Evaluation Approaches for Software Architectural Documents: a Systematic Review

Rafael Ferreira Barcelos        Guilherme H. Travassos
{barcelos,ght}@cos.ufrj.br

Universidade Federal do Rio de Janeiro – COPPE/Sistemas

Abstract. Due to the importance of software architectural documents, its review became a fundamental activity for the success of software projects and for the software quality improvement. Software architecture evaluation is a subject that has been intensively studied by diverse research groups. However there is a lack of formal reviews aiming at identifying the existence of software evaluation approaches. The bibliographical reviews that have been accomplished can not be audited or repeated, making them entirely dependent upon the researchers for the obtained results, which could introduce bias on the research results. To prevent these limitations, this paper describes a systematic review aiming at the identification of existing architectural evaluation approaches. Based on the result of this review, we summarize the main characteristics, advantages and limitations of the identified evaluation approaches and propose the requirements for a checklist based software architectural documents evaluation approach.

1. Introduction

As the complexity and size of software applications grow, it has been more and more difficult to define at the firsts stages of the software development process a computational solution to attend to the specified requirements.

In this context, software architecture has been used as a tool for this kind of activity, since it assists in the definition and visualization of the computational solution at a high abstraction level [1].

Software architecture is the structure, or set of structures, which comprises software elements, the externally visible properties of those elements, and the relationships among them [2]. This structure is an artifact from a software development process and is represented by a document composed by one or more models, which represent different perspectives about how the system will be structured, and information sets that facilitate the understanding of the proposed computational solution. It is defined based on the software requirements. Among the different types of requirements, the quality requirements are the most important for the specification of an architecture since it exerts considerable influence over it structure [2].
The representation that describes the architecture, known as architectural document, is extremely important for the stakeholders because of the advantages provided to diverse aspects of software development. Garlan [3], for instance, reports the use of architectural document as a tool to facilitate the understanding of the proposed solution and to communicate this solution to the stakeholders.

Therefore, due to the importance of architectural documents, its review became a fundamental activity for the success of software project and for the software quality improvement. This claim is fortified by some facts such as (1) the review of an artifact prevents defects propagation to the subsequently created software artifacts and (2) the defects correction is less expensive if performed during the first stages of the software development process [4], like at the architecture specification phase that usually is performed in the beginning of the software development process.

Software architecture evaluation is a subject that has been intensively studied in diverse research groups, like SEI’s Product Line Systems Program [5, 6]. The review of an architectural document is extremely important and desired by the stakeholders because it may avoid the use of a defective document on the creation of subsequently artifacts [7]. An architectural evaluation consists in characterizing and evaluating architectural documents by means of systematic methods or procedures [8]. During this evaluation, it is verified if the information described on the architectural documents are consistent and if the represented architecture attends to the software requirements.

However, what could be considered regarding software architectural documents review? Is it possible to identify the existence of architectural evaluation approaches and characterize how such approaches could be accomplished?

This work intends to characterize how software architecture and architectural documents have been evaluated aiming at its quality improvement. The bibliographical reviews that have been previously conducted [5, 7-9] do not describe neither the searched scope nor the criteria used to select the evaluation approaches. Consequently, these reviews couldn’t be audited or repeated, making them entirely dependent upon the researchers for the obtained results, which could introduce bias on the research results [10].

To prevent those reviews limitations, this paper describes a systematic review [11] aiming at collecting, on some of the Software Engineering main sources [12, 13], studies about software architectural documents evaluation that allow the identification of existing architectural evaluation approaches. Based on the result of this systematic review, it is possible to summarize knowledge and to identify indications related to the main characteristics, advantages and limitations of the existing evaluation approaches. Besides, some perspectives for future research concerned with the inspection of software architectural documents are discussed.

This paper has been structured in more five sections besides this introduction. In section 2 and 3, the concepts related to systematic review and how it was conducted aiming at identifying architectural evaluation approaches are described. In section 4, an analysis regarding the identified evaluation approaches is done. This analysis points out some advantages and limitations on their use. In section 5, our proposal for a software architectural documents evaluation approach is described. And section 6 presents conclusions.
2. Systematic review basic concepts

During the study of a new knowledge area, researchers usually conduct a bibliographical review to identify publications related to a specific subject. However, this kind of review doesn’t use a systematic approach and doesn’t offer any kind of support to avoid bias during the selection of the publications that will be analyzed. A way to systematize the identification and analysis of publications, and primary studies in general, is using systematic reviews [14].

A systematic review is a means of identifying, evaluating and interpreting all available research relevant to a particular research question, topic area, or phenomenon of interest [14]. This approach is based on a specific sequence of activities that have to be accomplished, according to a research protocol previously defined.

The studies identified by a systematic review are classified as primary studies and the systematic review is a form of secondary study, since it depends on the primary studies results to be conducted. In the context of this work, the primary studies are scientific publications.

Kitchenham [14] describes diverse reasons for conducting a systematic review. We conducted one to summarize knowledge and identify indications related to the architectural evaluation approaches and consequently to be able to investigate the advantages and limitations of these approaches.

The information related to the conduction of this systematic review was analyzed and registered in a systematic review protocol, allowing its audit or repetition independently. An example can be found on [11, 14].

During the conduction of this systematic review the following process was used [11]:

– **Review planning**: During this step, the research’s purpose has to be specified by means of defining what will be searched, the information sources where the search will be executed have to be identified and the criteria used to select the studies that will be analyzed have to be defined. At the end of this step, a version of the protocol has to be created and the feasibility of the review has to be evaluated.

– **Review execution**: During this step, the studies related to the research goals and that satisfy the selection criteria are identified. This identification is executed on the selected information sources, using the search strings that have been created based on the keywords defined on the protocol.

– **Result analysis**: During this step, the identified studies are analyzed for the answers to the research questions. In the context of this work, the research questions aiming to identify software architectural evaluation approaches.

3. Conducting a systematic review to identify architectural evaluation approaches

The systematic review that was conducted aiming at identifying architectural evaluation approaches was based on the concepts described in [11].

The objective of this study was defined as follows:
Table 1 – Objective of the study defined using the GQM method [15]

| Analyze the software architectural evaluation approaches |
| For the purpose of characterize |
| With respect to the software inspection technique characteristics |
| From the point of view of software artifacts inspectors |
| In the context of the primary studies that describe these approaches |

On the following sections, the main information related to the conduction of this systematic review was described. A more detailed description can be found at [16].

Review planning

During the planning of this systematic review, diverse decisions were made related to the scope and the purpose of this search. This information was registered on a protocol. On Table 2, the main information is described.

Table 2 – Our systematic review protocol main information, according to the template described in [11].

| Research question focus: To identify and characterize approaches used to evaluate architectural documents; |
| Question: How is software architecture or an architectural document evaluated? Which are the approaches and what is the context that they are executed? |
| Context: These evaluation approaches have to be executed on software development projects that (1) have a requirement specification document, (2) have activities to project its architecture and (3) evaluate its architecture by using evaluation approaches. |
| Keywords: “software architecture”, “software modularity”, “architectural model”, “high level design”, “high level model”, “evaluate”, “assurance”, “review”, “inspection”, “verification”; |
| Sources selection criteria: The sources have to (1) make available a search engine allowing the conduction of the search on the web and (2) be evaluated by Qualis Capes1 [17] higher than international level “C”; |
| Inclusion and exclusion criteria: The publications have to be available and describe architectural evaluation approaches; |

Review execution and obtained results

Two main steps compose the execution of a systematic review. The first one consists on the identification of scientific publications related to software architecture evaluations. This identification is done by searching in the sources’ search engine for approaches, using the keywords defined on the protocol and organized according to specific search strings (Table 3).

1 Qualis Capes: Brazilian system used to rank scientific periodic and proceedings. The grade range from C to A, being level C the lowest but still good.
Evaluation Approaches for Software Architectural Documents: a Systematic Review

Table 3 – IEEE Xplore’s search string example created with keywords defined on the protocol

| (software <and> <not> (hardware, synthesize, circuit) <and> architecture <phrase> ( <or> (evaluate, assurance, review, inspection, verification))) | in | metadata |

After the identification, each publication had its abstract and introduction analyzed and, based on the inclusion and exclusion criteria, they were selected or not to a more thorough analysis.

During the first step, we had some problems with the ACM search engine [18]. These problems were related to inconsistencies observed when searches were executed on its engine, which made impossible to obtain the same results when the search was repeated. So, due to the sake of repeatability, we eliminated this source and the publications identified by its search engine.

This systematic review was conducted through December 2004. During this review, 80 scientific studies were identified and after its analysis, only 54 described at least one evaluation approach. Based on these studies, 20 software architectures evaluation approaches were identified.

During the identification of these approaches, we observed similarities on their evaluation technique. So, for the sake of space, we described these methods grouped by its evaluation technique.

The different kinds of evaluation technique that can be applied to architectural evaluation are [20]:

- **Questioning techniques**: These techniques can be used to investigate all software process artifact at any condition [5]. This group is characterized by the use of scenarios and checklists to accomplish this investigation.
- **Measuring techniques**: These techniques result in quantitative results. Rather than provide ways to generate the questions that will be asked about the software architecture, these techniques provide answers to questions an evaluation team might already have about particular qualities of the architecture. However, these techniques need a complete artifact to be measured during the evaluation. This kind of technique is composed mainly by simulation and prototypes approaches.
- **Hybrid techniques**: These techniques use simultaneously measuring and questioning techniques. This approach is used when some measuring techniques can be used to answer questions made by the questioning methods. [5].

### 3.1.1 Methods based on questioning techniques

Among the 20 identified evaluation methods, 11 use questioning techniques to evaluate software architectural documents. The most common questioning technique among them is scenario based execution, since scenarios allows the representation of the characteristics to be analyzed in an easy and complete way [20].

The method that stands out on this group against the others is SAAM [6], since it is used as the basis to create a great number of other evaluation methods.

SAAM (Software Architecture Analysis Method) evaluates the architecture by the analysis of how the changeability, variability and functional requirements have been attended. The main inputs of SAAM are the problem description, requirements documents and architectural representations.
This method can be used either for a single quality evaluation or for comparison of multiple ones. In the case of a single SA analysis, its main activities are (1) Scenario development based on the stakeholder’s point of view, (2) Mapping the scenarios onto the architecture’s structural decomposition and (3) Discrepancies identification based on the mapping results. Table 4 describes the main characteristics of questioning based methods, which are classified according to their publication date.

Table 4 – Main characteristics of questioning based methods

<table>
<thead>
<tr>
<th>Evaluation Method</th>
<th>Main characteristics</th>
<th>Ref.</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAR</td>
<td>SAR (Software Review Architecture) evaluates an architecture by the perspective of evolution and confiability attributes. It uses checklists based on the system’s characteristics and on criteria defined by the stakeholders. However, because the checklist depends on the system’s characteristics, a new checklist has to be created for each new evaluation candidate.</td>
<td>[21]</td>
</tr>
<tr>
<td>SAAM</td>
<td>SAAM (Software Architecture Analysis Method) evaluate the architecture’s suitability to the modificability, variability and functional requirements. It is the base of a great number of evaluation methods.</td>
<td>[6]</td>
</tr>
<tr>
<td>SAAMER</td>
<td>SAAMER (Software Architecture Analysis Method for Evolution and Reusability) is a SAAM extension and was defined aiming at evaluating an architecture through the perspective of evolution and reusability attributes.</td>
<td>[9]</td>
</tr>
<tr>
<td>PASA</td>
<td>PASA (Performance Assessment of Software Architecture) evaluates mainly the suitability to the performance requirements.</td>
<td>[8]</td>
</tr>
<tr>
<td>ALPSM</td>
<td>ALPSM (Architecture-Level Prediction of Software Maintenance) evaluate the maintainability of an architecture using a set of scenarios defined based on the desired adaptations and improvements.</td>
<td>[7]</td>
</tr>
<tr>
<td>ESSAMI</td>
<td>ESSAMI (Extending SAAM by Integration in the Domain) is a combination of analytical and reuse concepts achieved by integrating SAAM in domain-specific and reuse-based development processes. The main differences introduced by ESSAMI knowledge domain reuse by using analysis templates.</td>
<td>[9]</td>
</tr>
<tr>
<td>SAAF</td>
<td>SAAF (Software Architecture Analysis of Flexibility) is very similar to SAAM, besides it supports the identification and use of high complexity scenarios during the evaluation.</td>
<td>[7]</td>
</tr>
<tr>
<td>ARID</td>
<td>ARID (Active Reviews for Intermediate Designs) performs suitability analysis of intermediate architectural artifacts. It combines the active review’s philosophy with scenario based evaluation methods, like SAAM.</td>
<td>[5]</td>
</tr>
<tr>
<td>4+1</td>
<td>This method is used on architectural documents based on the 4+1 architectural representation [22]. By means of questioning, this method evaluates the coherence between the architectural visions and also the architectural decision made to attend to the requirements.</td>
<td>[23]</td>
</tr>
<tr>
<td>ALMA</td>
<td>ALMA (Architecture-Level Modificability Analysis) evaluates architectures using modificability scenarios. These method consists of the ALPSM and SAAF union, creating an unified architecture-level analysis method that focuses on modificability, distinguishes multiple analysis goals, has explicit assumptions and provides repeatable techniques for performing the steps.</td>
<td>[7]</td>
</tr>
<tr>
<td>ASAAM</td>
<td>ASAAM (Aspectual Software Architecture Analysis Method) is complementary to SAAM and uses aspects and refactoring concepts to evaluate the quality of an architecture.</td>
<td>[24]</td>
</tr>
</tbody>
</table>

3.1.2 Methods based on measuring techniques

The identified evaluation methods that are based on measuring techniques use mainly simulation and metrics to analyse the architecture. Table 5 summarizes the main...
characteristics of the selected methods, which are classified according to their publication date.

Table 5 - Main characteristics of measuring based methods

<table>
<thead>
<tr>
<th>Evaluation Method</th>
<th>Main characteristics</th>
<th>Ref.</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAEM</td>
<td>SAEM (Software Architecture Evaluation Model) uses quality metrics to evaluate an architecture and its representation. One of the main characteristics of this method is the use of GQM [15] to organize the metrics.</td>
<td>[9]</td>
</tr>
<tr>
<td>Model Checking based method</td>
<td>This method combines simulation and complex evaluation techniques, like model checking, to evaluate the reusability of an architecture. This method is one of the few that is supported by a tool (ARCADE - Architecture Analysis Dynamic Environment).</td>
<td>[25]</td>
</tr>
<tr>
<td>Simulation using RAPIDE</td>
<td>This method uses simulation to evaluate the performance of an architecture. To be evaluated, the architecture has to be represented by an ADL (Architecture Description Language) called RAPIDE.</td>
<td>[26]</td>
</tr>
<tr>
<td>OASE</td>
<td>OASE is used to experimentally evaluate the stability of an architecture. This method uses mainly metrics to summarize the software evolution pattern between different software product releases.</td>
<td>[8]</td>
</tr>
<tr>
<td>ArchOptions</td>
<td>ArchOptions evaluates the stability of an architecture by using some insights about possible modifications and informations about the architecture behavior in relation to these modifications.</td>
<td>[8]</td>
</tr>
</tbody>
</table>

3.1.3 Methods based on hybrid techniques

Among the methods based on hybrid techniques, ATAM is the most popular. ATAM (Architecture Tradeoff Analysis Method) [27] is an evolution of SAAM. Its purpose is to assess the consequences of architectural decisions in light of quality attribute requirements. So, it provides insights into how quality goals interact and trade off against each other.

To achieve these goals, ATAM uses both scenario based analysis and theoretical model of each considered quality attribute to evaluate the software architecture. The artifacts used as input to this method are the architectural document, the business goals and scenarios describing stakeholders’ perspectives.

Table 6 - Main characteristics of hybrid based methods

<table>
<thead>
<tr>
<th>Evaluation Method</th>
<th>Main characteristics</th>
<th>Ref.</th>
</tr>
</thead>
<tbody>
<tr>
<td>SBAR</td>
<td>SBAR (Scenario-Based Architecture Reengineering) is an architectural reengineering method that can use different evaluation techniques, depending on the system characteristics, to evaluate the architecture and identify if the reengineering process can finish.</td>
<td>[7]</td>
</tr>
<tr>
<td>ATAM</td>
<td>ATAM (Architecture Tradeoff Analysis Method) is an evolution of SAAM. It provides insights into how quality goals interact and trade off against each other. It uses both scenario based analysis and theoretical model of each considered quality attribute to evaluate an architecture</td>
<td>[27]</td>
</tr>
<tr>
<td>CBAM</td>
<td>CBAM (Cost Benefit Analysis Method) is an ATAM extension that does an economic modeling of the software by evaluating its architecture. So, it adds a monetary dimension to ATAM as an additional attribute to be traded off, allowing to model the costs and benefits of architectural design decisions and to provide means of optimising such decisions.</td>
<td>[8]</td>
</tr>
<tr>
<td>AD</td>
<td>This method is used to evaluate the performance of embedded systems. It uses scenarios and simulations techniques specific to embedded domain to evaluate software architecture.</td>
<td>[28]</td>
</tr>
</tbody>
</table>
Table 6 summarizes the main characteristics of the methods based on hybrid techniques, which are classified according to their publication date.

4. Obtained results analysis

After the identification and characterization of the evaluation methods, we accomplished two analyses: a general analysis about the scientific studies that describes the evaluation methods and another analysis about the characteristics of these methods. These analyses were accomplished to allow the comprehension about how the methods evaluate software architectural documents.

Analysis of the selected publications

During the analysis of the selected publications, we noted that the collected information weren’t enough for the characterization of some methods. This happened because the selected publications weren’t the one that originated the methods. So, to allow the characterization, we had to locate when possible those publications.

However, during the analysis of the selected publications, some features have been observed.

The first one is the lack of experimental studies. We looked for this kind of study because they allow a more thorough comprehension of the method and its application.

The studies accomplished with these methods are basically proof of concepts that aimed at to show its application and the quality improvement that the method could bring to the architectural documents [6, 29]. However, none of the selected publications describe how these studies were done or describe a comparison, for example, between the described method and an ad-hoc approach that also aimed at the evaluation of software architectures documents.

The second feature is the existence of bibliographical reviews that also aimed to identify architectural evaluation approaches [5, 7-9]. These reviews neither describe the inclusion and exclusion criteria used to select the approaches nor the sources used on the search, preventing the repetition of the review. However, two stands out because they define a framework to characterize the selected methods [7, 9].

Another characteristic of some of these reviews is that they were conducted aiming at the identification of motivations to justify their own evaluation approach. So, when a review is executed with these purposes, for example [8], usually all the selected approaches present a specific limitation that is treated by the new proposed method. The obtained results of this kind of review can be biased, since the researches may identify only methods that have the limitations that their approach deals with.

The third observed feature is the number of architectural evaluation methods that the systematic review was able to identify when compared to the other reviews. This systematic review identified 20 methods whereas the others ad-hoc reviews [5, 7-9] identified, on average, 8 methods.
Evaluation Approaches for Software Architectural Documents: a Systematic Review

Analysis of the identified software architecture evaluation methods

The main goal of the identified methods is the improvement of software architecture quality by the analysis of the document that describes it. Since experimental studies have demonstrated that the use of inspections techniques also improves the software artifact quality [30, 31], we used a set of characteristics present on inspections based approaches to characterize the identified evaluation methods aiming at their better understanding.

Based on these characteristics, the following ones have been extracted from the methods: the kind of evaluation technique, its evaluation process, inputs and outputs artifacts, the type of identified defects, the moment that it is executed, studies accomplished to evaluate the method and tools to support its use. A more detailed description of this characterization can be found in [16].

After the characterization and analysis of each method, four issues regarding these approaches have been identified (Table 7): some methods are too subjective, some have high execution costs, some doesn’t allow simultaneously evaluate the attendance to several quality requirements and some of them have a limited application context.

Table 7 – Mapping the limitations to the identified evaluations methods

<table>
<thead>
<tr>
<th>Evaluation Method</th>
<th>High Subjectivity</th>
<th>High execution costs</th>
<th>Restrict number of quality requirements</th>
<th>Limited application context</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAR</td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>SAAM</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>SAAMER</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>ATAM</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>PASA</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>SBAR</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ALPSM</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>ESAM</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>SAAF</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>ARID</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Model Checking based method</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>4+1</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>CBAM</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Simulation using RAPIDE</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>OASE</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Archoptions</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ALMA</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>ASAAM</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>AD</td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

These limitations were identified after an analysis of the information collected during the methods characterization. Therefore, the subjectivity of some methods was based on the kind of its evaluation technique. The execution costs was identified based on the activities presented on the evaluation process, the non-evaluation of the architecture by the attendance to multi-quality characteristics was identified based on

2 The obtained information related to the method SAEM were not enough to characterize it, so we didn’t include it on Table 6.
the types of defects identified by each method. The application on a limited context was identified based on the input artifacts required by each method.

In the context of this work, the subjectivity of an evaluation method consists on the lack of a deterministic approach or an exact and complete definition of the architectural artifacts analyzed during an evaluation method.

The methods defined as subjective are methods that mainly use scenario-based technique for evaluate architectural documents. This subjectivity occurs due to the incapacity of the stakeholders on identify and generate all possible scenarios that could be used during the evaluation. So, the definition of the selected scenarios set used during the evaluation depends on the subjectivity and creativity of the stakeholders [9]. Also, during the scenarios specification, problems related to bias can arise if the stakeholder unconsciously defines scenarios that don’t completely evaluate the architecture due to his/her familiarization with this artifact, for instance.

The execution cost of an evaluation method is defined by the number of stakeholders needed to take part of the evaluation process and by the complexity of the techniques used during this evaluation, like simulations and formal methods, which require specialists. Due to this, an evaluation method could require a great amount of time, human resources and monetary resources to be executed.

The methods that present high execution costs are related to the fact that they were developed to be used in academic or in large projects, which have high resources availability. So, because of the need of this high availability, only a small number of companies apply the methods in the right way [32].

Another deficiency observed is that these methodologies evaluate the architectures against a restrict number of qualities attributes. Dobrica and Niemela [9] suggest that the evaluation has to be done against several attributes, allowing a better understanding of the week points and the strengths of the complex systems actually developed.

Therefore, some approaches [33], knowing its limitation related to the number of quality attributes evaluated, execute several evaluation methods simultaneously aiming at a more complete analysis, increasing the evaluation costs.

The fourth identified problem is the main one present in Software Architecture field, which is related to the immaturity of this knowledge field. It is related to the lack of consensus in the community on the basics concepts and on how architectures have to represented [19, 34]. The software architectural evaluation methods, for example, are based on their own and specific documentation architecture approaches, what makes it difficult to apply in different project or contexts.

5. Proposal for an inspection based approach

The results obtained by the systematic review have identified limitations related to the use of architectural evaluations approaches that make difficult to apply them in industrial environments [32]. Several approaches based on checklist have already been defined aiming at the defects identification in software architecture models. However, the questionings asked by these checklists are either domain specific [35, 36], what makes it hard to apply in a different context from the one where the
checklists were projected, or are solution specific [21], what requires the creation of a new checklist for each new evaluation.

Based on this context, some improvements should be done regarding the checklists to make them effective and scalable to fit in industrial software projects. Our hypotheses is that a more general checklist based inspection approach, if used to identify defects on architectural documents related to the attendance of the software specified requirements, could minimize the limitations described previously.

The suggestion of keeping the checklist based evaluation technique is justified by its properties when compared to the others available evaluation techniques:

− An ad-hoc inspection offers neither support nor a formal and systematic process for the realization of the inspection. Additionally, the results obtained depend on the inspector’s expertise, skill, competence and experience [37].

− Reading technique is a procedure that aims to guide the inspectors to understand a software artifact and, as consequence, to identify discrepancies [30]. Evaluation approaches based on this technique [30, 31] have shown be more efficient in detecting defects when compared to others inspection techniques, as checklist for example. However, to use this technique, the artifact has to be represented in a specific and standardized way that can’t be achieved with software architectures yet.

So, the checklist based software architectural documents evaluation approach should be created with the following characteristics:

− The questioning items shall use concepts from the architectural construction activities, taking into account mainly how the suitability of the architectural models to the desired software quality characteristics can be done. The results of this analysis can avoid subjectivity in the choice of what will be evaluated.

− The items shall be grouped by the type of requirements that they evaluate. The existence of these items groups allows the evaluation of the suitability for the desired quality requirements.

− The evaluation of an architecture using the proposed approach shall be executed without the need of elaborated activities, such as the scenarios specification, allowing low investments.

− The questioning items shall evaluate the information on architectural documents which makes it independent of the architectural representation approach used, making possible to apply this method on different contexts and projects.

6. Conclusions and next steps

The conduction of a systematic review, mainly its planning step, when used as a tool on a bibliographical review allows a more complete specification about what has to be searched on technical sources, facilitating the search and improving the researcher’s focus during the read and analysis of the publications.

As result of this work, some indications related to the use of systematic review in Software Engineering can be identified. One of these indications are the advantages obtained on using systematic review as a tool to accomplish bibliographical review, due to the number of identified evaluation approaches when compared to the ad-hoc reviews.
Therefore, the main contribution of this work for the software architectural evaluation area is a more complete identification of evaluation methods and its characterization. Based on this characterization, that wasn’t entirely described here due to space limitations, it was possible to identify the advantages and limitations of these methods.

Besides, the obtained results motivated the proposal of a software evaluation approach. This inspection approach is being composed by a configurable checklist that evaluates the suitability of the architecture model, described by means of an architectural document, regarding the specified quality requirements [38] (Table 8).

This checklist has been defined based on the hypothesis that (1) knowledge used by the software architect to construct software architectures and that was used to define de questioning items and (2) configuration of the checklist according to the characteristics that have to be evaluated, allow the identification of defects on architectural documents and minimize the limitations presented on the current evaluation approaches.

Table 8 – An evaluating item from the proposed checklist

<table>
<thead>
<tr>
<th>Consistency evaluating item</th>
<th>Item description</th>
<th>Expected answer</th>
<th>Type of defect</th>
<th>Item purpose</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Item description</td>
<td>After the analysis of all graphical diagrams, have you found an architectural element without relationships, isolated from the others?</td>
<td>No</td>
<td>Extraneous information or Incorrect fact</td>
<td>To identify architectural elements that has no purpose on the software architecture</td>
<td><img src="image" alt="Diagram" /></td>
</tr>
</tbody>
</table>

This checklist based approach is going to be evaluated by two experimental studies. The first one is a pilot study that will be conducted aiming at evaluating the feasibility of such evaluation approach. The second, we have planned a case study that will be conducted to evaluate the application of the approach to evaluate software architectural documents elaborated on real industrial projects.
Acknowledgment

The authors acknowledge FAPEAM and CNPq – eSEE Project (4732135/2005-0) for the financial support for this work. Thanks to Arilo Claudio Dias Neto for his comments and suggestions regarding this text.

References


