How Do Open Source Communities Document Software Architecture: An Exploratory Survey

Wei Ding1,4, Peng Liang1,2*, Antony Tang3, Hans van Vliet2, Mojtaba Shahin1,5
1 State Key Lab of Software Engineering, School of Computer, Wuhan University, China
2 Department of Computer Science, VU University Amsterdam, The Netherlands
3 Faculty of Science, Engineering and Technology, Swinburne University of Technology, Australia
4 Key Laboratory of Earthquake Geodesy, Institute of Seismology, China Earthquake Administration, China
5 Department of Computer Engineering, Neyriz Branch, Islamic Azad University, Iran

* Corresponding author

tingwhere@whu.edu.cn, liangp@whu.edu.cn, atang@swin.edu.au, hans@cs.vu.nl, mojtabashahin@gmail.com

Abstract—Software architecture (SA) documentation provides a blueprint of a software-intensive system for the communication between stakeholders about the high-level design of the system. In open source software (OSS) development, a lack of SA documentation may hinder the use and further development of OSS, but how much “architecture” documentation is enough and appropriate is largely dependent on the contextual factors of development. In order to understand the state of the practice of SA documentation in OSS projects, we conducted a documentation-based survey to explore how SA is documented in OSS projects. Out of 2,000 OSS projects from four major OSS sources, we found that 108 projects have some SA documentation, which shows that the SA documentation is scarce in OSS development. We analyzed these 108 projects to understand what SA information has been documented and how they have been described. We have found that frequently-documented architectural information is model, system, and mission; natural language is the most frequently-used architectural language for specifying architectural information in OSS SA documents. The results also show that the likelihood that an OSS project will document SA is increased when more developers are involved in the project, and industry and research OSS projects are more likely to create SA documents than freelance projects.

Keywords—open source software, software architecture, architecture documentation, architecture document elements

I. INTRODUCTION

Open source software (OSS) has received a lot of attention in software development in the past decade [1]. The source code of OSS is available to anyone for studying, modifying, and distributing under various open source licenses, and branch versions are developed to accommodate varying needs. OSS can be published under these licenses so that the whole OSS community can benefit [2]. The existence of OSS allows faster adoption of technology, increases innovation, and reduces costs and time-to-market [3]. The number of OSS projects and their developers has been on the rise, e.g., there are 324,000 projects hosted in sourceforge.com with 3.4 million developers as of March 2013. It is estimated that 95% of commercial software packages will include OSS as components by 2016 [4]. Most OSS projects are controlled by core developers and supported by large numbers of contributors [5]. The formation and enactment of OSS development processes is performed by loosely coordinated software developers and contributors, for instance, the majority of OSS projects typically has no management staff to centrally administrate the development processes [6], and OSS developers tend to code solutions without producing adequate documentation [7]. Consequently, the documentation of OSS projects is less emphasized in at least some OSS projects [8]. The information contained in those documents is presented in various forms such as Office files, web pages, and images [9].

A software architecture (SA) represents the structure or structures of a system, which comprises elements, their externally-visible behavior, and the relationships among them [10]. The existence of a SA is one of the characteristics of successful OSS projects that promote anarchic collaboration while at the same time preserving centralized control over the interfaces [11][12]. SA documentation is an essential and integrated part of architecture design process that helps architects identify and record the necessary design decisions [13]. It provides stakeholders (e.g., architect) the information they need in an accessible and non-ambiguous form to do their job [14]. When adopting an OSS, software documentation has a positive impact on the degree and cost of OSS adoption [15], for example, the information in software architecture documents (SAD)1 can help users of OSS to understand the transition from design problems to architecture solutions, the model of a system, its mission, and architectural requirements and so on. Some research has indicated that a lack of SAD in OSS projects may hinder the use and further development of OSS [16], but whether OSS developers really need SA documentation and how much “architecture” documentation is enough and appropriate [17] is largely dependent on the contextual factors of software development, such as development method, rate of change, size of project, and architecture stability [18]. Understanding the state of the practice on what and how SA is documented in existing OSS projects will provide opportunities for studying architectural knowledge management issues in OSS development. For

1 SAD is singular as well as plural based on the context in which the term is used.
example, what mechanisms OSS developers use for communicating architectural knowledge when SA documentation is not available.

To this end, we conducted this exploratory survey [19] by investigating 2,000 OSS projects from four major OSS sources (i.e., 500 projects from each source) and finally collected 108 projects that have SA documentation. By extracting and encoding the data from the investigated OSS projects and collected SAD, we present our survey results with some findings. We suggest several research directions based on the survey results and findings.

The rest of the paper is organized as follows: Section II introduces and discusses the related work to this survey. Section III describes the design of this survey. Results of the survey are presented in Section IV, with the survey findings discussed in Section V. Section VI discusses the threats to the validity. Section VII concludes this survey with further work directions.

II. RELATED WORK

To the best of our knowledge, there is a lack of understanding on the state of the practice of how SA is documented in existing OSS development. Related work has been done using survey on various aspects of OSS development and software documentation.

Some literature has investigated the use of SA documentation in OSS projects based on interviews with developers of OSS. For example, a study used grounded theory to find out how development teams manage and document SA in long-term software product evolution [20]. The results reported in [20] indicate that a lack of up-to-date SAD and missing architectural information are problematic, especially for the evolution of software products. Only two of the eight interviewees in this study came from OSS projects, so the conclusions cannot be generalized to other OSS projects. Ajila and Wu interviewed 18 OSS managers and developers to investigate the effects of OSS adoption on software development economics [15]. Their work reveals that documentation, used as a criterion of OSS adoption, has a statistically positive correlation with the rate of OSS adoption. The data collection and results of these studies are from and based on OSS developers, which is different from our survey, that is based on SAD.

The importance of SA documentation in both OSS and industrial software development has also been investigated through literature study and survey. Stol et al. used a systematic literature review method to identify the challenges in integrating OSS and inner source software based on existing literature [21]. Their work reveals that lack of (or low quality of) documentation and architecture information sharing are two challenges in OSS development, which provides the motivation for our survey on understanding how SA is documented in OSS development. de Silva and Balasubramaniam surveyed and further categorized the approaches that have been proposed to control architecture erosion [22], in which SA documentation is one of the approaches. Their survey results present how various aspects (e.g., documentation models, architecture description language (ADL)) of SA documentation are practiced to enhance the effectiveness of controlling architecture erosion, which can be compared with our survey results on SA documentation in an OSS development context.

Other surveys investigate documentation in software development, which is not specifically related to SA documentation of OSS projects. For instance, Lethbridge et al. conducted three studies (including a survey) to elucidate the patterns by which software engineers use and update software documentation [23]. The work reported in [23] shows that software documentation is often out of date, poorly written, and time consuming to create, which also holds for SA documentation [24]. Tang et al. surveyed SA practitioners to get an understanding on architects’ perceptions on the use and documentation of architecture design rationale [25], which is an important part of SA documentation. They identified the barriers to use and document architecture design rationale by practitioners, such as no time/budget, no standards, etc.

III. SURVEY DESIGN

This survey is designed for collecting SAD of OSS projects. The SAD investigated in this survey are collected from four major OSS sources. In this section, we describe the design of this survey, including the selection criteria of SAD (Section A), the research questions to be answered in this survey (Section B), and the survey process (Section C).

A. Selection Criteria of SAD

The essence of SA documentation is writing down and keeping current the results of architectural decisions to keep stakeholders informed [14]. There is no consistent understanding about what SA documentation should comprise of and what architectural information should be included in a SAD [26]. In order to have a clear and definite scope on the topic of this survey: **SA documentation** in OSS projects, we adopt the conceptual framework for describing SA defined in IEEE 1471-2000 standard “recommended practice for architectural description of software intensive systems” [27], as a reference selection criterion to SAD included in this survey. **IEEE 1471-2000 standard** prescribes how a SAD should be documented and what architecture document elements should ideally be included in a SAD. The IEEE 1471-2000 standard [27] prescribes structural elements as an indispensable part and design rationale and principles as an optional part of a SAD.

We defined the following inclusion and exclusion criteria to select SAD for this survey:

**Inclusion criteria**

11: The document is created in an OSS project.

12: The document should contain at least one of the architecture elements (except System, Mission, and Environment elements, see Section IV.B) as specified in IEEE 1471-2000 standard, the conceptual framework for

---

2 The reason that cited IEEE 1471-2000 in this survey instead of ISO 42010:2011, the successor of IEEE 1471, is that the new standard ISO 42010:2011 was published after many OSS projects had started.
describing SA [27], e.g., architecture design elements, the externally visible properties of those elements, and the relationships among them. The reason we ignore the three architecture elements (System, Mission, and Environment) is that they are the elements which are not unique to a SAD, for example System can also be described in a requirements specification. The architecture elements were identified by the first author according to their descriptions in IEEE 1471-2000 [27] (see Section IV.B), and verified with the second author.

Exclusion criteria

E1: If the document only introduces how to use design elements without describing their relationships, e.g., a user guide document of application programming interfaces (API) or user interface (UI), the document is excluded.

E2: Any document that is not written in English is excluded.

E3: A document which has an updated version is excluded if the updated version has been included, in order to prevent from analyzing duplicated document information.

B. Research Questions

Survey research is used to identify the characteristics of a broad population of individuals [19]. This survey is an exploratory survey to explore facts, seek new insights, and to generate ideas and hypotheses for new research [28][29]. The “population” of our survey is SAD in OSS projects. As suggested in [19], research questions (RQ) are a precondition for conducting a survey to identify a representative subset and determine appropriate sampling technique. Three RQs of this survey are described as follows:

**RQ1.** What architectural information has been documented in SAD of OSS projects?

**Rationale:** Selective architecture information (e.g., stakeholders, concerns, architecture views) is documented according to the needs of the OSS project. In this survey, we try to understand what architecture information is documented in SAD of OSS projects, covering any formats for archiving architectural information within the OSS repositories, e.g., files, web pages. We use the conceptual model for architecture description in IEEE standard 1471-2000 [27] as a benchmark model to identify the documented architectural information, i.e., architecture document elements, in SAD of OSS projects.

**RQ2.** What architectural languages are used in SAD of OSS projects?

**Rationale:** Architectural information documented in SAD is specified using various languages (e.g., UML, formal ADL, or both e.g., [30]). Understanding the way architectural information in SAD of OSS projects is documented can help researchers and practitioners to identify mainstream architectural languages for OSS projects as well as their needs [31].

**RQ3.** How architectural languages are used in specifying architectural information?

**Rationale:** This RQ is a further analysis based on the results of RQ1 and RQ2. Various architectural languages are used to describe specific architectural information. For example, a UML model is mostly used to describe architecture design results in the model architecture element of IEEE 1471-2000. The results of this RQ will be helpful to understand which architectural languages are mostly used to describe specific architectural information in OSS SAD.

C. Survey Process

This survey is conducted in three steps. First, we select OSS sources to search SAD in OSS projects. Second, SAD of OSS projects are retrieved from target OSS sources selected in the first step. Third, quantitative data of SAD in OSS projects are extracted to answer the research questions. The survey method is further elaborated below.

To be comprehensive and representative, four major OSS sources are selected: Sourceforge, Github, Google Code, and Tigris as listed in TABLE I, in which Sourceforge, Github, Google Code are the top three open source sites and Tigris focuses on building software development tools [32]. Note that some OSS sources are excluded in this survey, for two reasons: (1) The OSS sources are restricted to specific development languages or platforms, which may make the survey results limited and ungeneralizable. For example, Eclipse Labs provides a repository of OSS projects developed in Eclipse platform; Ruby Forge hosts the OSS projects implemented in the Ruby language. (2) Some OSS sources provide limited access to the OSS projects hosted. For instance, BitBucket only provides free access to OSS projects with few users (<5).

**TABLE I. OSS SOURCES INCLUDED IN THIS SURVEY**

<table>
<thead>
<tr>
<th>#</th>
<th>OSS Source</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Sourceforge</td>
<td><a href="http://www.sourceforge.com/">http://www.sourceforge.com/</a></td>
</tr>
<tr>
<td>2</td>
<td>Github</td>
<td><a href="http://www.github.com/">http://www.github.com/</a></td>
</tr>
<tr>
<td>3</td>
<td>Google Code</td>
<td><a href="http://code.google.com/">http://code.google.com/</a></td>
</tr>
<tr>
<td>4</td>
<td>Tigris</td>
<td><a href="http://www.tigris.org/">http://www.tigris.org/</a></td>
</tr>
</tbody>
</table>

The search processes in various OSS sources are different, depending on the organization of and the search functions provided by each OSS source. For example, the OSS projects hosted at Sourceforge are classified into 10 categories based on their application domains. The projects in each category are sorted according to their download counts. We checked OSS projects about whether they have SAD by manually-browsing the top 500 projects in Sourceforge (i.e., the top 50 projects in each of the 10 categories). In this survey we regard the architectural information archived within the OSS sources (repositories) as SA documentation if it contains architecture description or diagrams (e.g., UML) that visualize architecture models. The SA documentation can be captured in specifications, design documents in files or web pages. The architectural information recorded and scattered in resources outside the OSS repositories, e.g., project forums, mailing lists, developer blogs, and social media, is not considered SA documentation in this survey. We checked the same number (i.e., 500) of OSS projects in the remaining three OSS sources. Note that the total number of OSS projects hosted at Tigris is less than that of the others [32], i.e., only 795 projects are

---

3 http://www.eclipselabs.org/
4 http://rubyforge.org/
5 http://bitbucket.org/
hosted at Tigris (till December 2013), and we manually checked the first 500 projects sorted by project name (since the download count information of the projects is not available) in Tigris to verify whether they have SAD. We present the survey results and discuss the survey findings in Section IV and Section V respectively.

IV. SURVEY RESULTS

A. Overview of Survey Results

a) Adoption of SAD in OSS projects

108 OSS projects that have SA documentation were selected from the four OSS sources listed in TABLE I. Fig. 1 shows the numbers of investigated OSS projects in each OSS source that have at least one SAD. For example, there are 45 OSS projects that have SA documentation in the investigated 500 projects from Sourceforge. Overall, the percentage of OSS projects that have SA documentation is 5.4%. This result implies that SA documentation (1 out of 20 OSS projects) has not been widely employed, and consequently SAD have not been widely created and used in OSS development.

![Fig. 1. Percentage and number of OSS projects that have SA documentation in four major OSS sources](image)

b) Project types and SAD

The data about the background of OSS projects and the number of developers who participate in OSS projects are extracted in this survey. OSS projects can be classified into three categories according to their project description.

Industry project: the projects that are developed by industrial organizations or commercially sponsored by industrial communities [33] (e.g., Liferay Portal project\(^8\)).

Research project: the projects that are developed by research organizations (research institutes or universities) or supported by research fund (e.g., ArgoPrint project\(^8\)).

Freelance project: the projects that are developed by freelance developers, as well as the projects that didn’t explicitly mention their development background (e.g., Joosy project\(^8\)).

Fig. 2 shows the distribution of OSS projects that have SA documentation over three project types. We can find that industry and research OSS projects are more likely to create SAD in their development (i.e., 11.7% industry OSS projects and 11.5% research OSS projects have SA documentation). Freelance OSS projects pay less attention to create SAD (i.e., only 3.8%, 60 out of 1,588 freelance OSS projects have SA documentation).

![Fig. 2. Distribution, percentage, and number of OSS projects that have SA documentation over project types](image)

c) Number of developers and SAD

Fig. 3 shows the distribution of OSS projects that have SA documentation over number of developers. According to the result of this survey, only 3.5% OSS projects developed by 1-2 developers have SAD and more (10.8%) OSS projects developed by 3-10 developers have SAD. When OSS projects are developed by more than 10 developers, the percentage of OSS projects that has SA documentation increases to 13.0%.

![Fig. 3. Distribution, percentage, and number of OSS projects that have SA documentation over number of developers](image)

d) Format of SAD

SA documentation is one of the means to communicate architectural knowledge between project stakeholders, who mostly work in a distributed environment in OSS development. Fig. 4 shows the distribution of OSS projects that have SA documentation over formats of SAD. It is clear from Fig. 4 that Html is the main format used for documenting SA in OSS development (i.e., 70.4% SAD use Html). A picture format, such as JPEG and PNG, is also frequently employed by OSS projects to document SA design (i.e., 20.4% SAD use pictures). Few OSS SAD use PDF and WORD (i.e., 5.6%)

\(^{8}\) http://sourceforge.net/projects/lportal/
\(^{8}\) http://argoprint.tigris.org/
\(^{8}\) https://github.com/joosy/joosy/
using PDF and 3.7% using WORD), although developers claim that these two formats are widely used to document SA in industrial software development [34]. It is one of the differences on SA documentation between OSS and industrial software development (more discussion about the differences and similarities in SA documentation between OSS and industrial software development can be found in the last paragraph of Section V). Note that, the sum of the numbers of OSS projects shown in Fig. 4 (119) exceeds the number of OSS projects that have SA documentation (108), because one OSS project may employ several formats (e.g., both HTML and picture) to document SA.

In the following subsections, we report the survey results to answer the research questions described in Section III.B. TABLE II shows the relationship between the aspects of the survey results and the respective RQ they try to answer. The detailed survey results are elaborated below in each subsection to answer the respective RQ.

**TABLE II. RELATIONSHIP BETWEEN RESEARCH QUESTIONS AND SURVEY RESULTS**

<table>
<thead>
<tr>
<th>Aspects of survey results</th>
<th>RQ</th>
</tr>
</thead>
<tbody>
<tr>
<td>Architectural document elements in SAD of OSS projects</td>
<td>RQ1</td>
</tr>
<tr>
<td>Architectural languages used in SAD of OSS projects</td>
<td>RQ2</td>
</tr>
<tr>
<td>Relationship between architectural languages and architectural document elements</td>
<td>RQ3</td>
</tr>
</tbody>
</table>

**B. Architecture Document Elements in SAD**

This subsection provides answers to RQ1: what architectural information has been documented in SAD of OSS projects? Using the conceptual model of architectural description proposed in IEEE 1471-2000 standard as a benchmark model [27], we identify various architecture elements documented in a SAD. IEEE 1471-2000 standard suggests ten main architecture elements: **system, mission, environment, stakeholder, concern, model, rationale, view, viewpoint, and library viewpoint**, which are elaborated below:

**System**: A collection of components organized to accomplish a specific function or set of functions [27].

**Mission**: a use or operation for which a system is intended by one or more stakeholders to meet some set of objectives [27].

**Environment** determines the setting and circumstances of developmental, operational, political, and other influences upon that system [27].

**Stakeholder**: An individual, team, organization who has an interest in a system [27].

**Concern**: Interest which pertains to the system’s development and is related to one or more stakeholders [27].

**Model** is developed using the methods established by its associated architectural viewpoint [27], e.g., in UML models [35].

**Rationale** records explanation, justification, or reasoning about architectural decisions that have been made in architecture design [27].

**View**: A representation of a whole system from the perspective of a related set of concerns [27].

**Viewpoint**: A specification of the conventions for constructing and using a view [27].

**Library viewpoint**: A viewpoint that is defined in other literature and can be directly used (referred) [27], e.g., the 4+1 view model [36].

As described in inclusion criteria I2 in Section III.A, the architecture document elements were initially identified by the first author and further verified by the second author based on their descriptions above, and any disagreements were resolved.
through discussion between the two authors in order to mitigate the identification bias by one researcher. Fig. 6 shows what architecture document elements (i.e., architectural information) were identified in the SAD of the collected OSS projects that have SA documentation. The three most documented types of architectural information are: model (98.1%, 106 out of 108 projects), system (97.2%, 105 out of 108 projects), and mission (91.7%, 99 out of 108 projects). No SAD refers to viewpoint or library viewpoint, consequently, very few SAD document architecture views (7.4%, 8 out of 108 projects). Other architecture document elements receive considerable attention in OSS SAD, e.g., stakeholder (41.7%, 45 out of 108 projects) and concern (33.3%, 36 out of 108 projects).

C. Architectural Languages Used in SAD

The categorization of architectural languages for describing architectural information are adopted from [34]. Fig. 7 shows the distribution of OSS projects that have SA documentation over four architectural languages, which are elaborated below:

Natural language: Architectural information is described with informal specification written in natural language [37], e.g., using technical or domain terms to interpret architecture model elements or views.

Diagram: Architectural information is illustrated with informal diagrams [34], e.g., architectural models created by Microsoft Visio or PowerPoint. UML is intentionally excluded in this architectural language because UML is a standard notation for architecture description.

UML: Architectural information is specified with UML models [35], a notation-based standard language for architecture description, e.g., using UML activity diagram to describe the process view in SA design.

ADL: Architecture description languages (ADL) provide formally-specified modeling notations and constrains to describe architectural information, for example, using C2 to formally specify components and connectors [38].

It is clear from Fig. 7 that natural language plays a dominant role in describing architectural information, i.e., 88.9% (96 out of 108) OSS projects use natural language in their SAD. ADL is not used in any OSS projects to describe and document SA design. Diagram is also widely used (41.7%, 45 out of 108) in SAD of OSS projects to present architectural information. UML, as a standard modeling language, is used in 17.6% (19 out of 108) OSS projects. Note that, the sum of the numbers of OSS projects shown in Fig. 7 (160) exceeds the number of OSS projects that have SA documentation (108), because one SAD may employ several architectural languages, e.g., using both UML and natural language to describe an architecture design.

D. Relationship between Architectural Languages and Architecture Document Elements

Fig. 8 presents the distribution of OSS projects that have SA documentation in two dimensions: architectural languages and documented architecture elements in SAD. The number in a bubble represents the number of OSS projects that use certain architectural language to describe specific architecture information (i.e., architecture document elements). For example, the SAD of the Data Crow project [39], which develops an OSS tool for cataloging and organizing media collections, uses UML class diagram to represent the system architecture element. Note that the sum of all the bubbles (i.e., the number of OSS projects) in a certain row may be greater than the total number of OSS projects that have SA documentation (108), because the SAD in an OSS project may contain several architecture document elements that are described using a certain architectural language. The sum of all the bubbles in a certain column, e.g., 158 in the column “model”, likewise may exceed the total number of OSS projects that have SA documentation (108), because the SAD in an OSS project may employ various architectural languages (e.g., UML and natural language) to describe “model” architecture elements.

From Fig. 8, we can get the same answer as presented in Section IV.B: architectural languages are mainly used to describe model, system, and mission architectural elements in OSS SAD. Moreover, natural language as a simple and easily-understandable language is used to describe all the architecture document elements in SAD except viewpoint and library viewpoint, which are not documented in the OSS SAD in this survey.
which are directly helpful for the adaptation of OSS, and **model**, **system**, and **mission** description in SAD provides such help for developers to get a quick overview of the system, so that they can decide how to adapt it to different customer needs or deployment environments. Second, the OSS development provides limited incentives for developers to document SA design, and important knowledge on architecture design is mentally distributed over developers [43]. Third, some architecture document elements may not be included in SAD of OSS projects (e.g., **concerns**, **stakeholders**, and **system**-of-interest), but rather in other documents, such as user guide or system introduction.

**Relationship between documenting SA in OSS projects and agile software development:** Agile development methods are widely employed in current software development practices, which represents a major departure from traditional, plan-based approaches to software development [44]. OSS development is an ideal case that is suitable to employ agile development by which a distributed development team can often create high quality software [45]. However, creating a comprehensive SAD in OSS projects is time-consuming and difficult because of distribution of design knowledge over developers as well as the tension between agile development and software documentation [46][47]. From Fig. 5 in Section IV.A, we can find that the architectural information of 26.9% OSS projects is documented in a single document. From Fig. 7 in Section IV.C, we see that **natural language** and **diagram** (e.g., boxology), the two simplest architectural languages, are used more often in documenting SA design compared to UML and ADL. In order to develop OSS in an agile way, developers of OSS tend to use the architectural languages as simple as possible, which is beneficial to both the creators and users of SAD in a distributed context. For example, natural language is frequently-used to describe **model** elements in SAD according to the result shown in Fig. 8. Meanwhile, according to the result shown in Fig. 3, the likelihood that an OSS project will document SA is increased when more developers are involved in the project. This result suggests that SA documentation is deemed useful to coordinate project development independent of what development methods (e.g., agile) are employed.

**Relationship between architectural languages and documented architecture elements:** As shown in Fig. 8, various architectural languages are used to specify different architectural elements. Natural language is used to describe all architectural elements, the reason is that natural language is ideal and commonly-used for human communication in software development [37]. Diagram and UML are often used to describe **model** and **system** elements since these two architectural languages are graphical representations suitable for representing the structural aspect of an architecture (i.e., **model** and **system** elements). They can represent and facilitate interaction with non-technical stakeholders [31], e.g., UML can be used to describe and communicate the static structure of an architecture [35]. However, diagram and UML are not used to describe **mission** and **environment** elements. It is not
surprising since these two elements are often described in natural language.

**Differences and similarities on SA documentation between OSS and industrial software development:** (1) SA documentation is mostly archived in WORD or PDF formats in industrial software development (87% developers using WORD or PDF to document SA) [34], while these two formats are less used to document SA in OSS development (5.6% projects using PDF and 3.7% using WORD) according to the result presented in Section IV.A. Html is the most popular format to document SA in OSS development, which is to be expected since OSS development is largely coordinated in a web environment, and Html format is ready-to-use for documentation in such a context. (2) Combination of architecture views, viewpoints, and mappings among views has been popularly used to capture and describe different aspects of an architecture [22] (e.g., the 4+1 view model [36] and views and beyond [14]), while SAD in OSS development pay less attention on employing views (7.4%) and viewpoints (0.0%) to document SA as the result presented in Section IV.B. This difference shows that the SA in OSS development is less commonly described and documented in architectural views compared to the SA description and documentation in industrial software development. For example, in a recent survey [31], about 85% of architects from industry use multiple views for architectural description, and 59% of them use mechanisms to ensure cross-view consistency. (3) ADL is rarely used to describe and document SA in industrial software development (5% SAD using ADL to document SA) [34] and has never been used in OSS development (0.0%) according to our survey result presented in Section IV.C. This finding is consistent with another industrial survey [31] suggesting that there is a gap between the large number of ADLs produced by academia and the ones that reach industrial adoption. (4) Diagrams, UML, and natural language are decently popular architectural languages wanted in industry (19.7%, 16.2%, and 14.5% respectively) [34], while natural language is dominant (88.9%) in describing architecture of OSS projects, and diagram (41.7%) and UML (17.6%) are also well used as shown in Fig. 7.

VI. **THREATS TO VALIDITY**

The validity of the results and findings of this survey may be influenced by e.g., the definition and scope of SAD, bias on SAD selection, and other factors. The relevant threats to the validity of the survey results are discussed in this section according to the guidelines in [48]:

**Construct validity** in a survey focuses on whether the survey constructs are defined and interpreted correctly. The main construct in this survey is the definition and scope of SAD in OSS projects. To mitigate the bias on SAD definition, we chose the architectural description model in IEEE 1471-2000 standard as a base model to select SAD in OSS projects and identify architecture document elements. About the scope of SAD covered in this survey, we explicitly limited the SAD that document architectural information within the OSS repositories, e.g., files, web pages. The architectural information recorded and scattered in resources outside OSS repositories, e.g., project forums, mailing lists, developer blogs, and social media, was not included in this survey.

**Internal validity** focuses on the unknown factors that may have an influence on the survey results. This threat in this survey is minimized because we employed a descriptive statistical method, but not data analysis to present the results, in other words, we did not intend to establish any causal relationship between the OSS projects that have SA documentation and the characteristics of these OSS projects in this survey.

**External validity** refers to the degree to which our findings from this survey can be generalized. It is impossible to investigate all OSS projects in all OSS sources. To mitigate this threat, we chose four popular and representative OSS sources and include 500 OSS projects from different categories (application domains) in each source to ensure that the survey findings can be generalized.

**Reliability** focuses on whether the survey yields the same results if other researchers replicate it. There is a risk that the outcome of this survey might be affected by the interpretation of the SAD selection criteria by different researchers. To mitigate this threat, we designed and carefully discussed the selection criteria of SAD with a pilot SAD selection to minimize the bias of researchers.

VII. **CONCLUSIONS AND FUTURE WORK**

This survey presents the results of an investigation of SA documentation practice in OSS development. We manually checked 2,000 OSS projects from four major OSS sources and collected 108 projects that have a certain form of SA documentation from those OSS projects. Our main findings indicate that (i) SA documentation has not been widely used in OSS development. The likelihood that an OSS project will document SA is increased when more developers are involved in the project, which suggests that SA documentation appears to be more useful to OSS projects when there are more developers. Participants in these projects may use the SA to align architecture design ideas between many developers. (ii) Industry and research OSS projects are more likely to create SA documents than freelance projects in OSS development. This phenomenon may be a result of the exposures to training and corporate standards. (iii) SAD in OSS projects focus on certain architecture document elements. Frequently documented architectural elements in OSS projects are: **model, system, and mission**. This information provides developers a quick overview of the system. This finding is similar to the SA documentation practice in industry [34], in which architectural information about components, interfaces, relationships (i.e., **models**) and a big picture (part of **system** and **mission**) is mostly needed to support development tasks. (iv) Frequently-used architectural languages in SA documentation of OSS projects are **natural language** and **diagram**, which are suitable, and probably simple to create, for architecture description and communication in an agile development. Formal language such as ADL was not used at all, whilst UML, as a semi-formal language, was used sparingly (17.6%).
This is a reflection on how developers capture and communicate their architectural knowledge using convenient means.

These results show that OSS projects do not have much SA documentation in general, and most of the SA documentation is done informally with some diagrams and natural language. These results contradict to the advocacy of traditional software engineering practice in which capturing knowledge for communicating requirements and architecture design is essential; it is also different to the agile development practice where daily face-to-face scrum meeting is used to communicate knowledge. OSS developers work collaboratively and yet remotely from each other, how do they cope with the lack of SA documentation in general is an interesting question that we want to explore further.

With this contradiction, we need to examine our positions in how architectural knowledge is represented and used: (1) Analyze and quantify the quality attributes of SAD in OSS projects (e.g., using the quality attributes of SAD identified in [50]), which can provide a basis to investigate the correlation between quality of SAD and characteristics of OSS projects (e.g., quality of OSS, efficiency of OSS maintenance), and for further quantitative measurement of the costs and benefits of using SAD in OSS projects. Part of this investigation will involve understanding the needs of OSS developers about SAD. (2) Obtain architectural information in other documentation sources, for example, OSS project blogs [51], mailing lists [52], social media [41], forums, code comments, and commit data [53], which are not covered and analyzed in this survey. We plan to further investigate these related architectural information sources on how architectural knowledge is shared between stakeholders and employed in OSS development (e.g., using various knowledge-based approaches [54]) with these sources, and how these sources can be collected to enrich OSS SA documentation.

ACKNOWLEDGMENT

This work has been sponsored by the Natural Science Foundation of China (NSFC) under the Grant No.61170025, KeSRAD: Knowledge-enabled Requirements to Architecture Documentation, and the Dutch Regeling Kenniswerkers project KWR09164, Stephenson: Architecture Knowledge Sharing Practices in Software Product Lines for Print Systems. We would also like to thank all the reviewers and chairs for their constructive comments.

REFERENCES


