OSS is very common, the idea of free documentation was proposed with specific licenses for it. This idea is pretty much the same as for OSS, giving all users freedom to use, redistribute, and modify the documents. Furthermore, authors and publishers can get credit from their documents. Thus, applying this idea also to reference architectures seems to be very interesting.

III. OPEN SOURCE REFERENCE ARCHITECTURE

It is observed that both academy and industry, as well as particular initiatives, have been developing similar systems for a long time and have accumulated a huge amount of knowledge about diverse application domains, both technical and contextual [35]. In general, this knowledge is partially implicit, for example in the heads of designers and engineers, and partially explicit in design repositories and documentation. Thus, making this knowledge explicit by means of reference architectures is interesting. Furthermore, it is also interesting that these architectures together with knowledge encompassed by them can be freely shared, in the sense of using, updating, and evolving them.

In this perspective, we have proposed the term “Open Source Reference Architecture” (OSRA) to designate a reference architecture freely available by any person, group, or organization that would like to share the knowledge contained in such architecture with all SE community. In particular, we adopted the term “Open” in the same perspective of the OS community that has been concerned with providing freedom to the users, in the sense of, for instance, using and redistributing software systems; however, we have considered reference architecture instead of software systems. Besides that, we have also adopted “Open Source” in order to state that the “source” of the reference architecture should be available too. For instance, if the reference architecture was designed in UML using a modeling tool, the source files should be also freely available. Moreover, it is very interesting that the OS modeling tools used to describe the reference architectures are open source too. A number of modeling tools are already available as OSS; two good examples are ArgoUML\(^8\) and StarUML\(^9\). The use of free tools makes it possible for someone interested in using reference architectures to get these tools more easily than if compared with commercial tools, and to access the source files of the reference architectures. Thus, it is intended that in fact reference architectures are more easily disseminated.

A similar term — “Open Reference Architecture” — has been already used; however, with a different meaning. They refer to architectures that have as basis existing open standards, aiming at achieving integration of, for instance, different technologies and platforms. UniversAAL is an example of an open reference architecture for AAL (Ambient Assisted Living) [56].

It is well-known that in OSS, source code written in diverse programming languages are available. Regarding reference architectures, an effective reuse of knowledge of these architectures depends not only on raising the domain knowledge, but also documenting and communicating efficiently this knowledge through an adequate architectural description. However, there is not a consensus in the literature about how to build this architectural description [28]. For instance, in general, each reference architecture has been described in a different way, mainly using an informal box-and-line representation. Thus, in a previous work [39], we have established a set of architectural views to represent reference architectures. In this perspective, to represent OSRA, we have initially suggested three views and UML techniques:

- module view shows the structure of the architecture in terms of code unites. Packages and classes can be used to represent this view, as well as containment, specialization/generalization, and dependency relations. For this view, the UML class diagram is an adequate technique;
- runtime view shows the structure of the software system built from the architecture when this system is executing. The elements that compose this view are components that have runtime presence, data and connectors. The UML component diagram can be used to represent this view; and

\(^8\)http://argouml.tigris.org/
\(^9\)http://sourceforge.net/projects/staruml/
• deployment view describes the machines, software that is installed on which machines and network connections that are used by the software systems resulted from the architecture. Thus, an adequate technique is UML deployment diagram. Additionally, we have observed that only these views and UML diagrams are not sufficient to completely represent reference architectures, since these diagrams do not provide elements to explain each concept/term that is present in these architectures. We have proposed an additional view, named conceptual view [39], that aims to describe each concept of the domain. For describing this view, ontologies, controlled vocabularies, taxonomies, thesauri, concept maps, among others can be used as a supporting element to describe the terminology related to that domain. Therefore, OSRA could be represented using these four architectural views and related UML techniques, aiming at its more effective dissemination. Besides that, ADL (Architectural Description Language) could be also used to represent reference architectures. From a previous work [28], we can identified two main ADLs — WRIGHT [24] and π-ADL [8] — that have been used to represent reference architectures. In general, these ADLs intend to describe reference architectures in a more formal perspective [8].

To guarantee the freedom on reference architectures, we have proposed the use of a legal instrument: an Open Source Reference Architecture License (OSRA). This license is also responsible to establish the copyright of the OSRA, i.e., a set of exclusive rights granted to the author or creator of the reference architecture, including the right to copy, distribute, and adapt the architecture. The copyright establishes also who is the owner of the reference architecture.

A. Reference Architecture Licenses

In order to establish which licenses are more adequate for reference architectures, we have investigated licenses for OSS and published by Free Software Foundation\(^\text{10}\) (FSF), Open Source Initiative\(^\text{11}\) (OSI), and Creative Commons\(^\text{12}\) (CC). In the essence, licenses for documentation better correspond to our expectations, since reference architectures, more specifically, their architectural description, can be considered as documents. Thus, we have suggested that, at first, an OSRA is available under a documentation license. There is a number of documentation licenses, each one with its characteristics and particularities. Main examples are presented in Table I. In particular, GNU FDL and CC licenses could be used as RAL.

The purpose of GNU FDL is to make a document, including manual, textbook, or other functional and useful document, “free” in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially [22]. Secondarily, this license preserves the right for the authors and publishers to get credit for their work. Besides that, derivative works of this document must themselves be free in the same sense.

In the same perspective, CC licenses allow the distribution of copyrighted works, such as books, movies, music, articles, photographs, and so on [10]. CC makes possible combinations of four main conditions (Attribution, Noncommercial, No Derivative Works, and Share-alike), resulting in eleven valid CC licenses. For instance, the Attribution + Noncommercial + ShareAlike (or simply CC BY-NC-SA) license makes it possible to copy, distribute, display, and perform the work and to create derivative works based on it only for noncommercial purposes; furthermore, derivative works must be distributed only under a license identical to the license that governs the original work. Besides these RAL, other modified or derived licenses could be written, if there is a need to adjust terms and conditions of the license according to specific interest on the OSRA.

For a reference architecture to become an OSRA, i.e., to license a reference architecture under a RAL, the process is similar to that performed for OSS. The process involves adding two elements to each source file of the reference architecture: (i) a copyright notice; and (ii) a statement of copying permission, saying that the reference architecture is distributed under the terms of the license. The copyright notice should include the year in which the reference architecture was built, in addition to the name of the individuals or organizations responsible for the architecture. An example of copyright notice is “Copyright 2005 John Smith”. Furthermore, a copy of the license itself, in the textual version, should be included together with the source files of the reference architecture. In the same perspective of OSS, this textual version might be in a file called COPYING.

IV. OUR INITIATIVE OF OSRA

Our research group has been working in the last years establishing reference architectures for different domains, such as software testing [41], [46], computer game [34], mobile robot [16], software engineering environment [38], educational environment [21], and visual mining [40]. Our idea is to make all these architectures available as OSRA, since they are the result of considerable research efforts and we believe that they could be a valuable basis to the development of new systems in those domains or even to conduct architectural refactoring of existing software systems. Moreover, we intend that, if available as OSRA, these architectures could receive important contributions to their update and evolution from the community. Following,
we discuss how we have made one of our reference architectures — the RefTEST (Reference Architecture for Testing Tools) [41] — available as an OSRA.

A. Making a Reference Architecture as an OSRA

RefTEST is a reference architecture for software testing domain that aims at supporting development of tools for that domain. It provides information about modules and sub-modules that compose testing tools, a set of functional requirements of testing tools, as well as architectural styles and patterns to organize these tools. Thus, it intends to comprise the knowledge of the testing domain, aiming at improving reuse, evolvability, and maintainability of tools built from it. Thus, it seems to be in fact interesting to make it available as an OSRA. In order to facilitate its dissemination, RefTEST is then available by means of: (i) a requirement document containing architectural requirements that refer to requirements of the reference architecture that describe textually common functionalities and configurations presented in testing tools and; (ii) four architectural views — module view, runtime view, deployment view, and conceptual view — represented in UML.

In order to collaborate to the Software Testing community, we have made RefTEST available as an OSRA. In particular, we have adopted GNU FDL as the RAL for RefTEST. To illustrate the license adopted for RefTEST, Figure 1 shows the first part of GNU FDL, contained in the COPYING file and available together with other source files of this architecture.

An important part of RefTEST is the requirement document. Since RefTEST is an OSRA, the beginning of this document presents information about copyright and the years when RefTEST was created and evolved — “Copyright (C) 2006-2011 Elisa Yumi Nakagawa and José Carlos Maldonado (ICMC/USP)” — as illustrated in Figure 2. Furthermore, other source files related to four architectural views have also this same information. Figure 3 illustrates three of these views. In Figure 3.a, the module view of RefTEST is presented; in Figure 3.b, the runtime view is presented; and in Figure 3.c, the deployment view is showed. In order to illustrate the copyright notice, Figure 4 detaches that one of module view of RefTEST.

The fourth view of RefTEST — the conceptual view — is represented by an ontology that establishes the domain concepts (i.e., the domain terms) besides definition of each term and relationships among terms. Hence, we have used OntoTest [5] as the conceptual view of RefTEST, contributing to a more complete architectural description. In order to illustrate the conceptual view of RefTEST, part of OntoTest is presented in Figure 5. Furthermore, as illustrated for

Test Case in this figure, each concept has a description. Thus, this conceptual view provides relationship among concepts, as well as the description of each concept, both of which can improve the understanding of the RefTEST. This view presents also information about copyright as in other views. More detailed information about each view can be found in [36]. RefTEST has been used to the development of new software testing tools and to the architectural reorganization of existing testing tools as well. RefTEST has also been modified, updated, and evolved considering new approaches, such as, new architectural styles and patterns.

B. Using RefTEST

RefTEST has been successfully used in the development of tools for the software testing domain [18], [43], [51]. For instance, Proteum/AJ was developed using knowledge contained in RefTEST; i.e., it was developed through specialization of the RefTEST for the mutation testing of aspect-oriented programs, specifically written in AspectJ [18], [19]. Proteum/AJ automates a set of AO-specific mutation operators [17] and supports the basic steps of mutation testing [13], such as mutant handling, test case handling, and mutant analysis.

It is important to observe that the license of RefTEST imposes that derivative work from RefTEST must be freely distributed; therefore, any testing tool built from RefTEST must be also an OSS. Thus, Proteum/AJ and other derived tools have been also made available under a license for OSS. Proteum/AJ will be available soon as OSS 14.

Besides development of testing tools, RefTEST has been used as basis to the establishment of another reference architecture for the testing domain, but one based on SOA (Service-Oriented Architecture) [48]. We have then established RefTEST-SOA (Reference Architecture for Software Testing Tools based on SOA) [46] that comprises knowledge and experience about how to structure testing tools organized as services, aiming at pursuing a better integration, scalability, and reuse provided by SOA. By being a derivative work of an OSRA and considering the license of RefTEST, this and other reference architectures must be made available as an OSRA.

V. BRIEF DISCUSSION

Since reference architectures aggregate consolidated knowledge from experimented and tested solutions, if freely available, they could certainly bring a considerable impact to a more effective development of software systems. In this section, we briefly discuss some issues related to the impact of OSRA, lessons learned, limitations we have noticed, as well as future work.

The main impacts of OSRA that we have observed are:

12http://fedoraproject.org/wiki/Licensing/Common_Documentation_License
13http://www.eclipse.org/aspectj/
Differently from reuse of source code, OSRA intends to promote reuse in a broader sense, i.e., the reuse of knowledge and experience regarding development of systems of a whole domain. Thus, the concept of OSRA becomes quite interesting;

- Considering that reference architectures also aggregate experimented, better solutions, and architectural decisions regarding software quality for systems of a given domain, systems developed from these architectures have more chances of presenting good quality than those developed from scratch. Therefore, an OSRA contributes to improve the software quality, including important attributes, such as the reliability, portability, and maintainability;
- Since OSRA is accompanied and protected by an RAL, the idea is that the same reasons that motivate people and organizations to make OSS available could motivate them to also make available reference architectures.
- RAL provides guarantee that makes it possible for contributions from the community to be aggregated in the reference architecture, in the same perspective of the Bazaar Model for OSS. These contributions could be scrutinized by the owner of the OSRA;
- When adopting a license that imposes more freedom such as GNU FDL, OSRA contributes to the OS community through development of systems that must be also available as OSS. Moreover, these OSS collaborate also to the entire community of software users (e.g., in academia and industry), who need these software systems.

We have also observed that, in general, reference architectures must be still better represented if it is desired the effective reuse of knowledge contained in these architectures. Thus, if a reference architecture is available as an OSRA, they could provide a detailed architectural description (for instance, represented by architectural views using UML or an ADL), as well as a set of requirements that describe common functionalities to systems for that domain, aiming at achieving better dissemination, use, and evolution.
Fig. 2. Part of requirements of the RefTEST.

Our initiatives of OSRA has achieved qualitative, positive results. For the future, we intend to conduct quantitative studies to observe the impact of OSRA to software development; i.e., which benefits these architectures could bring to the entire software development process. We intend also to establish mechanisms to more easily disseminate these architectures, such as OSRA repositories, in the same perspective of OS repositories, in the same perspective of OS repositories (e.g., SourceForge, Freshmeat, and Google Code), pattern repositories (e.g., Open Pattern Repository\footnote{http://code.google.com/p/openpatternrepository/}, and decision repositories (e.g., Open Decision Repository\footnote{http://code.google.com/p/opendecisionrepository/}). It is also interesting to investigate and establish a set of Internet services, such as revision control systems and discussion forums, to specifically manage, disseminate, update, and evolve reference architectures. Besides that, we also intend to conduct further investigation about the adequacy of other licenses for OSRA, as well as to propose specific licenses for reference architecture, if necessary.

VI. CONCLUSIONS

Considering the relevance of knowledge dissemination to software development, in the same perspective adopted currently by the OS community, the main contribution of this paper is to present the concept of OSRA that refers to reference architectures freely available under OS licenses. Our essential idea is to provide means to make reference architectures available, establishing a way to distribute, use, and evolve these architectures and to encourage groups, organizations, and anyone that has established reference architectures to freely share them.

As future work, we plan to invest in OSRA and in a more effective dissemination and evolution of reference architectures, aiming at improving reuse and, as a consequence, the productivity in software development, which are important concerns of the Software Engineering area.

Acknowledgments: This work is supported by Brazilian funding agencies: FAPESP, CNPq, and Capes.

REFERENCES

Fig. 3. Architectural View of ReTTEST.

(a) Module View

(b) Runtime View

(c) Deployment View
Fig. 4. Copyright Notice in Module View of ReFTEST.

Fig. 5. Part of Conceptual View of ReFTEST


