Fraunhofer
IESE

FRAUNHOFER INSTITUTE FOR EXPERIMENTAL SOFTWARE ENGINEERING IESE

ARCHITECTURE DOCUMENTATION FOR DEVELOPERS: A SURVEY

Dominik Rost
Matthias Naab
Crescencio Lima
Christina von Flach Chavez

Fraunhofer IESE is an institute of the Fraunhofer-Gesellschaft.
The institute transfers innovative software development techniques, methods and tools into industrial practice, assists companies in building software competencies customized to their needs, and helps them to establish a competitive market position.

Fraunhofer IESE is directed by
Prof. Dr. Dieter Rombach
(Executive Director)
Prof. Dr.-Ing. Peter Liggesmeyer
(Scientific Director)
Fraunhofer-Platz 1
67663 Kaiserslautern
Germany

Executive Summary

Software architecture has become an established discipline in industry and documentation is the key for its efficient and effective usage. However, many companies do not have any architecture documentation in place or, if they do, the available documentation is not perceived as adequate by developers to support them in their tasks. To complement our experiences from projects with industry customers and as a foundation for the improvement of methods and tools in architecture documentation, we conducted a survey among 147 industrial participants, investigating their current problems and wishes for the future. Participants from different countries in Europe, Asia, North and South America shared their experiences. This report presents the results of the survey.

We identified five main findings. The results confirm the common belief that architecture documentation is most frequently outdated, updated only with strong delays, and inconsistent in detail and form and backed it up with data. Further, developers perceive difficulties with a "one-size-fits-all" architecture documentation, which does not adequately provide information for their specific tasks and contexts. Developers seek more interactive ways of working with architecture documentation that allow them to find needed information more easily with extended navigation and search possibilities. And finally, what developers perceive as relevant in terms of architecture information gives, in our opinion, a very complete and mature picture of what software architecture and its documentation should consist of.

Based on these results, we discuss directions for further research and the development of advanced methods and tools for architecture documentation in this report. Centralization with powerful tooling and automated generation mechanisms are possible means for addressing the problems of outdated and unspecific architecture documentation. Additionally, a clear, easy-to-follow connection between architecture an code is a major concern of developers for which new techniques have to be created. To achieve increased uniformity, organizations should invest in internal standardization in terms of form, details, and terminology, as well as in establishing a single source of architecture documentation to avoid inefficiency due to scattered information. We understand that static architecture documents must be replaced by new and interactive ways to convey the information that allow easy searching and navigation. And finally, to increase readability and understandability, we see an additional need for standardization and clarity through reduction of information.

Keywords: Software architecture, documentation, developers, industry, survey

Table of Contents

1 Introduction

1.1 The Practical Problem

Software architecture is accepted as an integral part of software engineering and as an enabler for efficient and effective software development. Increasing system size and complexity, as well as the employment of multiple, globally distributed development teams pose new challenges and increase the importance of documenting software architecture.

Nevertheless, in many projects with industry, we experienced industrial organizations that face these challenges but still *do not have any architecture documentation in place*. One major reason for this is that the creation of architecture documentation is cost-intensive. As a consequence, their software development suffers from growing communication and alignment effort, which makes implementation increasingly inefficient, inconsistent, and incompliant with the architecture. During system evolution, this leads to architecture erosion [20], which can prevent the achievement of essential system qualities and leads to decreasing maintainability. This holds true even for the initial development of the system [14].

However, we also observe organizations that *do have architecture documentation but are not able to leverage the potential it offers*. The reasons for this are diverse. For instance, the information provided in architecture documentation is often too unspecific for a concrete usage or task. Software architects create models and documents when they design the system and provide these as a single piece of comprehensive architecture documentation to all software developers. The developers then have to understand the complete architecture documentation in order to extract the information that is relevant for the local scope of their task and adapt it to their context. Another problem is inconsistencies in content and form, making it a challenge to find and understand relevant architecture information. Very often, the effectiveness of architecture documentation decreases over time because it is not kept up to date. The cost-benefit ratio of such architecture documentation may be a reason for an organization to decide to stop investing in architecture in general.

These challenges should be addressed by applied research on architecture documentation. Enhanced methods and tools shall support architects in creating and maintaining architecture documentation that allows highly efficient and effective implementation for software developers. To collect empirical facts for backing up our project experiences and as a foundation for improving methods and tools, we conducted a survey with developers in industry and asked them about their work with architecture documentation. In total, 147 developers from different countries in Europe, Asia, North and South America participated, working in organizations from two to more than 100,000 employees. In this report, we report on the creation and the results of the study and discuss our main findings.

1.2 This Study

We focused on software architecture documentation for developers to complement our experiences from industry projects with the views of software developers. By doing this, we aim to create a basis for future improvement of methods and tools for architecture documentation to make implementation more efficient and effective. Therefore, we defined the overall goal of the study according to the GQM template [3] as:

"Characterizing the current situation and improvement potential of software architecture documentation with respect to architectural information and its representation from the perspective of developers in industry as the basis for developing practically applicable methods and tools to make implementation work more efficient and effective."

There are some aspects that need to be emphasized in this goal statement: Our main focus is on software developers. While methods and tools might target architects in the creation of documentation, in this study we asked developers about their view as users of the documentation. A second aspect is that we distinguish two dimensions: (1) architecture information vs. architecture representation and (2) characterization of the current situation vs. requirements for the future. The combination of the two dimensions gives us four areas in which we asked developers about their views. And the last aspect is that this study constitutes the basis for the improvement of methods and tools for advanced architecture documentation that shall help developers to fulfill their implementation tasks in less time, with high-quality results.

From the goal statement following the two dimensions, we derived four research questions, which are the source of the structure and content of the survey:

- *RQ1: Which architectural information do developers currently receive for implementation activities and which problems do they perceive?*
- *RQ2: Which representation of architectural information do developers currently receive for implementation activities and which problems do they perceive?*
- *RQ3: Which architectural information would developers like to get for their implementation activities?*

- *RQ4: Which representation of architectural information would developers like to get for their implementation activities?*

1.3 Intended Audience

This report on our study has several main target groups:

- Architects in industry can compare the state and practices concerning architecture documentation in their organizations to those of other companies. This will allow them to identify strength and weaknesses and get ideas for improving the capabilities in their own organization.
- Researchers in software architecture can get directions for future applied research in the field of architecture documentation. They will find ideas for the development of enhanced methods and tools for software architecture documentation for industrial organizations.
- Participants of the study can compare their answers to those of other organizations. They can get an idea of what other developers perceive as important in architecture documentation for their implementation work and can initiate improvement processes based on this, for example.

1.4 Report Outline

The remainder of this report is structured as follows: In Section 2, we introduce fundamental work on architecture documentation. In Section 3, we describe our research methodology. In Section 4, we describe the results of our study; the main findings are presented Section 4.2. In Section 5, the main findings are discussed and the report is concluded. Section 6 presents acknowledgments for the collaboration between IESE and UFBA and for the participating companies. The appendix contains a discussion of related studies and threats to the validity of the study.

2 Architecture Documentation in Research and Practice

Making software architecture explicit and persistent is a key factor in utilizing the potential it offers. This is reflected by the fact that almost all comprehensive approaches for software architecture also cover documentation. Examples are [4], [24], [26] or [11]. There is even a standard in place for the description of software architectures (ISO/IEC/IEEE 42010) [13].

Architecture views have been introduced to address the need to deal with the complexity of software systems and are still one of the central concepts of the discipline. They help to separate different concerns of the software systems according to the needs of different stakeholders. Different view sets have been presented since then; some of the best known and most frequently applied ones include Kruchten's 4+1 View model [16], the Siemens Four Views model [12], or SEI's Views and Beyond approach [5]. However, the usage of different architecture view types is not sufficient anymore to handle the complexity of modern software systems and to describe them adequately for different stakeholders. The amount of information can be still so high that efficient working is still hampered, making studies as ours necessary.

For description languages, different ways are used and have been proposed in practice and research. These range from simple whiteboard sketches to formal architecture description languages, with the degree of formality being the main varying factor. In research, high levels of formality are typically valued in architecture description languages, as they allow sophisticated analyses and automated processing of information. Examples are ACME [10] or AADL [9]. In contrast, practice values fast creation and understandability, as architecture documentation is mainly used as a vehicle for information exchange. This is why whiteboard sketches and “PowerPoint architectures” are widely used. However, despite the disadvantages for which it is frequently criticized, the predominant description language for architectures is UML [19]. Often UML diagrams are complemented with descriptions in natural language. Accordingly, the format in which architecture documentation is distributed also varies. This includes electronic documents and presentation files and webpages, but also model files created with modeling tools like Sparx System's Enterprise Architect [6] or IBM's Rational Software Architect [22].

In recent research, the relatively new discipline of architecture knowledge management has emerged for explicating and persisting architecture information. Farenhorst and de Boer published a state-of-the-art survey on this topic [8] and observed four main directions of architecture knowledge management: *1. Shar-*

ing architecture knowledge to make architecture information efficiently available to stakeholders, as in [7] or [1]. *2. Aligning architecting with requirements engineering* to create a connection between architecture information and requirements, as in [21]. *3. Towards a body of knowledge*, to create a comprehensive encyclopedia of architecture information, as in [2]. *4. Intelligent support for architecting* to enable working efficiently with architecture and its documentation, as in [25]. However, architecture knowledge management methods are not yet widely applied in practice at this point in time.

3 Research Methodology

For the description of our methodology research, we distinguish the following phases: planning the survey, designing and conducting the survey, and analyzing the data.

3.1 Planning the Survey

We defined the overall goal of the survey and the four research questions as introduced in Section 1.2. Based on these, we planned and designed the survey and derived the actual survey questions for the participants.

The target group for the survey consisted of software developers in industry. Thereby it was not important whether they actually had software architecture documentation available in their implementation work because even if they did not, they could still be asked about their wishes for the future. To invite participants we decided to use email; however, we did not want to merely contact random software companies. To increase the chances of a high response rate, we compiled a list of past and current customers and project partners from industry. As we typically have one or a small number of contact persons, we contacted them directly and asked them to distribute the information about the survey internally to software developers in their organization with the request to participate. In this way we contacted 92 IT organizations from Europe, Asia, North and South America, ranging from two to around 130,000 employees. Additionally, BITKOM[1], the German Association for Information Technology, Telecommunications and New Media, Software Foren Leipzig[2], and the Software Technologie Initiative e.V[3]. provided assistance by distributing the information via their mailing lists.

3.2 Designing and Conducting the Survey

The four research questions (see Section 1.2) provided the framework for the derivation of our survey questions. Figure 1 depicts the resulting structure of survey questions as a matrix. The key distinctions are between architectural information and its representation and between the as-is situation for the participant and wishes for a to-be situation related to architecture documentation.

[1] http://www.bitkom.org

[2] http://www.softwareforen.de

[3] http://www.sti-ev.de/

Additionally, we asked for information about the participants' background (e.g., regarding their company, see Section 4.1).

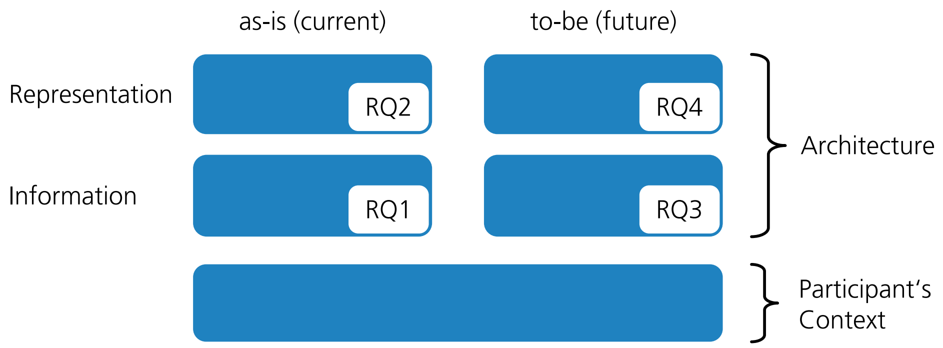


Figure 1. Structure of survey questions and relationship to research questions

In Figure 2, the flow of survey questions is presented. It starts with a question about the preferred language for conducting the survey. As this research was done in a German-Brazilian cooperation with many participants from Germany and Brazil expected, we offered the languages German, Portuguese and in addition, English.

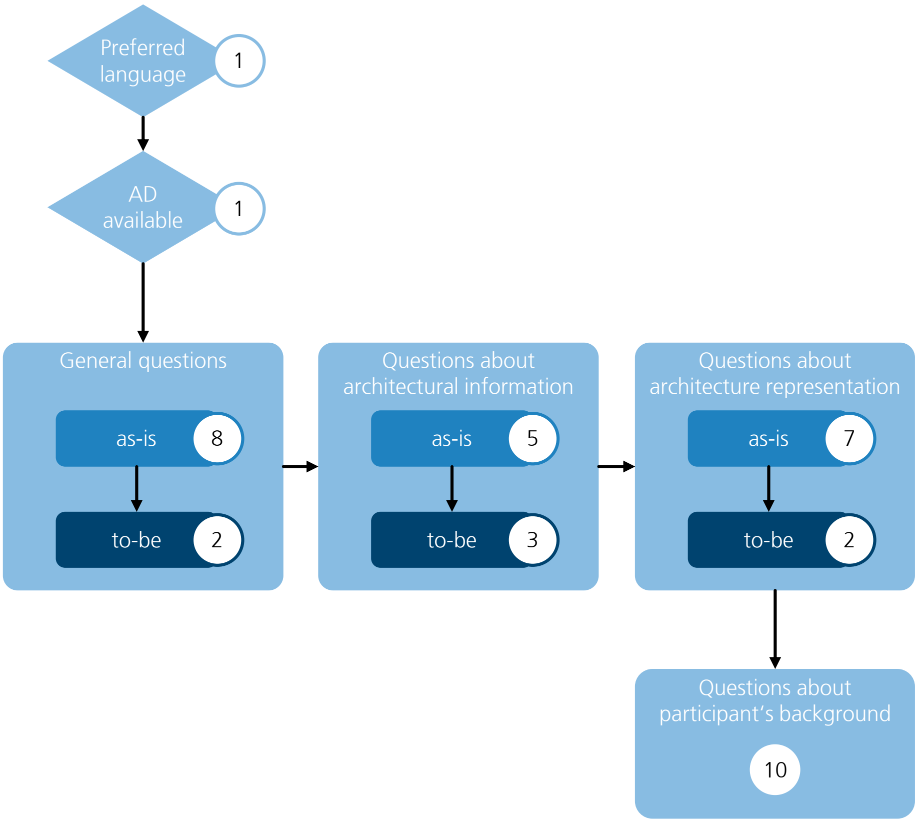

Preferred language
1
AD available
1
General questions
as-is
8
to-be
2
Questions about architectural information
as-is
5
to-be
3
Questions about architecture representation
as-is
7
to-be
2
Questions about participant's background
10

Figure 2. Flow of questions in the survey. Medium blue blocks (as-is) were only asked if architecture documentation was available. The circled number indicates the number of questions in the section

Then we asked about the availability of architecture documentation for the participants and their tasks as a developer. This question had an impact on the further flow of survey questions: Only if a participant indicated that architecture documentation was available were the questions about the as-is situation asked; otherwise they were not visible for the participant.

The main part of the survey consisted of three pages of questions, each visually separated into a set for the as-is situation and a set for the to-be situation: First, general questions about architecture documentation were asked, without any differentiation between the information aspects and their representation. Second, questions with a focus on architectural information in architecture documentation were asked. Third, questions about the representation of architecture information were asked. Finally, a set of questions about the participant's background were asked (cf. Section 4.1). We had two types of questions: First, questions with a fixed set of answers, partly single- and partly multi-selection ones. Second, there were questions with a free text answer.

We created an online questionnaire containing 42 questions using the Enterprise Feedback Suite by questback[4]. We conducted the survey in the period from December 1st, 2012 to January 31st, 2013.

3.3 Analyzing the Data

Only a subset of the participants who started the survey actually finished it. We considered the survey as finished when the participants clicked the submit button. Regarding the analysis and evaluation, when we talk about participants we refer to those who finished the survey.

A total of 147 participants (N=147) have been included in the data analysis. The net sample was 345. This figure encompasses both the completed surveys and the ones that have been interrupted. The total sample, i.e. how many people saw was 572, making it a response rate of 60.31% and a completion rate of 25.7%. Nevertheless, we do not have a complete data set for each question, as not all questions were mandatory. That is, for each question the sample size might vary.

As described above, we asked about the availability of architecture documentation and excluded the respective questions if no such documentation was available. Not all participants had architecture documentation available for their tasks. Thus, for the questions about current architecture documentation, we have only answers from a subset of the participants (N=109).

For questions with fixed answers, we counted the results in the analysis. For the evaluation of free-text results, we grouped the answers into coherent categories with an appropriate name to cover the full range of answers. Then we additionally aligned these answer categories across questions wherever this was meaningful.

[4] http://www.questback.com/solutions/market-research/

4 Results

In the following sections, we will describe the results of the survey.

4.1 Overview of Survey Participants and their Context

All participants are employed in industry and are somehow related to software development. Participants are affiliated with companies in eight different countries, mainly in Germany (59%), Brazil (23%), and Finland (13%). Further participants came from France, Japan, Sweden, Switzerland, and the United States.

Although the survey aimed at studying the development perspective on software architecture, many participants with a slightly different focus in their own position contributed to the survey. Figure 3 depicts the distribution of the participants' occupational positions. The largest group are developers (46%), followed by architects (23%) and managers (20%). The participant's position was asked as free text, thus we consolidated the answers into the depicted categories. In the category *manager*, in particular, we grouped quite different roles, ranging from team leader via product manager to chief executive officer.

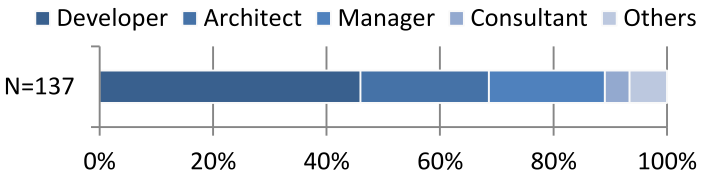


Figure 3. Current position of the participants in their companies

In order to judge the professional experience of the participants, we asked for the number of years they had already been in their current or in a similar position. The answers, grouped from an open question, were the following: 0 to 3 years – 27%, 4 to 7 years – 34%, 8 to 11 years – 17%, 12 to 15 years – 10%, more than 15 years – 12%.

In order to characterize the companies the participants are affiliated with, we asked for the industry sectors they work in (see Figure 4). Most participants work for companies whose main business is software development for multiple industries (27%). The other companies are developing software for customers from a dedicated sector or for their own business units. Other strong sectors in our survey were building construction management (10%), automotive (8%), energy (8%), and finance (8%). While the survey in general was anonymous, we asked the participants at the end whether they agreed to have their company's name published in the study. The participating companies included, among others: Accenture, amiando, arago, Best Code – Qualidade de Software, Daubit Development Service, Deloitte Consulting, Denso, Diamant Software, GEA Group, Gebhart Quality Analysis (QA) 82, IF Sertão PE, Kienbaum Management, KSB, msg systems, Murex, Partec, psb intralogistics, Radix Engenharia e Software, Rohde & Schwarz, SALT Solutions, SAP, Saphir Gesellschaft für Softwaresysteme, SEEBURGER, Software AG, SOLUTIS Tecnologias, Systemum, T-Systems, Talend, Tekla, Thoughtworks, UNIT4 Business Software, WIKON Kommunikationstechnik, and White Lion Technologies.

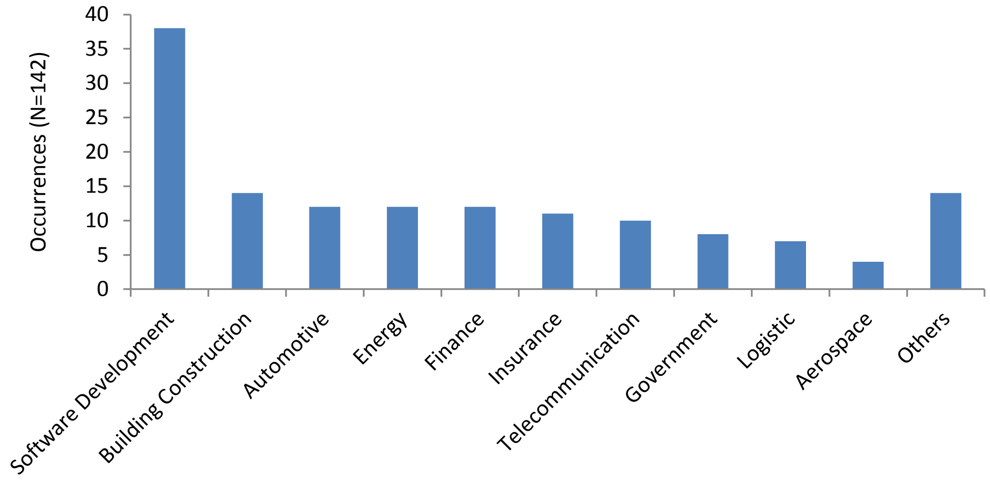


Figure 4. Sectors of the participants' companies

There are significant differences in the participants' companies regarding the number of people contributing to software development. 41% reported fewer than 100 people in software development, 41% reported 100 to 1000, 11% reported 1000 to 5000, and 7% reported more than 5000 people in software development.

The majority (50%) of the participants develop software according to a combination of agile and conventional development processes. 33% work completely with agile development processes, 7% work with purely conventional development processes. 10% do not use a structured development process at all.

Finally, we asked the participants to rate the size of the product they are contributing to. 22% contribute to a very large product, 32% contribute to a large product. 34% contribute to a medium-size product, 11% contribute to a small product, and 1% to a very small product. How to estimate product size was left up to the participants for the sake of simplicity in answering the question.

4.2 Main Findings

This section describes the results of the survey. Please note that the results of the *general* questions are consistently integrated into this structure. We identified five main findings, which are summarized below and discussed in Section 5.

1. Architecture documentation is often not up to date and thus strongly lacks utility. In particular, architecture documentation is not kept up to date following changes in requirements or changes in the source code.
2. Architecture documentation is often provided in a "one-size-fits-all" manner. Consequently, it does not provide the right information for specific stakeholders and their current tasks. Developers, in particular, have strongly varying needs regarding information and the level of detail, which can only be covered by more specific architecture documentation.
3. Architecture documentation is often inconsistent. Inconsistency comes in different forms, like inconsistent structure within and across documents, inconsistent notations, or contradictory information. A higher level of consistency is desirable for developers in order for them to understand the architecture more easily and to come up with higher-quality implementation.
4. Architecture documentation often does not provide sufficient navigation support to easily find the right information. Developers want a more interactive way of working with architecture documentation: better navigation (along the hierarchical decomposition and general traceability to related aspects) and a powerful search functionality ("Google-like" was often mentioned).
5. Architecture documentation is often scattered across different artifacts, such as several documents, presentations, or emails. This makes it difficult for developers to efficiently find information that is relevant for their implementation tasks. Developers ask for a central access point where they can easily find all the architecture information they need.
6. Aggregating all the answers to the question "What architecture information do you need for best support of your development tasks?" provides a very complete and mature picture of which information architecture documentation should contain. This serves as helpful confirmation of what we see as architecturally relevant is also demanded by developers. Nevertheless, it has to be taken into account that architecture documentation should be strongly tailored to the specific usage.

4.3 Architectural Information: The as-is Situation

In the *general questions*, we asked: "What do you consider as the main problems with the architecture documentation you work with?". With respect to architectural information, the most frequent answers given are listed below:

- Outdated architecture documentation (25 [occurrences])
- Inadequate level of granularity (19)
- Implementation not in sync with architecture anymore (17)
- Not specific for stakeholders and concrete situation (10)
- High costs/effort for creation and maintenance (6)
- Rationale missing (4)

We asked about the amount of architecture documentation available. The results are depicted in Figure 5. The majority of participants have fewer than 100 pages of architecture documentation available in their development projects.

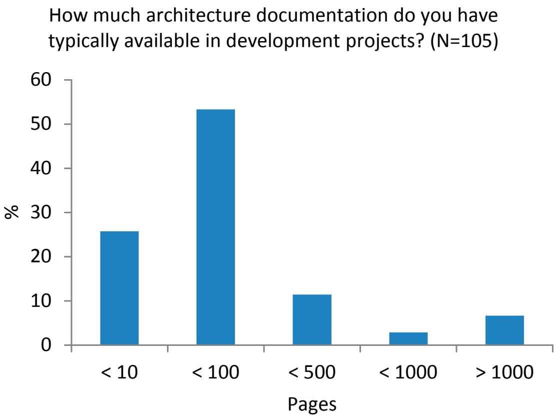
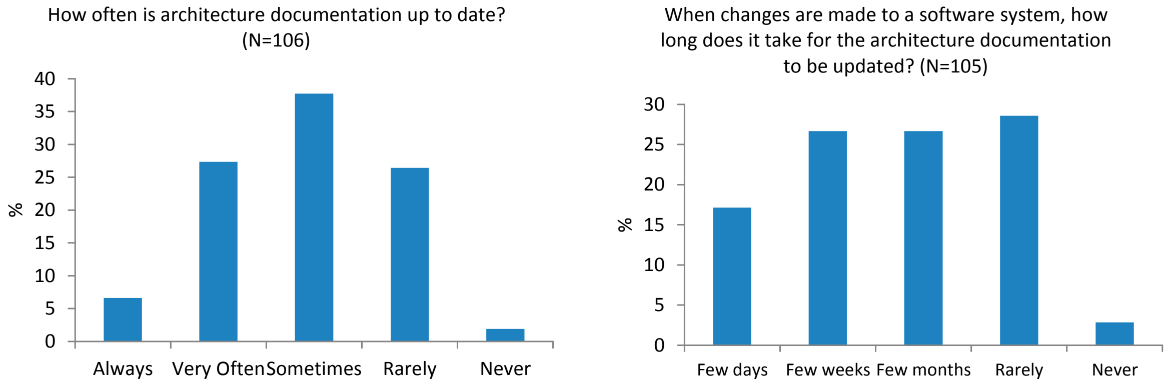


Figure 5. Amount of architecture documentation

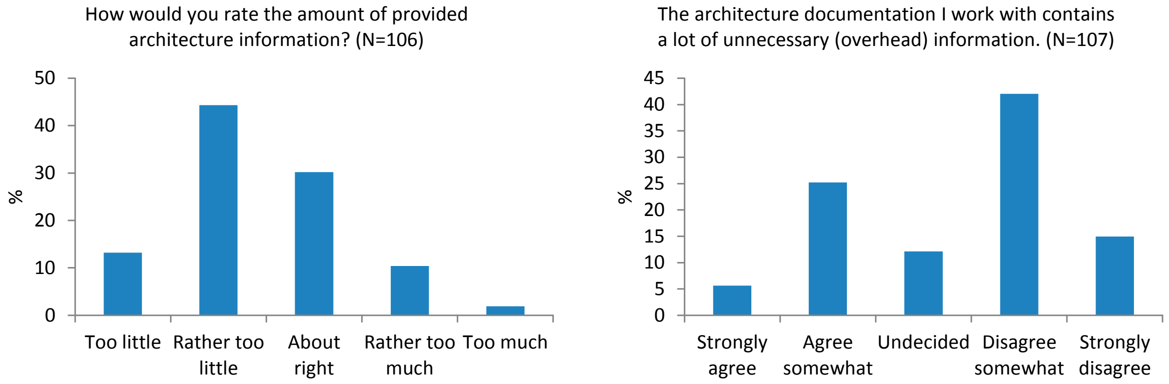
Figure 6 shows the answers to our questions about the up-to-dateness of architecture documentation. They confirm that architecture documentation is often outdated and if updated at all, with a strong delay. The results show remarkably high similarity to those reported in [17]. Based on these results we have to say that almost no improvement has been achieved within a whole decade.


How often is architecture documentation up to date? (N=106)

When changes are made to a software system, how long does it take for the architecture documentation to be updated? (N=105)

Figure 6. Up-to-dateness of architecture documentation

We asked the participants to rate the perceived adequacy of the amount of architecture information provided (see Figure 7). A tendency can be observed that there is rather too little architectural information available. Some participants agree that architecture documentation also contains unnecessary information but most participants see no unnecessary information provided. Keeping in mind that many participants have too little architecture information, it is no surprise that they do not see much overhead. When architecture documentation becomes extensive, the need for better orientation and specific tailoring increases.


How would you rate the amount of provided architecture information? (N=106)

The architecture documentation I work with contains a lot of unnecessary (overhead) information. (N=107)

Figure 7. Adequacy of amount of architecture documentation provided

Finally, for this category, we asked the participants about the overall quality of their architecture documentation. The majority of the participants (48%) rated it "average", 29% rated it "good" and 18% "poor". Despite the overall tendency being slightly positive, 71% of the participants giving a rating below "good" shows that there is still enough improvement potential for the quality of software architecture documentation. Figure 8 illustrates the results.

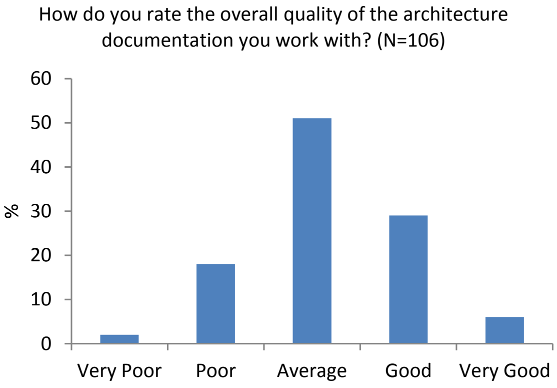


Figure 8. Overall quality of architecture documentation

4.4 Representation of Architectural Information: The as-is Situation

In the *general questions*, we asked: "What do you consider as the main problems with the architecture documentation you work with?" and received the following most frequent answers with respect to the representation of architectural information:

- Inconsistencies and missing structure (15)
- Information scattered across documents (8)
- Missing traceability to other artifacts (5)
- Missing evolution and version support (2)
- Degree of formality too high (1)

In order to get additional insights into the problems with the representation of architecture information, we asked: *"What problems do you see in the way architecture information is described?"* The results are shown in Table 1.

Table 1. What problems do you see in the way architecture information is described?

Answer Category	Occurrences	% (N=42)
Missing common formats and structures	6	14,3
Targets too many groups and thus not specific	5	11,9
Unnecessary information	5	11,9
Wrong or varying degree of abstraction	4	9,5
Missing traceability to external information	3	7,1

Missing details in the written description	3	7,1
Hard to consistently update	3	7,1
Missing information about business logic, focus only on infrastructure	2	4,8
Description of things far in future, which developers don't know exactly	1	2,4
Missing consistency	1	2,4
Missing highlighting of information	1	2,4
Non-standardized diagrams, missing common terminology	1	2,4
Information duplication	1	2,4
Missing overview on architecture documentation	1	2,4

Additionally, we asked: *"What problems do you see in terms of finding the architecture information you need?"* The results are shown in Table 2. The answers to these two questions confirm those given to the question about the main problems and provide additional details.

Table 2. What problems do you see in terms of finding the architecture information you need?

Answer Category	Occurrences	% (N=42)
Missing clarity in structure	13	23,2
No consistent storage of documents	11	19,6
Missing strong search functionality	10	17,9
Missing relationships / traceability (inside and to other artifacts)	8	14,3
Documents not up-to-date	8	14,3
Inconsistent terminology and notation	5	8,9
Too much information	4	7,1
Missing information	4	7,1
Mixing up information	1	1,8
Missing consistency among multiple documents	1	1,8
Information is too abstract	1	1,8
No knowledge, which information is there at all	1	1,8
Only understandable for people with same mindset as creators	1	1,8
Too complex diagrams	1	1,8

We asked about the main formats in which architecture documentation is provided. Architecture documentation is mostly provided as electronic documents, like Word or pdf (87%), model files (50%), and web pages (45%).

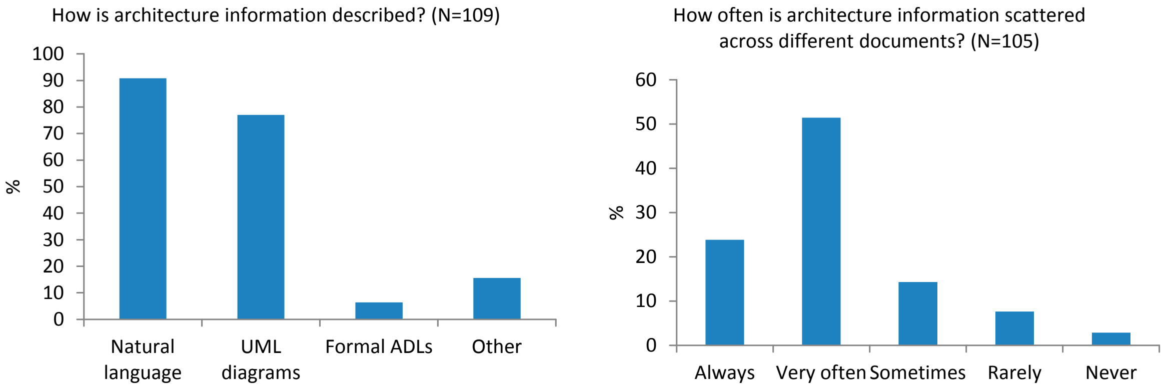


Figure 9. Notation of architecture documentation and scattering across documents

The two key notations for the description of software architecture are natural language and UML (see Figure 9). Formal ADLs are used very rarely. This confirms the findings of [18]. Informal diagrams in Visio or PowerPoint are also used. Another result is that architecture information is typically not consolidated in a single source of information but scattered across documents (see Figure 9).

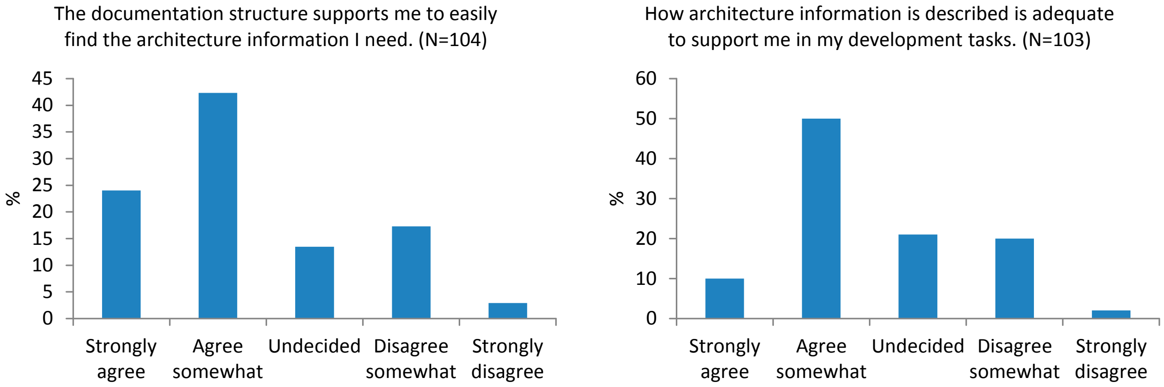


Figure 10. Perceived adequacy of representation of architecture information

We asked the participants how they perceive the support of their architecture documentation in finding specific information and performing their development tasks. Figure 10 depicts the results, showing that there is a tendency for the participants to perceive the representation as adequate.

4.5 Architectural Information: The to-be Situation

In the *general questions*, we asked: "What are your wishes in general for the future of architecture documentation?" and received the following most frequent questions with respect to architectural information:

- Up-to-date (19)
- In sync with implementation (18)
- Specific for stakeholders, concerns, tasks and contexts (14)
- Providing a system overview and the big picture (11)
- Presenting design decisions and rationale (10)
- Automatic creation/generation from code or requirements (10)
- More complete and more detail (7)
- Low cost and effort (3)

As we explained in our main findings, up-to-date architecture information that is in sync with the implementation is the main concern, but specificity for stakeholders and conveying the big picture are also important aspects.

In order to get a deeper insight, we asked the participants more specifically: "What architecture information do you need for best support of your development tasks?" From the answers it becomes evident that it is most important to developers to get an overall understanding of the complete system, as well as detailed information about the components in their scope together with interfaces and relationships to other components. More generally, we can say that they are interested in the complete information that constitutes software architecture. This is also one of our main findings as described in Section 4.2. Table 3 shows an overview.

Table 3. What architecture information do you need for best support of your development tasks?

Answer Category	Occurrences	% (N=103)
Components, interfaces, relationships, decomposition	44	42,7
Big picture	20	19,4
Mapping to implementation	12	11,7
Functional modularization	11	10,7
Data model and data flow	11	10,7
Deployment and deployment alternatives	9	8,7
Patterns and best practices	9	8,7
Project context	9	8,7
Technologies, frameworks and standards	9	8,7
(Discarded) Architecture decisions and rationale	8	7,8
Architecture drivers	8	7,8
Behavior	7	6,8
Restrictions & Constraints	3	2,9
Details on communication (protocols, latency, throughput, state, …)	2	1,9

Ownership & team responsibilities	2	1,9
Critical elements with particular caution in development	2	1,9
Testing information	1	1
Glossary	1	1
Tutorial	1	1
Traceability in architecture model	1	1
Evolution of architecture over time	1	1
Detailed functional requirements	1	1
Operation information	1	1
History of architecture	1	1

Additionally, we asked the participants how much architecture documentation they perceive as optimal. This question is a bit provocative and as expected, the most frequent answer was: "It depends". Mainly it depends on the target group and task, but also on the system and context. However, the answers also suggest that reducing the amount of information to what is really indispensable is desirable. Table 4 shows the results.

Table 4. How much architecture documentation is optimal?

Answer Category	Occurrences	% (N=83)
Depends on target group and task	16	19,3
Depends on project / system	15	18,1
Enough to provide an overview of the system and context	9	10,8
As minimal as possible	9	10,8
Enough to allow impact analyses, down to the code	5	6
An amount that is still possible to keep up-to-date	5	6
Other	32	38, 6

4.6 Representation of Architectural Information: The to-be Situation

In the *general questions*, we asked the question: "What are your wishes in general for the future of architecture documentation?" and received the following most frequent questions with respect to representation of architectural information:

- Easy creation, handling, updating, and maintenance (20)
- Connected and integrated information, artifacts and tools (16)
- Readable and understandable (12)
- Consistent and systematically structured and described (11)
- Hierarchical navigation (6)
- Good searching functionality (5)
- Guidance and connection for subsequent activities (5)
- Basis for automated activities (4)
- Centralization of information (3)

In order to get a deeper insight, we asked the participants more specifically: “In what format should architecture documentation be provided in the future?” Table 5 shows an overview. Webpages, electronic documents, and diagrams are the predominant wishes. UML also plays a significant role; formal ADLs do not. Additionally, the participants frequently voted for standard formats for which no specific (reader) tool has to be installed. It has to be noted that the types of answers given are not disjoint: UML diagrams can be provided in electronic document files or on webpages. While the intention of this question was to learn about the actual formats (as in the as-is part of the survey (cf: Section 4.4)), the answers still show the priorities of the participants quite well.

Table 5. In what format should architecture documentation be provided in the future?

Answer Category	Occurrences	% (N=117)
Webpages	25	21,4
Electronic documents	24	20,5
Diagrams	23	19,7
UML	19	16,2
Natural language	17	14,5
Standard formats	11	9,4
Architecture models	10	8,5
Wikis	9	7,7
Other	44	37,6

We asked: “What means should architecture documentation provide to help you find the information you need?” It can be observed that developers want interactive ways of working with architecture documentation, where it is possible to search information in different ways and navigate through hierarchical structures and related elements. Also, traces to other artifacts, like requirements documents have been rated as important. Table 6 shows an overview of the result data.

Table 6. What means should architecture documentation provide to help you find the information you need?

Answer Category	Occurrences	% (N=84)
Interactive search functionality	25	29,8
Links and navigation	24	28,6
Traces to artifacts	15	17,9
Directories	10	11,9
Clear structure	9	10,7
Mapping to implementation	8	9,5
Overviews: system overview, topic-map, ...	7	8,3
Centralized repository	6	7,1
Entrance points for goals, stakeholders, concerns, …	4	4,8

Other	11	13,1

Finally we asked how architecture should be described to make it more useful for the developer's implementation task. Table 7 shows the results. It becomes evident that clarity and structure in diagrams and language are of the highest importance.

Table 7. How should architectural information be described to make it more useful for your development tasks?

Answer Category	Occurrences	% (N=71)
Self-explaining, simple diagrams	15	21,1
Clear, concise, uniform, consistent	10	14,1
Clear terminology and language	6	8,5
Using a suitable notation	5	7,0
Providing detailed descriptions	4	5,6
Giving examples	3	4,2
Being traceable to other artifacts	3	4,2
Giving architecture decisions and rationale	3	4,2
Augmented with source code information	2	2,8
Tailored to the needs of reader	2	2,8
Describing the used patterns	2	2,8
Using overview diagrams	2	2,8
Other	5	7,0

5 Discussion

In the following sections, we will discuss the survey result and present our conclusions and next steps.

5.1 Survey Results

With this study, we aimed at confirming our experiences from many industrial projects and at laying the foundation for innovative and practically applicable architecture documentation methods for improved implementation. From this perspective, we revisit our main research questions and the responses received from the participants.

Concerning *architectural information*, it became evident that one of the participants' main concerns is up-to-dateness. Architecture documentation suffers significantly from being outdated in the majority of cases, making it less relevant and useful for developers. To improve this situation, the maintenance of architecture documentation has to be simplified and made more efficient. Centralization of architecture information is the most feasible way we see to achieve this. However, this requires powerful tooling to allow the efficient and automated creation of architecture documentation tailored to the needs of individual developers.

Such specific architecture information was another one of the main aspects mentioned by the participants. General and all-encompassing architecture documentation seems not adequate anymore for dealing with the size and complexity of modern software systems and development situations. Developers ask for architecture information that is specific for their scope, task, and context. Analogously to the aspect mentioned above, centralization of architecture information and automatic generation may be feasible strategies for addressing this. We outlined a first idea of such an approach in [23]. In terms of required architectural information, developers mainly ask for a system overview that provides the big picture of the system and its basic principles, complemented with detailed information within their scope, like components, interfaces, relationships, data, patterns, deployment, technologies, architectural drivers, etc. However, it is of major importance to reduce the amount of overhead information to the necessary minimum, without leaving out needed aspects. In general, we can say that the closer to the scope of the developer, the more details are needed, and analogously, the more details that need to be left out, the further away from the scope.

A clear, easy-to-understand and easy-to-follow connection between architecture information and implementation is a major concern for developers. While, such a connection can be established for detailed design using model-driven development and code generation techniques, this is currently not possible for architecture in general. Here we see the potential for further research and development of advanced tool support.

Concerning the representation of architecture information, consistency and uniform structure were two of the main concerns voiced by the developers. Organizations might need to invest more effort into establishing internal standards and a common terminology. Extended automation, for example using generation techniques, might also contribute to achieving this goal. This fits also quite well with the need for a single source of information, which was repeatedly mentioned by the developers.

In addition, the developers predominantly asked for interactive documentation that allows easy navigation and searching. We understand that static architecture documents as they are common are not adequate for serving the needs of developers. Future research needs to concentrate on such forms of documentation.

And finally, readability and understandability need to be increased. We consider this to be another argument for standardization and clarity through reduction of information.

5.2 Conclusions

We conducted an international study on the as-is and to-be situation of software architecture documentation from the perspective of developers. We are happy to having received contributions to this research from a total of 147 participants from industry.

The study confirmed that software architecture is a very important topic in industrial software development and that many companies are successfully engaged in it. Aggregating the answers to what practitioners want as architectural support for their development activities results in an impressive list covering nearly all the literature topics on software architecture documentation. We identified a lot of interesting improvement opportunities regarding how software architecture can become even more helpful.

Our main findings are that architecture documentation has to become up to date and consistent in order to better serve the developers’ needs. Additionally, developers demand more specific architecture documentation targeted at their concrete context and tasks. Such architecture documentation should be complemented by improved navigation and search possibilities.

As researchers of Fraunhofer, we strongly aim at improving the industrial applicability of software architecture methods and tools. We conducted this survey to complement our own experiences from projects and discussions with practitioners. The survey confirms that architects need more tool support for the creation of adequate architecture documentation. For the future, we plan to extend tools and increase the level of automation as next steps towards realizing the identified improvement potentials.

6 Acknowledgments

We would like to thank all the participants who contributed by answering our questions. Additionally, we would like to thank our colleagues, business partners, and networks (Bitkom, Softwareforen Leipzig, STI e.V.), who supported us in inviting and reaching so many participants for the study. We thank our colleagues Jessica Jung for her help regarding the empirical aspects and Jens Knodel for reviewing the report.

A scientific publication will be available from Springer-Verlag Heidelberg Berlin: Rost, D., Naab, M.. 2013. Software Architecture Documentation for Developers: A Survey. *European Conference on Software Architecture* (ECSA 2013) (July 2013), LNCS 7957, pp. 72–88.

6.1 Collaboration between IESE and UFBA

This survey was conducted as part of the cooperation between the Fraunhofer Institute for Experimental Software Engineering (IESE), Germany and the Federal University of Bahia (UFBA), Brazil. Both are collaborating to establish a Fraunhofer Project Center on Software and Systems Engineering in Bahia. Its goal is to expand the activities of UFBA beyond its academic and scientific activities in order to act as a strong source of new technologies and solutions to the industry both nationally and internationally.

6.2 Participating Companies

We would like to particularly thank the following companies for their participation: Accenture, amiando, arago, Best Code – Qualidade de Software, Daubit Development Service, Deloitte Consulting, Denso, Diamant Software, GEA Group, Gebhart Quality Analysis (QA) 82, IF Sertão PE, Kienbaum Management, KSB, msg systems, Murex, Partec, psb intralogistics, Radix Engenharia e Software, Rohde & Schwarz, SALT Solutions, SAP, Saphir Gesellschaft für Softwaresysteme, SEEBURGER, Software AG, SOLUTIS Tecnologias, Systemum, T-Systems, Talend, Tekla, Thoughtworks, UNIT4 Business Software, WIKON Kommunikationstechnik, and White Lion Technologies.

7 Appendix

7.1 Related Studies

In 2003, Lethbridge, Singer, and Forward published the results of three studies on how software engineers use documentation [17]. Unlike the work presented in this report, their studies were not focused on architecture documentation only. Most of their main findings confirm our experiences in industry projects. They state: *"Documentation of all types is frequently out of date"*, *"Much mandated documentation is so time consuming to create that its cost can outweigh its benefits"*, and *"A considerable fraction of documentation is untrustworthy"*. We were interested in whether there has been any improvement in the past ten years concerning these factors. To investigate this, we asked questions like *"How often is architecture documentation up to date?"* or *"How much architecture documentation do you have typically available in development projects?"* and specifically replicated the question of *"in your experience, when changes are made to a software system, how long does it take for the architecture documentation to be updated to reflect the changes?"*. It is fair to say that the problems from one decade ago persist to a large extent until today.

In 2006, Koning and van Vliet reported on their study of four architecture descriptions from industry and their distances to the IEEE Standard 1471 in [15]. Specifically, they studied which parts of the documents were relevant to which stakeholder concern. They stated that "Our research makes it very understandable that readers complain about too much information," as well as "Almost none of the stakeholders is interested in the full report." This supports our assumption that unspecific architecture is a factor in inefficient and ineffective implementation activities. We included questions in our survey concerning the amount and specificity of architecture documentation provided to developers, like *"Please rate your agreement to the statement: "The architecture documentation I work with contains a lot of unnecessary (overhead) information.""*.

In 2012, Malavolta et al. reported the results of their study on the industrial usage of architecture description languages in [18]. The main findings of their study include: *"Organizations (even in domains involving critical systems) prefer semi-formal and generic ALs than formal and domain-specific ones like ADLs"* and *"[…] Code generation is not often required. Link to requirements (elicitation and specification) is important as well"*. We included questions about the usage of formal ADLs in our survey, as well as about developers' wishes concerning the features of architecture documentation. Besides this, we largely adopted their paper's structure as we found it quite compelling.

7.2 Validity

In this section, we describe threats to validity and limitations of the survey.

- Not all participants finished the study and submitted their results. As described in Section 4.3, we found that the answers of the participants who did not finish diverged considerably from the participants who did finish. However, we decided not to include unfinished surveys as they were not confirmed by the participants.
- The survey included questions that were not mandatory. Thus, not all participants filled in all questions. We always took the number of answers given as the reference and typically indicated how many results we got.
- We did not restrict the number of participants in a single company. This leads to the effect that some companies are represented by a single participant while others are represented by multiple participants. However, contexts and projects in larger companies are so different that we considered it valuable to get multiple contributions.
- Our survey was mainly targeted at developers, as indicated in the research question. Although we clearly put this in the survey invitation, several participants indicated that their main role is rather architect or manager. However, we nevertheless see this as valuable input and assume that these participants took a developer perspective (currently doing actual implementation work, having done it earlier, or supervising people who do implementation work).
- The questions we raised in the survey are not fully disjoint. So we received partially similar answers to our questions. However, this confirmed the general tendencies and top results fairly well.
- For free text questions, we derived categories from the participants' answers for aggregation purposes. These categories might depend on our background. However, we see a good match of these categories and typical topics in the literature.
- This study is related to other research performed by us [23]. Although we tried to maintain neutrality, we might have been biased in the survey design and analysis.

References

[1] Babar, M.A. and Gorton, I. 2007. A Tool for Managing Software Architecture Knowledge. *Second Workshop on Sharing and Reusing Architectural Knowledge - Architecture, Rationale, and Design Intent (SHARK/ADI'07: ICSE Workshops 2007)* (May. 2007), 11–11.

[2] Babu T., L. et al. 2007. ArchVoc--Towards an Ontology for Software Architecture. *Second Workshop on Sharing and Reusing Architectural Knowledge - Architecture, Rationale, and Design Intent (SHARK/ADI'07: ICSE Workshops 2007)* (May. 2007), 5–5.

[3] Basili, V.R. and Rombach, H.D. 1988. The TAME project: towards improvement-oriented software environments. *IEEE Transactions on Software Engineering*. 14, 6 (Jun. 1988), 758–773.

[4] Bass, L. et al. 1998. *Software Architecture in Practice*. Addison-Wesley Professional.

[5] Clements, P. et al. 2002. *Documenting Software Architectures: Views and Beyond*. Addison-Wesley Professional.

[6] Enterprise Architect: *http://www.sparxsystems.com/*. Accessed: 2012-03-05.

[7] Farenhorst, R. et al. 2008. A Just-In-Time Architectural Knowledge Sharing Portal. *Seventh Working IEEE/IFIP Conference on Software Architecture (WICSA 2008)* (Feb. 2008), 125–134.

[8] Farenhorst, R. and De Boer, R.C. 2009. Knowledge Management in Software Architecture: State of the Art. *Software Architecture Knowledge Management*. M. Ali Babar et al., eds. Springer Berlin Heidelberg. 21–38.

[9] Feiler, P. et al. 2006. *The Architecture Analysis & Design Language (AADL): An Introduction*.

[10] Garlan, D. et al. 2000. Acme: architectural description of component-based systems. *Foundations of component-based systems*. 47–67.

[11] Gorton, I. 2011. *Essential Software Architecture*. Springer Berlin Heidelberg; Auflage: 2nd ed. 2011.

[12] Hofmeister, C. et al. 1999. *Applied Software Architecture*. Addison-Wesley Professional.

[13] International Organization Of Standardization 2011. *ISO/IEC/IEEE 42010:2011 - Systems and software engineering -- Architecture description*.

[14] Knodel, J. 2011. *Sustainable Structures in Software Implementations by Live Compliance Checking*. Fraunhofer Verlag.

[15] Koning, H. and Vliet, H. Van 2006. Real-life IT architecture design reports and their relation to IEEE Std 1471 stakeholders and concerns. *Automated Software Engg.* 13, 2 (Apr. 2006), 201–223.

[16] Kruchten, P. 1995. The 4+1 View Model of A rchitecture. *IEEE Software*. 12, 6 (1995), 42–50.

[17] Lethbridge, T.C. et al. 2003. How software engineers use documentation: the state of the practice. *IEEE Software*. 20, 6 (2003), 35–39.

[18] Malavolta, I. et al. 2012. What Industry Needs from Architectural Languages: A Survey. IEEE.

[19] OMG 2011. *UML Superstructure Spcification 2.4.1*.

[20] Perry, D.E. and Wolf, A.L. 1992. Foundations for the study of software architecture. *ACM SIGSOFT Software Engineering Notes*. 17, 4 (Oct. 1992), 40–52.

[21] Pohl, K. and Sikora, E. 2007. COSMOD-RE: Supporting the Co-Design of Requirements and Architectural Artifacts. *15th IEEE International Requirements Engineering Conference (RE 2007)* (Oct. 2007), 258–261.

[22] Rational Software Architect: 2012. *http://www-01.ibm.com/software/awdtools/swarchitect/websphere/*. Accessed: 2012-03-05.

[23] Rost, D. 2012. Generation of task-specific architecture documentation for developers. *Proceedings of the 17th international doctoral symposium on Components and Architecture - WCOP '12* (New York, New York, USA, Jun. 2012), 1.

[24] Rozanski, N. and Woods, E. 2005. *Software Systems Architecture: Working With Stakeholders Using Viewpoints and Perspectives*. Addison-Wesley Professional.

[25] Tang, A. et al. 2011. Software Architecture Documentation: The Road Ahead. *2011 Ninth Working IEEE/IFIP Conference on Software Architecture* (Jun. 2011), 252–255.

[26] Taylor, R.N. et al. 2009. *Software Architecture: Foundations, Theory, and Practice*. Wiley.

Document Information

Title:	Architecture Documentation for Developers: A Survey
Date:	June 2013
Report:	IESE-028.13/E
Status:	Final
Distribution:	Public Unlimited