

# Software Architecture Documentation for Developers: A Survey

**Authors:**

Dominik Rost  
Matthias Naab  
Crescencio Lima  
Christina von Flach Chavez

IESE-Report No. 025.14/E  
Version 1.0  
June, 2014

---

A publication by Fraunhofer IESE



Fraunhofer IESE is an institute of the  
Fraunhofer Gesellschaft.

The institute transfers innovative software  
development techniques, methods and  
tools into industrial practice, assists com-  
panies 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



# Software Architecture Documentation for Developers: A Survey

Dominik Rost<sup>1</sup>, Matthias Naab<sup>1</sup>, Crescencio Lima<sup>2</sup>, Christina von Flach Chavez<sup>2</sup>

<sup>1</sup> Fraunhofer Institute for Experimental Software Engineering  
Kaiserslautern, Germany  
{dominik.rost, matthias.naab}@iese.fraunhofer.de

<sup>2</sup> Fraunhofer Project Center on Software and Systems Engineering  
Software Engineering Laboratory, Department of Computer Science  
Federal University of Bahia, Bahia, Brazil  
{crescencio, flach@dcc.ufba.br}

**Abstract.** Software architecture has become an established discipline in industry. Nevertheless, the available documentation of architecture is often not perceived as adequate by developers. As a foundation for the improvement of methods and tools around architecture documentation, we conducted a survey with 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 paper presents the results of the survey. The results confirmed the common belief that architecture documentation is most frequently outdated and inconsistent and backed it up with data.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 task and context. Developers seek for more interactive ways of working with architecture documentation that allow finding needed information more easily with extended navigation and search possibilities.

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

## 1 Introduction

### 1.1 The practical problem

Software architecture is accepted as an integral part of software engineering and 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, many industrial organizations still do not have *any architecture documentation in place*. A main reason for this is that creation of architecture documentation is inherently cost intensive. As a consequence, their software development suffers

from growing communication and alignment effort, which makes implementation increasingly inefficient, inconsistent, and in compliant to the architecture. During system evolution, this leads to architecture erosion [1] which can prevent the achievement of essential system qualities and leads to a decreasing maintainability. This holds true even for the initial development of the system [2].

However, we also observe organizations that *have architecture documentation but are not able to leverage the potential* that lies in it. 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 comprehensive architecture documentation to all software developers. Developers then have to understand the complete architecture documentation to extract the information relevant for the local scope of their task and adapt it to their context. Another problem is inconsistencies in content and form, making finding and understanding relevant architecture information a challenge. 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 found the decision 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 paper, we report on the creation and results of the study and discuss our main findings.

## 1.2 This Study

In this study, we focused on software architecture documentation for developers to complement our experiences from industry projects with the views of software developers. By this, we aim to create a basis for future improvement of methods and tools for architecture documentation, to make implementation more efficient and effective. So, 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 software developers. While methods and tools might target architects in the creation of documentation, in this study we ask developers about their view as users of the documentation. A second aspect is that we separate two dimensions: (1) architecture information vs. representation and (2) characterization of the current situation vs. requirements for the future. The combination of both dimensions gives us

four areas in which we ask 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 that 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 Related Studies

In 2003, Lethbridge, Singer, and Forward published the results of three studies on how software engineers use documentation [4]. Unlike the work presented in this paper, 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 [5]. 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 for 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 results on their study on the industrial usage of architecture description languages in [6]. Main findings of their study include “*Organizations (even in domains involving critical systems) prefer semi-formal and ge-*

*neric 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, as we found it quite compelling, we adopted to a large extent their paper’s structure.

The remainder of this paper 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 participants of our study. In Section 4.2 we present the results and our main findings. In Section 5 the main findings are discussed together with threats to the validity of the study and conclusions.

## 2 Architecture Documentation

Making software architecture explicit and persistent is a key factor of utilizing the potential that it offers. This is reflected by the fact that almost all comprehensive approaches for software architecture also cover documentation. Examples are [7] or [8]. There is even a standard in place for the description of software architectures [9].

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. Several different view sets have been presented since then and some of the most known and applied ones include Kruchten’s 4+1 View model [10], the Siemens Four Views model [11], or the SEI’s Views and Beyond approach [12]. However, the usage of different architecture view types is not sufficient anymore to handle the complexity of modern software systems and to describe them in an adequate form for different stakeholders. The amount of information can be 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. This reaches from simple whiteboard sketches to formal architecture description languages. Thereby, the degree of formality is 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 [13] or AADL [14]. In contrast, practice values fast creation and understandability, as it is mainly used as a vehicle for information exchange. The predominant description language for architectures is UML [15]. Often UML diagrams are complemented with descriptions in natural language. Accordingly, also the format, in which architecture documentation is distributed, varies. This includes electronic documents and presentation files, webpages, but also model files, created with modeling tools like Sparx System’s Enterprise Architect [16].

Recent research brought up the relatively new discipline of architecture knowledge management to explicate and persist architecture information. Farenhorst and de Boer published a state of the art survey on this topic [17] and observed four main directions of architecture knowledge management: 1. *Sharing architecture knowledge*, to make architecture information efficiently available to stakeholders, like in [18] or [19]. 2.

*Aligning architecting with requirements engineering*, to create a between architecture information and requirements, like in [20]. 3. *Towards a body of knowledge*, for the creation of a comprehensive encyclopedia of architecture info, like in [21]. 4. *Intelligent support for architecting*, for efficient working with architecture and its documentation, like in [22]. However, architecture knowledge management methods are currently not applied to a large extent in practice.

### 3 Research Methodology

For our 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 the four research questions as introduced in Section 1.2. Based on these, we planned and designed the survey and derived the concrete survey questions for the participants.

The target group for the survey was software developers in industry. Thereby it was not important whether they actually had software architecture documentation available in their implementation work, because asking them about their wishes for the future was possible in either way. To invite participants we decided to use e-mail, however we did not want to just contact random software companies. To increase the chances for a high response rate, we compiled a list of fitting past and current customers and project partners from industry. As we typically have only one or two contact persons, we contacted them directly and asked 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<sup>1</sup>, the German Association for Information Technology, Telecommunications and New Media and the Software Foren Leipzig<sup>2</sup> assisted by distributing the information via their mailing lists.

#### 3.2 Designing and Conducting the Survey

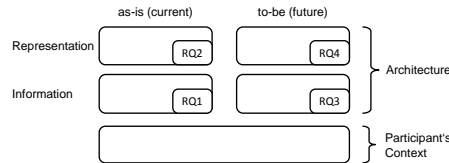
The 4 research questions (see Section 1.2) provided the framework for the derivation of our survey questions. Fig. 1 depicts the resulting structure of survey questions as a matrix. The key distinctions are between architectural information and its representation and the distinction between the as-is situation for the participant and wishes for a to-be situation related to architecture documentation. Additionally, we asked for information about the participants' background (e.g. on their company, see Section 4.1). In Fig. 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-

---

<sup>1</sup> <http://www.bitkom.org>

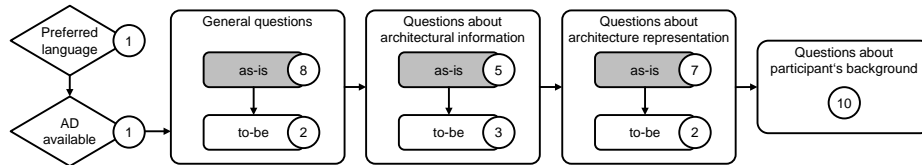
<sup>2</sup> <http://www.softwareforen.de>

Brazilian cooperation with many participants from Germany and Brazil expected, we offered the languages German, Portuguese and in addition, English.



**Fig. 1.** Structure of survey questions and relationship to research questions

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, the questions about the as-is situation were asked, otherwise they were not visible for the participant.



**Fig. 2.** Flow of questions in the survey. Grey blocks are only asked in case of architecture documentation available. The circled number indicates the number of questions.

The main part of the survey were 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, not differentiating between the aspects of information 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, partially single and partially multi selection ones. Second, there were questions with a free text answer.

We created an online questionnaire by using the Enterprise Feedback Suite by questback<sup>3</sup>, containing 42 questions. 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 starting the survey actually finished it. We considered the survey as finished when the participants clicked the submit button. For the analysis and evaluation, when we talk about participants we refer to the ones having finished the survey.

A total of 147 participants (N=147) have been included in the data analysis. Nevertheless, we did not have a complete data set for each question, as not all questions were mandatory. That is, for each question the sample varies.

<sup>3</sup> <http://www.questback.com/solutions/market-research/>

As described, we asked about the availability of architecture documentation and excluded respective questions if there was none. Not all participants had architecture documentation available for their tasks. Thus, for the questions about current architecture documentation we have only answers of 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 where it was meaningful.

Due to space limitations, we did not present all questions and answers in this paper<sup>4</sup>.

## 4 Results

### 4.1 Overview of Survey Participants and their Context

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

The survey aims at the development perspective on software architecture. Nevertheless, many participants with a slightly different focus in their own position contributed to the survey. Fig. 3 depicts the distribution of participants' occupational positions. The largest group is 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.

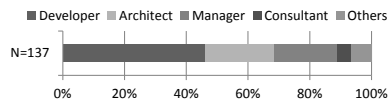


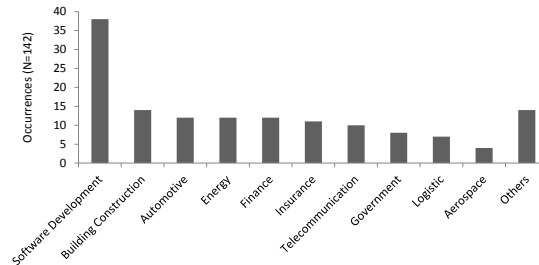
Fig. 3. Current position of the participants in their companies

In order to judge the professional experience of the participants we asked for the number of years they already are in their or 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 Fig. 4). Most participants work for companies that have software development for multiple industries (27%) as their main business. The other companies are developing software for customers of a dedicated sector or for their own business units. Other strong sectors in our survey are 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 agree with publishing their company's name in the study. Participating

<sup>4</sup> We provide the complete data set upon request.

companies included among others: Accenture, Deloitte Consulting GmbH, Denso, KSB, msg systems AG, Murex, SAP AG, Software AG, Talend, T-Systems, Tekla.



**Fig. 4.** Sectors of the participants' companies

There are significant differences, in the participants' companies in the number of people contributing to software development. 41% reported less 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 develops 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 the product size was estimated was left to the participants for simplicity of 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 5 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 with 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 the specific stakeholders and their current tasks. Developers in particular have strongly varying needs in information and the level of detail, which can only be covered with more specific architecture documentation.
3. Architecture documentation is often inconsistent. Inconsistency comes in different forms like inconsistent structure in and across documents, inconsistent notations, or contradicting information. A higher level of consistency is desirable for developers in order to easier understand the architecture and to come up with a higher quality implementation.

4. Architecture documentation does often not provide sufficient navigation support to easily find the right information. Developers wish a more interactive way of working with architecture documentation: Better navigation (along the hierarchical decomposition and general traceability to related aspects) and powerful search functionality (“Google-like” was often mentioned).
5. Aggregating all the answers to the question “What architecture information do you need for best support of your development tasks?” gives in our opinion a very complete and mature picture what information architecture documentation should contain. This is a helpful confirmation that 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 concrete 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?” and received with respect to architectural information the following most frequent questions:

- Outdated architecture documentation (25 [occurrences])
- Inadequate level of granularity (19)
- Implementation not in sync with architecture any more (17)
- Not specific for stakeholders and concrete situation (10)

We asked about the amount of architecture documentation available and its up-to-dateness. The results are depicted in Fig. 5. This also confirms that architecture documentation is often not up-to-date and if at all updated with a strong delay. This supports the findings reported in [4].

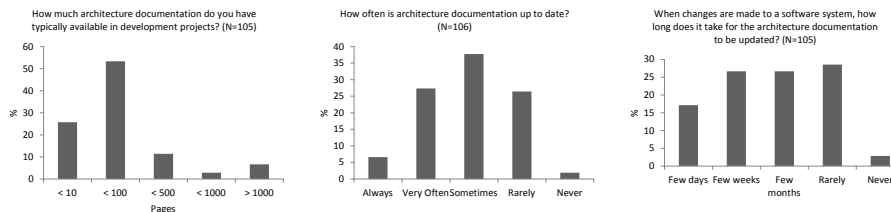


Fig. 5. Amount and up-to-dateness of architecture documentation

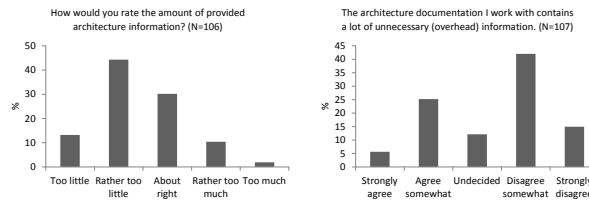


Fig. 6. Adequacy of amount of architecture documentation provided

We asked the participants to rate their perceived adequacy of the amount of architecture information provided (see Fig. 6). A tendency can be observed that there is rather

too little architectural information available. Some participants agree that there is also necessary information but most participants rather 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 arises.

#### 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 with respect to representation of architectural information the following most frequent questions:

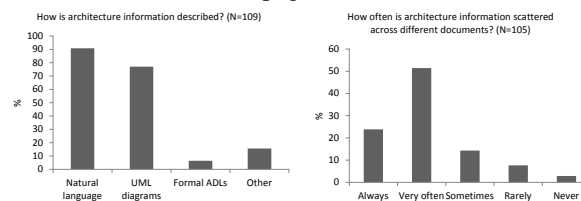
- Inconsistencies and missing structure (15)
- Information scattered across documents (8)
- Missing traceability to other artifacts (5)

In order to get some more insights into the problems with the representation of architecture information, we asked the question described in Table 1. The answers received, confirm and detail the answers to the general question described before. We summarized further, non-categorized answers as “other”.

**Table 1.** What problems do you see in the way how 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
Other	6	14,3

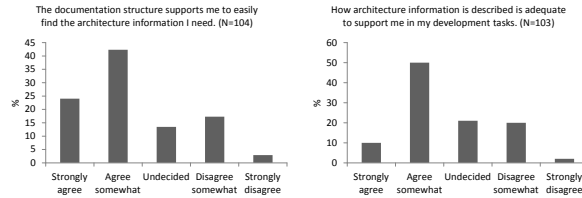
We asked about the main formats in which architecture documentation is provided. Architecture documentation is mostly provided as electronic documents, like Word or pdf (87%), model files (50%), and web pages (45%).



**Fig. 7.** Notation of architecture documentation and scattering across documents

The two key notations for the description of software architecture are natural language and UML (see Fig. 7). Formal ADLs are used very rarely. This confirms the findings of [6]. Mentioned under other, 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 Fig. 7).



**Fig. 8.** Perceived adequacy of representation of architecture information

We asked the participants how they perceive the support of their architecture documentation to find specific information and to conduct their development tasks. Fig. 8 depicts the results, showing that there is a tendency that the participants perceive the representation as adequate. To refine the insights about problems finding the needed architecture information, we asked the question described in Table 2.

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

Answer Category	Occurrences	% (N=56)
Missing clarity in structure	13	23,2
Information scattered across documents	11	19,6
Missing strong search functionality	10	17,9
Missing 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
Other	6	10,7

#### 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 with respect to architectural information the following most frequent questions:

- 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)

For 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 on components in their scope together with interfaces and relationships to other components. Table 3 shows an overview.

**Table 3.** What arch. 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
Other	17	16,5

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”. But mainly it depends on the target group and task, but also on the system and context. However the answers also suggest that a reduction of the information to the indispensable amount 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
Other	42	50,6

#### 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 with respect to representation of architectural information the following most frequent questions:

- 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)

For 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. It can be noted that webpages, electronic documents and diagrams are the predominant wishes. UML also plays a significant role, formal ADLs do not. Additionally, participants frequently voted for standard formats, for which no specific (reader) tool has to be installed.

**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 in finding the information you need?” It can be observed that developers wish for 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 in finding 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
Other	28	33,3

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 are of highest importance, in diagrams and language.

**Table 7.** How should arch. information be described to make it more useful for your dev. 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
Other	32	45,1

## 5 Discussion

In the following sections we discuss the survey result as well as threats to the validity of the study and present our conclusions and next steps.

### 5.1 Survey Results

With this study we wanted to confirm our experiences from many industrial projects and to lay 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 we 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 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 be made more efficient. Centralization of architecture information is the most feasible way we see to achieve this. But this requires powerful

tooling that allows efficient and automated creation of architecture documentation that is tailored to the needs of individual developers. Such specific architecture information was another of the main aspects, mentioned by the participants. General and all-encompassing architecture documentation seems not adequate anymore to face the size and complexity of modern software systems and development situations. Developers ask for architecture information specific for their scope, task, and context. Analogously to the aspect mentioned before, centralization of architecture information and automatic generation may be feasible strategies to address this. We outlined a first idea of such an approach in [23]. In terms of needed architectural information, developers mainly ask for a system overview, providing 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, but without leaving needed aspects out. In general we can say that the closer to the scope of the developer, the more details are needed, and analogously, the more details need to be left out, the farther away from the scope. And finally, a clear, easy to understand and to follow connection between architecture information and the implementation is a major concern of developers. While with model driven development and code generation techniques, such a connection can be established for detailed design, for architecture in general this is currently not possible. 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 of developers. More effort might be needed to establish internal standards and a common terminology within organizations. But also extended automation, for example with generation techniques might contribute to achieve this goal. This fits also quite well to the need for a single source of information, that developers mentioned repeatedly. Besides this, 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 to serve the needs of developers. Future research needs to concentrate on such forms of documentation. And finally, readability and understandability need to be increased, in which we see another argument for standardization and clarity through reduction of information.

## 5.2 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 having not finished considerable diverged from the participants having finished. However, we decided not to include unfinished surveys as they are not confirmed by the participants.
- The survey included questions that are 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 see it as valuable to get multiple contributions.
- Our survey is 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 that 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. These categories might depend on our background. However, we see a good match of these categories and typical topics in literature.
- The study is related to further own research [23] Although we tried to maintain neutrality, we might have been biased in survey design and analysis.

### 5.3 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 having received contributions from as much as 147 participants from industry to this research.

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 what practitioners wish as architectural support for their development activities is an impressive list, covering nearly the whole literature topics on software architecture documentation. We identified a lot of interesting improvement opportunities 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 for more specific architecture documentation, targeted at their concrete context and tasks. Such an 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 near future we plan to extend tools and increase the level of automation as next steps towards the identified improvement potentials.

**Acknowledgements.** This survey has been conducted in the context of the Fraunhofer Germany-Brazil cooperation between Fraunhofer IESE and UFBA. We would like to thank all the participants that contributed by answering our questions. Additionally, we would like to thank our colleagues, business partners, and networks (Bitkom, Softwareforen Leipzig) who support-

ed us in inviting and reaching so many participants for the study. We thank our colleagues Jessica Jung for her help on the empirical aspects and Jens Knodel for reviewing the paper.

## 6 References

1. Perry, D.E., Wolf, A.L.: Foundations for the study of software architecture. *ACM SIGSOFT Software Engineering Notes*. 17, 40–52 (1992).
2. Knodel, J.: *Sustainable Structures in Software Implementations by Live Compliance Checking*. Fraunhofer Verlag (2011).
3. Basili, V.R., Rombach, H.D.: The TAME project: towards improvement-oriented software environments. *IEEE Transactions on Software Engineering*. 14, 758–773 (1988).
4. Lethbridge, T.C., Singer, J., Forward, A.: How software engineers use documentation: the state of the practice. *IEEE Software*. 20, 35–39 (2003).
5. Koning, H., Vliet, H. Van: Real-life IT architecture design reports and their relation to IEEE Std 1471 stakeholders and concerns. *Automated Software Engg.* 13, 201–223 (2006).
6. Malavolta, I., Lago, P., Muccini, H., Pelliccione, P., Tang, A.: What Industry Needs from Architectural Languages: A Survey, <http://www.computer.org/csdl/trans/ts/preprint/tts2012990044-abs.html>, (2012).
7. Bass, L., Clements, P., Kazman, R.: *Software Architecture in Practice*. Addison-Wesley Professional (1998).
8. Rozanski, N., Woods, E.: *Software Systems Architecture: Working With Stakeholders Using Viewpoints and Perspectives*. Addison-Wesley Professional (2005).
9. International Organization Of Standardization: ISO/IEC/IEEE 42010:2011 - Systems and software engineering -- Architecture description. (2011).
10. Kruchten, P.: The 4+1 View Model of A rchitecture. *IEEE Software*. 12, 42–50 (1995).
11. Hofmeister, C., Nord, R., Soni, D.: *Applied Software Architecture*. Addison-Wesley Professional (1999).
12. Clements, P., Bachmann, F., Bass, L., Garlan, D., Ivers, J., Little, R., Merson, P., Nord, R., Stafford, J.: *Documenting Software Architectures: Views and Beyond*. Addison-Wesley Professional (2002).
13. Garlan, D., Monroe, R.T., Wile, D.: Acme: architectural description of component-based systems. *Foundations of component-based systems*. pp. 47–67 (2000).
14. Feiler, P., Gluch, D., Hudak, J.: *The Architecture Analysis & Design Language (AADL): An Introduction*. (2006).
15. OMG: *UML Superstructure Spcification 2.4.1*. (2011).
16. Sparx Systems: *Enterprise Architect*, <http://www.sparxsystems.com/>.
17. Farenhorst, R., De Boer, R.C.: Knowledge Management in Software Architecture: State of the Art. In: Ali Babar, M., Dingsøyr, T., Lago, P., and van Vliet, H. (eds.) *Software Architecture Knowledge Management*. pp. 21–38. Springer Berlin Heidelberg (2009).
18. Farenhorst, R., Izaks, R., Lago, P., Vliet, H. van: A Just-In-Time Architectural Knowledge Sharing Portal. Seventh Working IEEE/IFIP Conference on Software Architecture (WICSA 2008). pp. 125–134. IEEE (2008).
19. Babar, M.A., Gorton, I.: 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). pp. 11–11. IEEE (2007).
20. Pohl, K., Sikora, E.: COSMOD-RE: Supporting the Co-Design of Requirements and Architectural Artifacts. 15th IEEE International Requirements Engineering Conference (RE 2007). pp. 258–261. IEEE (2007).
21. Babu T., L., Seetha Ramaiah, M., Prabhakar, T.V., Rambabu, D.: 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). pp. 5–5. IEEE (2007).
22. Tang, A., Liang, P., Vliet, H. van: Software Architecture Documentation: The Road Ahead. 2011 Ninth Working IEEE/IFIP Conference on Software Architecture. pp. 252–255. IEEE (2011).
23. Rost, D.: Generation of task-specific architecture documentation for developers. Proceedings of the 17th international doctoral symposium on Components and Architecture - WCOP '12. p. 1. ACM Press, New York, New York, USA (2012).

# Document Information

Title: Software Architecture Documentation for Developers:  
A Survey

Date: June, 2014

Report: IESE-025.14/E  
Status: Final  
Distribution: Public Unlimited

Copyright 2014 Fraunhofer IESE.  
All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means including, without limitation, photocopying, recording, or otherwise, without the prior written permission of the publisher. Written permission is not needed if this publication is distributed for non-commercial purposes.