RARep: a Reference Architecture Repository

Tales Prates Correia*, Milena Guessi*†, Lucas Bueno Ruas Oliveira*‡ and Elisa Yumi Nakagawa*
*University of São Paulo - USP, São Carlos, Brazil
†University of South Brittany - UBS, Vannes, France
‡Federal Institute of São Paulo - IFSP, São Carlos, Brazil
Email: tales.correia@usp.br, {milena,oliveira,elisa}@icmc.usp.br

***Abstract*—Reference architectures are a special type of software architectures that have been proposed for supporting standardization, development, and evolution of software systems of a given domain. As these architectures could contribute for knowledge reuse and increased productivity, several companies have been creating specific reference architectures for their field of expertise. Nonetheless, there is no mechanism that enable recovering, publishing, and sharing existing reference architectures for the public. The main contribution of this paper is to present RARep (Reference Architecture Repository), a web-based reference architecture repository supporting the dissemination of materials related to reference architectures. As a result, this tool can facilitate the access to information on reference architectures and, hence, promote the sharing of architectural knowledge contained in such architectures.**

***Keywords*—Software architecture, reference architecture, web-based tool.**

I. Introduction

Software architectures have played a central role in the development of successful software systems over the past 20 years [15, 25]. Garlan and Perry [10] state that software architectures can have a positive impact in many aspects of software development, such as understanding, reuse, evolution, analysis, and management. Thereby, software architectures also play an important factor for guaranteeing software systems quality [6], such as maintainability, dependability, and interoperability [26].

As a particular type of software architectures, reference architectures have stood out as a structure that provides a characterization for software systems functionalities of a given domain [6, 17]. In other words, reference architectures encompass knowledge about a given domain, which can offer important information on how to build, develop, and maintain systems for that specific domain. In this scenario, the availability of reference architectures becomes an important concern since it impacts the dissemination and reuse of knowledge contained in these architectures as well as the productivity of software development processes. Considering its relevance, a variety of reference architectures for different domains can be found, such as the ones presented at [2, 5, 8, 11, 24]. Reference architectures has also been explored for the domain of embedded systems, such as automotive [4], ambient assisted living [1, 20, 23], and robotics [13].

In this perspective, the main motivation for developing an open repository is the fact that a mechanism to catalog, promote, and share reference architectures is still missing. Moreover, even though many reference architectures are published in articles, technical reports, or thesis, they are not easily available to the interested public. Thereby, we have designed a tool that supports the software architecture community to have easier access to information related to reference architectures, creating an open knowledge repository for software systems developers. Thereby, this repository can be used as an index for several types of material related to reference architectures aiming to facilitate the recovery and selection of reference architectures for the software design.

This paper is organized as follows. Section II introduces references architectures and details relevant works motivating the development of our repository. Section III discusses the requirements of this tool as well as its conceptual design. Section IV details how this repository is implemented and its main features besides an illustrative example of its use. Section V discusses some perspectives for extending this repository in future research. Finally, Section VI presents our final remarks.

II. Background

Decisions made at the architectural level can directly enable, facilitate, or interfere in the achievement of business goals, as well as functional and quality requirements [17]. Several terms are used to designate software architectures in different abstraction levels. In particular, the term reference model is frequently misused as a synonym for reference architecture. A reference model designates a structure that promotes the understanding on a given domain by sharing a common vocabulary and the parts and its interrelationships without considering implementation details [6]. The OASIS Reference Model for Service Oriented Architectures[1] is an example of reference model. On the other hand, reference architectures are less abstract than reference models as they provide concrete guidelines for the design of software architectures, such as architectural styles, best practices for software development, and software elements supporting the development of systems [18].

In this scenario, a reference architecture can combine reference models and architecture patterns and, hence, support the creation of several concrete software architectures that pertain to a given domain (Figure 1). Aiming to systematize

[1]OASIS, https://www.oasis-open.org/committees/download.php/19679/soa-rm-cs.pdf

the creation of reference architectures, several approaches have been proposed. For example, Muller (2008) [16] provides guidelines for establishing reference architectures. Angelov et al. (2012) [3] propose a classification framework for reference architectures that aims at promoting the analysis and design of reference architectures by assessing their adherence to this framework. Nakagawa et al. (2014) [19] present a process to establish, represent, and evaluate reference architectures that is focused on improving separation of concerns in such architectures.

Furthermore, we observe the proposition of a reference model for reference architectures, called RAModel [17], which aims at promoting a better understanding about the content of reference architectures in terms of their main elements and relationships with each other as well as the main tasks for establishing, using, and evolving such architectures. According to the RAModel, a reference architecture contains information about the domain, application, and infrastructure, as well as crosscutting information that is related to several of these dimensions.

The description of reference architectures is fundamental for their adoption and effective use during the software development process. In particular, the standard ISO/IEC/IEEE 42010:2011 specifies the main elements of an architectural description, the relationship with each other, and their organization [14]. The most important elements are: model type, architectural view, viewpoint and correspondence. Since reference architectures are more generic than concrete software architectures, current practices for describing software architectures must be tailored for references architectures [12]. These reference architectures are often described at a higher abstraction level, have no clear stakeholders, involve more architectural qualities, and have a larger scope.

Aiming at disseminating reference architectures to the interested public, an open knowledge repository is needed. This repository can help to publish, share, and find reference architetures that have been created for a given domain of expertise or that follow the same design principles. The design and implementation of such a repository can be inspired in currently available tools, such as one for robotics services semantic search tool, called RoboSeT[2], and another for software patterns, called Portland Pattern Repository[3], for devising a set of relevant features for a novel reference architectures repository.

III. Repository for Reference Architectures

Considering the relevance of reference architectures for promoting best practices in software systems development, we propose a software tool supporting the dissemination of reference architectures, hereinafter referred to as RaRep. This tool contains particular features allowing the interaction among users of the repository. For instance, users of this tool can post comments and up-vote reference architectures, which enables to establish a bidirectional communication channel between creators of a reference architecture and its public. Furthermore, the repository can be used as an index of reference architectures that is directly managed by the creators of reference architectures. As a consequence, we are able to provide an up-to-date catalog of reference architectures for several domains that make available their architectures.

Taking into account the RAModel and the standard ISO/IEC/IEEE 42010:2011, we specify which reference architectures concepts can be used in RaRep for documenting reference architectures. This specification is important since the repository's main goal is to act like a broker of reference architectures, i.e., enabling creators to share their own reference architectures or look for one or more that they could use in their project. In particular, we designed UML [22] models about the main concepts and functionality of this tool. Figure 2 shows the conceptual model for this repository, which identifies the main elements related to the creation and use of reference architectures. This representation simplifies the way users can post, select, and navigate reference architectures in our repository. Figure 3 shows the UML use case diagram for RARep, in which the main actors and actions are defined. In particular, we devise three types of actors in our tool (i.e., common users, registered users, and admin users) with different levels of permissions. For instance, only registered users can publish, comment, and post news about registered reference architectures.

This representation, however, does not preview more specific reference architecture elements such as business rules, constraints, risks, goals and needs. These elements are more related to the knowledge itself encompassed in a reference architecture description. Other elements that can be categorized as the core of architectural knowledge management is the architecture rationale, which represents the decisions made over the project and the alternatives of that decisions [17]. In another context, the main goal of architectural knowledge management is to prevent knowledge vaporization in software architectures. To do so, architecture rationale for significant architectural decisions made in the software architecture should also be included in the architecture description. Significant architectural decisions could be for example the selection of a particular concern or viewpoint, the definition of the most adequate abstraction level for a particular viewpoint, or the reason for adopting a particular design pattern. Furthermore, several aspects of an architectural decision can be relevant, such as their implications to the software architecture design, constraints and rules imposed by them, and also the reasoning that lead to them [7]. Therefore, architectural decisions play an important role for education, reuse, and evolution of software architectures as they are used for sharing expertise and best practices. In the context of reference architectures, significant architectural decisions encompass guidelines for deriving the reference architecture into concrete software architectures besides documenting the architectural knowledge of concrete software architectures of a given domain. Hence, reference architectures certainly need to address architectural decisions

[2]RoboSeT, http://www.labes.icmc.usp.br:8595/RegistroServicoWeb/

[3]Portland Pattern Repository, http://www.patternrepository.com

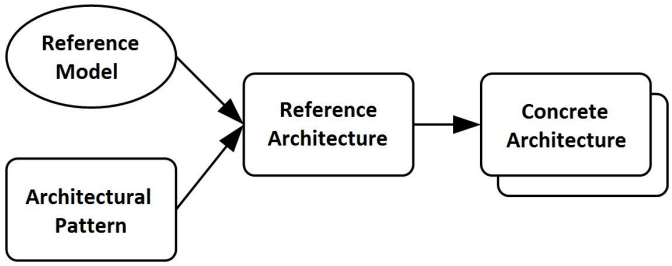

Fig. 1. Relationship among reference model, architectural pattern, reference architecture, and concrete architecture [6]

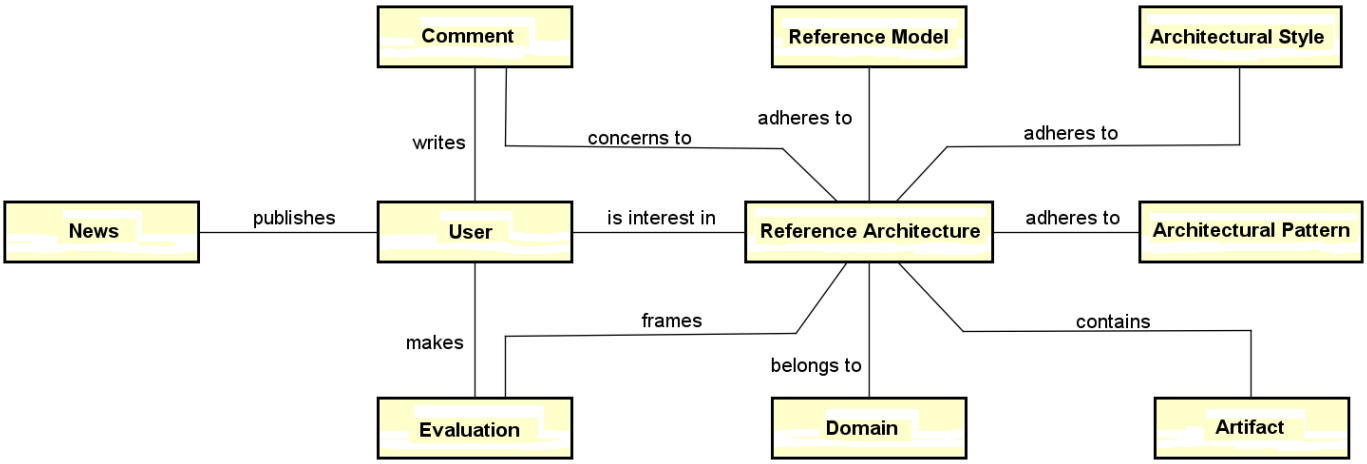

Fig. 2. Conceptual model of the reference architecture repository

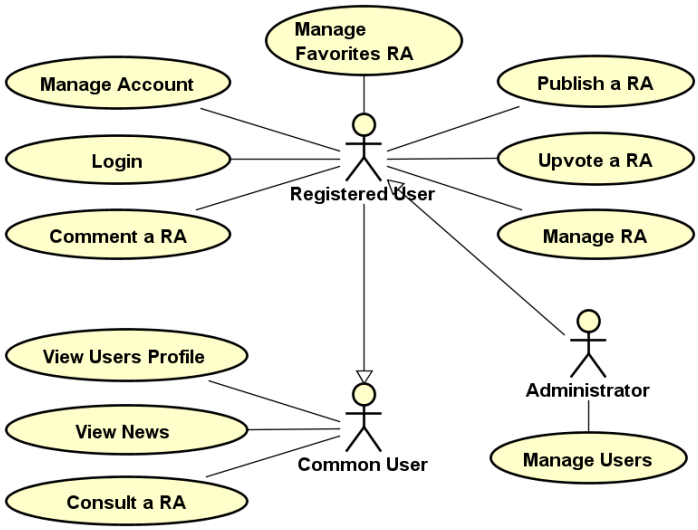

Fig. 3. Use case diagram of the reference architecture repository

in their architectural description.

IV. Development

This section details how this repository is developed. In particular, RARep[4] is implemented using current well known frameworks and technologies to the development of web systems, such as Java, MySQL[5], Hibernate[6], and Bootstrap[7].

A. Architectural Project

Using the conceptual model and use case diagram previously presented as baseline, we developed a software tool

[4]RaRep, http://www.labes.icmc.usp.br:9083/RARep/index.jsp

[5]MySQL, https://www.mysql.com/
[6]Hibernate, http://hibernate.org/
[7]Bootstrap, http://getbootstrap.com/

supporting reference architectures dissemination. These concepts were important for the development of a tool prototype that was used in the identification of additional features and refinement of these models. In particular, this tool standardizes the creation, presentation, and organization of reference architectures. Moreover, some of the features supported in RaRep are inspired in discussion forums and similar repositories, such as an up-vote feature and different search mechanisms. The registered user inherits the common user features and the administrator inherits the registered user features. The features related to the common users are: (i) view news, where the user can visualize news present in the repository; and (ii) consult reference architecture, where the user can see all details of a reference architecture available in the repository, such as the ones related to a reference architecture.

The features related to the registered users are: (i) login, which the user can authenticate in the repository; (ii) manage account: where the user can update his personal information; (iii) manage favorites reference architectures, which the user can add, remove or list his favorites reference architectures; (iv) up-vote a reference architecture, which the use can up-vote a reference architecture present in the system; (v) comment a reference architecture, which the user can post a comment in a reference architecture in the repository; and (vi) manage reference architectures, which the user can add, remove, and update its reference architectures in the repository. Administrators can also manage users, adding, removing, and updating every information in the repository.

B. Implementation

As mentioned before, software prototyping was used to mitigate uncertainties in the requirements. During the development process, the first part implemented was the front-end of the website. At this stage, all pages were developed as well as the transitions between them. The next step was the development of the back-end of the repository. At this step, we implemented all Hibernate XML for data persistence in the database. The conceptual model was used for mapping the classes of the repository. Then, we developed the update, insert, select and remove methods of these classes. After the development of the front-end and back-end, the next step was to integrate these two parts. The latest repository deployment has a total of 14 classes, around 5 kloc, and 22 hibernate XML files.

Similar software engineering tools helped us to come up with the features currently available in the reference architecture repository. The main activities of this repository are post and search for reference architectures. Aside from that, there are several features supporting the interaction among users of this repository, such as: (i) posting comments about a particular reference architecture; (ii) listing a particular reference architecture in its favorite list; (iii) evaluating a reference architecture by means of an up-vote system; and (iv) posting news about their own reference architectures. These features are available in a navigation bar at the top of the page. Apart from the search feature, the other features are only available to registered users of the system. We made

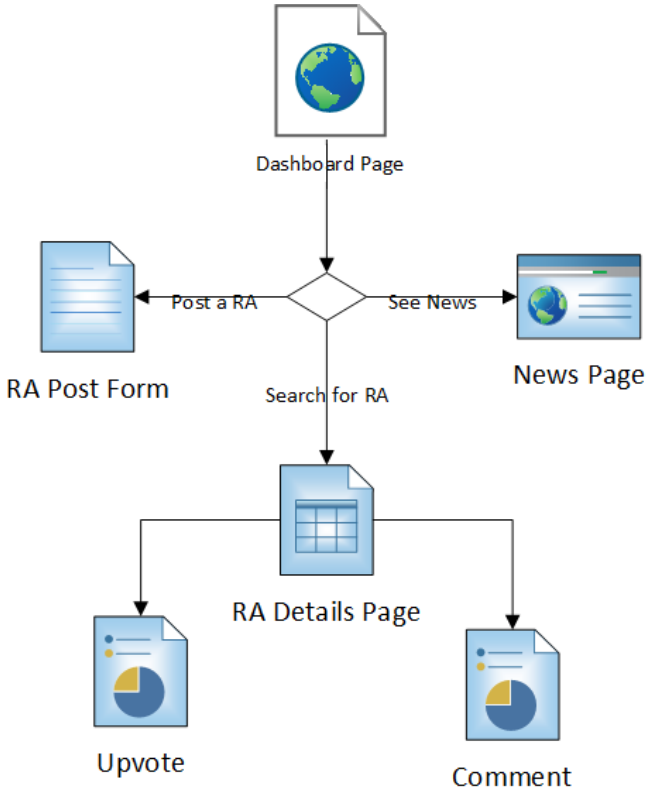

Fig. 4. Navigation flowchart for the repository

this decision because the main goal of this repository is to disseminate reference architectures to whoever access the repository. Figure 4 shows a navigation flowchart describing the different actions that an user can perform in this repository.

C. Illustrative Example

In this subsection we present and illustrative example using our repository to register an reference architecture. RefSORS (Reference Architecture for Service-Oriented Robotic Systems)[21] is a reference architecture for developing indoor, grounded mobile Service-Oriented Robotic Systems (SORS), i.e., robotic systems designed according to Service-Oriented Architecture (SOA). In order to post RefSORS, the first step is to login into the repository. In case the user is not registered into the repository, then the user has to fill a form with his account info such as full name, login, email, institutions he is affiliated and password. The next step is select the *post a reference architecture* feature at the navigation bar. At this point, the form presented in the Fig. 5 will be available to the user to complete the information of his reference architecture. For the RefSORS, the form can be filled in as:

Title: Reference Architecture for Service-Oriented Robotic Systems - RefSORS;
Authors: Lucas Bueno R. Oliveira, Elisa Yumi Nakagawa, and Flavio Oquendo;
Institutions: ICMC/USP (Brazil), UBS (France);
Date: Oct 10, 2014;
Domains: Robotic Systems, Embedded Systems;

Architectural styles: SOA;
Reference Models: OASIS Reference Model
Architectural Patterns: Layered Architecture;
Related Documents: SOA-RA technical standard, OASIS reference architecture, ArchSORS process and SoaML;

After posting a reference architecture in the repository, all registered users can discuss the architecture by posting comments. In addition, users can also upvote, favorite, and search for this architecture.

V. Future Activities

This work was motivated by the need of a supporting tool that promotes dissemination of reference architectures. This task is not simple given the diversity and amount of knowledge encompassed in reference architectures. As future activities, we plan to:

- **Improve reference architectures documentation:** increasing the amount of details and artifacts present in the representation of reference architectures without jeopardizing its clearance and easiness to identify its element. These new elements, for example, could be related to architecture decisions. As a consequence, instantiating the reference architecture into concrete software architectures could be resumed to the selection of architecture decisions.
- **Develop additional features:** Currently, our repository supports managing several artifacts related to reference architectures documentation and search for particular information regarding their design. However, we intend to develop additional features that could enhance users' experience with our tool. For instance, we intend to support the communication among different tools with RARep such as Research Gate[8], so users will be able to link account to RARep. We expect to identify other relevant features in a case study regarding the usability of this tool.
- **Treat variability of reference architectures:** The variability in reference architectures concerns the ability of a software artifact built from such architectures to be adapted for a specific context in a preplanned manner [9]. In this sense, documenting variability to architecture decisions and the reference architecture itself would help to create an even larger knowledge repository as all alternatives, dependencies, and options would be registered in the reference architecture.

VI. Conclusion

Reference architectures are key to knowledge reuse in software systems as they promote best practices. Despite their relevance, a mechanism supporting their open distribution to the public was still missing. The main contribution of this paper is to present the design and implementation of a web-based tool that supports the creation of a catalog of reference architectures. This web-based tool takes into account a reference model for reference architectures as well as best practices for software architecture description, including architectural viewpoints, styles, and concerns that have been considered in their design.

As future work, we intend to extend this web-based tools with additional features. As a consequence, we expect that this tool can contribute for further disseminating and facilitating the access of information related to reference architectures and, hence, promoting knowledge reuse.

[8]Research Gate, https://www.researchgate.net/

Acknowledgment

This work is supported by the Brazilian funding agency FAPESP, grants 2014/02244-7, 2014/25341-8, and 2012/24290-5.

References

[1] C. H. Alliance. *Continua Health Alliance*. On-line. Available at: http://www.continuaalliance.org/ (02/2016). 2013.
[2] S. Angelov, P. Grefen, and D. Greefhorst. "A Classification of Software Refererence Architectures: Analyzing Their Success and Effectiveness". In: *IEEE/IFIP Conference on Software Architecture (WICSA'09)* (2009), pp. 141–150.
[3] S. Angelov, P. Grefen, and D. Greefhorst. "A Framework for Analysis and Design of Software Reference Architectures". In: *Information and Software Technology* vol. 54 (2012), pp. 417–431.
[4] AUTOSAR. *AUTOSAR (AUTomotive Open System ARchitecture)*. On-line. Available at: http://www.autosar.org/ (Accessed 02/2016). 2013.
[5] P. Avgeriou, S. Retails, and M. Skordalakis. "An Architecture for Open Learning Management Systems." In: *Panhellenic Conference on Informatics (PCI'2003)* (2003), pp. 183–200.
[6] L. Bass, P. Clements, and R. Kazman. *Software Architecture in Practise*. Addison-Wesley, 2012.
[7] J. Bosch. "Software Architecture: the Next Step". In: *First European Workshop: Software Architecture (EWSA'04)* (2004), pp. 194–199.
[8] N. S. Eickelmann and D. J. Richardson. "An Evaluation of Software Test Enviroment Architectures". In: *International Conference on Software Engineering (ICSE'96)* (1996), pp. 353–364.
[9] M. Galster et al. "Variability in Software Architecture: Current Practice and Challenges". In: *SIGSOFT Software Engineering Notes* vol. 36 (2011), pp. 30–32.
[10] D. Garlan and D. Perry. "Introduction to the Special Issue on Software Architecture". In: *IEEE Transactions on Software Engineering* (1995), pp. 269–274.
[11] A. Grosskurth and M. W. Godfrey. "A Reference Architecture for Web Browsers". In: *IEEE International Conference on Software Maintenance (ICSM'05)* (2005), pp. 661–664.
[12] M. Guessi, L. B. R. Oliveira, and E. Y. Nakagawa. "Representation of Reference Architectures and Reference Models: A Systematic Review". In: *28th Int. Conference on Software Engineering and Knowledge Engineering (SEKE'11)* (2011), pp. 1–4.
[13] M. Hagele. *Project RoSta - Robot Standarts and reference architectures*. On-line. Available at: http://www.robot-standarts.org/ (Accessed 02/2016). 2013.
[14] ISO/IEC/IEEE. *ISO/IEC/IEEE 42010:2010 International Standard for Systems and Software Engineering – Architectural description*. 2011.
[15] P. Kruchten, H. Obbink, and J. Stafford. "The past, present, and future of software architecture". In: *IEEE Software* vol. 23. nº 2 (2006), pp. 22–30.
[16] G. Muller. *A Reference Architecture Primer*. On-line. Available at: http://www.gaudisite.nl/ReferenceArchitecturePrimerPaper.pdf (Accessed 02/2016). 2008.
[17] E. Y. Nakagawa, F. Oquendo, and M. Becker. "RAModel: A Reference Model of Reference Architectures". In: *IEEE/IFIP Conf. on Software Architecture & Eur. Conf. on Software Architecture (WICSA/ECSA'2012)* (2012), pp. 297–301.

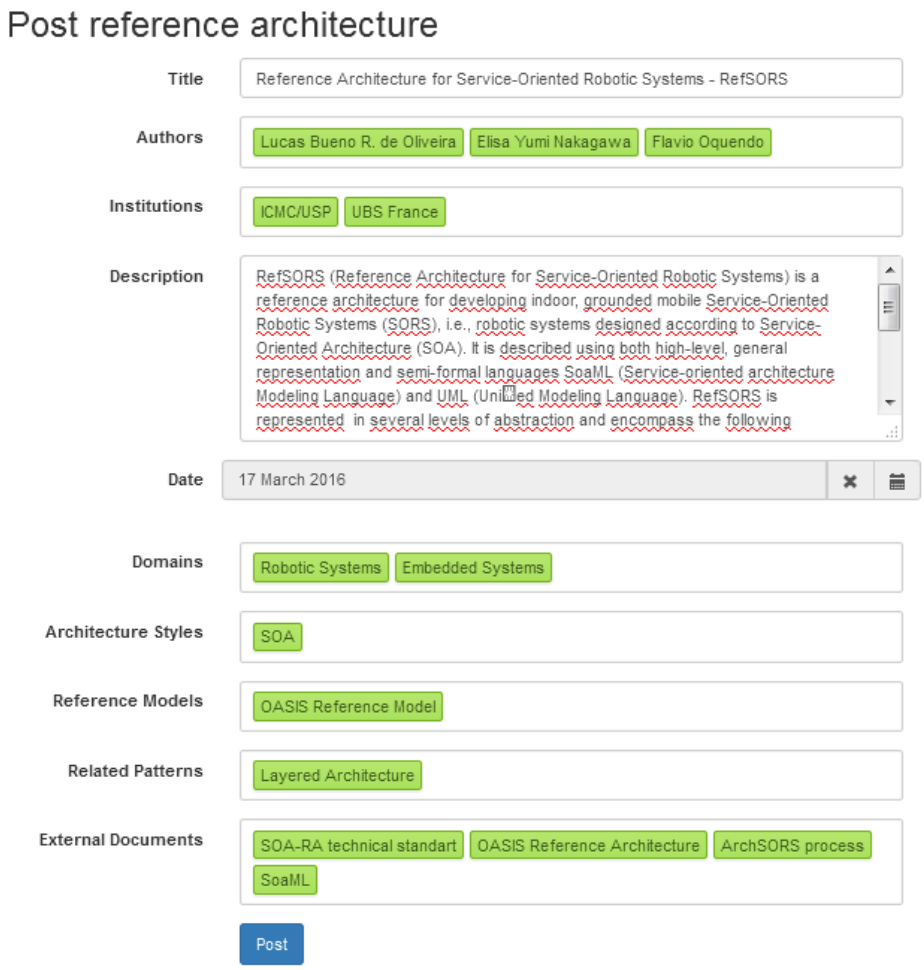

Fig. 5. Reference architecture post form

[18] E. Y. Nakagawa, F. Oquendo, and J. C. Maldonado. "Software Architecture: Principles, Techniques, and Tools". In: ed. by M. Oussalah. John Wiley & Sons, 2015. Chap. Reference Architectures, pp. 101–122.

[19] E. Y. Nakagawa et al. "Consolidating a Process for the Design, Representation, and Evaluation of Reference Architectures". In: *Working IEEE/IFIP Conference on Software Architecture (WICSA'14)* (2014), pp. 1–10.

[20] OASIS. *Reference Architecture Foundation for Service Oriented Architecture Version 1.0*. On-line. Available at: http://docs.oasis-open.org/soa-rm/soa-ra/v1.0/cs01/soa-ra-v1.0-cs01.html (Accessed 02/2016). 2013.

[21] L. B. R. Oliveira et al. "Towards a Process to Design Architectures of Service-Oriented Robotic Systems". In: *European Conference on Software Architecture (ECSA'14)* v. 8627 (2014), pp. 218–225.

[22] OMG. *Unified Modeling Language v. 2.4.1*. [On-line]. Available at: http://www.omg.org/spec/UML/2.4.1/ (Accessed 02/2016). 2011.

[23] U. Project. *The UniversAAL Reference Architecture*. On-line. Available at: http://www.universaal.org/ (Accessed 02/2016). 2013.

[24] K. Sandkuhl and B. Messer. "Towards Reference Architectures for Distributed Groupware Applications". In: *Euromicro Workshop on Parallel and Distributed Processing* (2000), pp. 135–141.

[25] M. Shaw and P. Clements. "The Golden Age of Software Architecture". In: *IEEE Software* 23.nº 2 (2006), pp. 31–39.

[26] A. I. Wasserman. "Towards a Discipline of Software Engineering". In: *IEEE Software* vol. 13.no. 6 (1996), pp. 22–31.