Impact of Experience and Team Size on the Quality of Scenarios for Architecture Evaluation

Stefan Biffl[1], Muhammad Ali Babar[2], Dietmar Winkler[1]

[1]Vienna University of Technology, Austria, [2]Lero, University of Limerick, Ireland
[1]{Stefan.Biffl, Dietmar.Winkler}@tuwien.ac.at, [2]malibaba@lero.ie

Software and systems architecture is a success-critical issue in software projects. Changing non-functional quality requirements, e.g., performance, modifiability, and maintainability, can have strong impact on software architecture and can result in a high rework effort in case of changes. Architecture reviews help evaluating architectural design with scenarios in early stages of product development. Quality-sensitive scenarios represent a set of software requirements (including non-functional quality attributes). In this paper we empirically investigate the impact of potentially important factors on the number and quality of scenarios elicited in an architecture evaluation workshop: (a) scoring schemes for scenario quality, (b) workshop participant experience, and (c) team size for workshop group work. We report data analysis results from an empirical study where 24 reviewers at different experience levels identified over 100 scenarios. Main findings are: (a) results of different scoring approaches (frequency-based and expert scoring) agree very well regarding critical scenarios, (b) the scenario elicitation method was more important than individual experience, and (c) adding a new person to a team of size 3 or more increases scenario coverage by less than 10%.

Keywords: Architecture Evaluation, Quality Attributes, Scenarios, Empirical Studies, Performance Measurement.

1. INTRODUCTION

Software architecture (SA) plays a vital role in achieving desired quality attributes (such as performance, security, and modifiability) in a system. That is why practitioners and researchers have been emphasizing the importance of addressing quality-related issues at architecture level. There is a range of scenario-based architecture evaluation methods for quality assurance (QA) such as Architecture Trade-offs Analysis Method (ATAM) [21], Architecture Level Modifiability Analysis (ALMA) [24], and Performance Assessment of Software Architecture (PASA) [32]. The effectiveness of these approaches depends on the ability to identify high-quality scenarios, as the scenarios are a key input to the evaluation process [10][22]. There are several scenario-generation approaches such as brainstorming workshops [5], interviews [24], and use case analysis [32]. In the architecture evaluation process there are two common steps to gather and refine scenarios (similar to the Fagan inspection process [16][17], another QA technique): 1. individual scenario generation and 2. team scenario generation based on individual scenarios in a team meeting to improve the quality (and possibly the quantity) of scenarios for architecture evaluation. The accuracy of the results of the above-mentioned architecture evaluation methods is largely dependent on the quality of the scenarios used for the SA evaluation as these are scenario-based methods [25].

Recent research in the area of software inspection identified experience of inspectors and reviewers as a success-critical factor in QA. Parnas and Weiss [28] regard the presence of wrong people in the design review sessions as the one of the major problems with conventional design review approaches. Clements *et al.* suggest to have stakeholders actively participate, that includes getting them fully integrated within the evaluation sessions [15]. During our academic lectures and professional training seminars, we have also observed that people with more industrial experience usually perform better on exercises and understanding the theoretical concepts underpinning architecture evaluation. Hence, we postulate that stakeholders with extensive experience will achieve better results, i.e., identify most important defects in inspections and important scenarios in architecture evaluation. However, highly experienced stakeholders are often expensive and/or unavailable. Thus, a general tendency of project managers is to include less experienced (and cheaper) stakeholders but still achieve good results from architecture evaluation. Common options are (a) to support architecture evaluation actively during the evaluation process [33], e.g., by providing evaluators with a basic set of scenario categories, and (b) to apply teams to achieve synergy effects and find additional scenarios [12].

Inspection research literature reported that nominal teams [2], i.e., non-communicating teams, do not require a team meeting but provide benefits from synergy effects (collecting individual scenario lists into a common team list) [12]. The effectiveness of inspection team meetings and nominal teams has been a topic of significant research over the years in the software inspection area (e.g. [12]); the effect of team meetings on the quality of scenarios generated in Quality Attribute Workshops' (QAWs [6]) team meetings during the software architecture evaluation process has not yet been empirically investigated. A secondary question is whether or not a structured technique for eliciting scenario profiles can improve the performance of architecture evaluation (compared to unguided scenario elicitation), measured as the quality score of scenario profile (i.e., a set of scenarios) developed by an architecture evaluation team. A third question covers the categorization and classification of scenarios, i.e. whether expert scoring of scenarios is necessary for prioritization of requirements and scenarios, or if frequency-based

scenario ranking scheme (based the frequency of occurrence of each scenario in scenario profiles of individual as well as real teams) is reasonable for scenario prioritization and classification.

This paper presents results from a controlled experiment [1] to explore the above-mentioned different aspects of developing quality sensitive scenarios for architecture evaluation. Main goal of the study is to characterize quality attributes required by a system regarding the effectiveness of experience and team work in a software architecture evaluation process with respect to different scenario development techniques (top-down and bottom-up) on the quality of scenario profiles. These scenario profiles are developed by a team of stakeholders.

The remainder of this paper is structured as follows. Section 2 provides a brief overview of related work and motivation for our research on different aspects of scenario developed activity in the software architecture evaluation process. Section 3 introduces the research issues and hypotheses. Section 4 summarizes the experiment performed to gather the empirical data and Section 5 presents the results. Section 6 concludes the papers with suggestions for further research work.

2. RELATED WORK

This section summarizes relevant research from the software architecture and software inspection disciplines.

2.1 Architecture Evaluation

Software quality attributes of a software system such as performance, security, or changeability can be supported or inhibited by the software architecture (SA) of a software-intensive system [8]. Thus the evaluation of SA at an early stage has gained significant interest from researchers and practitioners. Software engineering community have developed several approaches to evaluating architectures. Most of the mature and established architectural evaluation methods are scenario-based [4]. Scenarios provide context for evaluating architecture to work with concrete examples enabling the user to understand their detailed effect [25][27]. The accuracy of the results of scenario-based methods is largely dependent on the quality of the scenarios used for evaluating architecture [25]. A set of scenarios is called a scenario profile. The SA community has developed many frameworks for eliciting, structuring, and classifying scenarios. For example, Lassing *et al.* [26] proposed a two-dimensional framework for eliciting scenarios, Kazman *et al.* [22] proposed a generic 3-dimensional matrix to elicit and document scenarios. Bass *et al.* [8] provided a six-element framework to refine and structure scenarios. The Software Engineering Institute (SEI) has enumerated a collection of general quality-attribute scenarios that are intended to help characterize most commonly known quality attributes [9] such as performance, modifiability, and usability. A general scenario is, in effect, a template for generating a specific quality-attribute scenario. For example, two (abbreviated) modifiability general scenarios are: (a) Changes to the platform occur and (b) System needs to serve requests arrived from users.

Since not all general scenarios for a particular quality attribute may be relevant to a particular system or class of systems, an evaluator must identify relevant scenarios that should be considered and make these scenarios system specific [1]. We believe that general scenarios can also help instigate thinking for developing system-specific scenarios called concrete scenarios. However, general scenarios can also be classified according to domain-specific software change categories to help an evaluator to identify those general scenarios that may be more relevant and should be made system specific with the help of stakeholders for a particular system.

The research effort reported in this paper is motivated by the practical need to empirically determine and understand different aspects of the scenario development activity (one of the most important activities) of the software architecture evaluation process. As it has been reported that the accuracy of the results of scenario-based architecture evaluation methods is largely dependent on the quality of the scenarios used in the process [25], we assert that there is vital need for developing and empirically assessing more effective approaches to improving the scenario development process to generate better quality scenarios.

2.2 Software Inspection

Similar to the architecture evaluation, software inspection also aims at assessing the quality of artefacts in the software development process. Some software inspection approaches also use scenarios to support finding quality issues [20]. While the results of the software inspection process are defect reports, the activity of the architecture evaluation process (i.e., scenario gathering), we investigate in this study results in a list of quality attribute scenarios to be used for evaluating software architecture. We have observed that the general process of architecture evaluation seems to be sufficiently similar to consider drawing experiences from software inspection to generate hypotheses for studying different aspects of the architecture evaluation process.

Initially, Fagan's software inspection [16] viewed the inspection team meeting as the key process step, while more recent approaches focus more on individual inspection work to lower inspection effort as the team meeting is much more expensive than individual work. Overall, empirical results on software inspection meetings' effectiveness differ considerably. Fagan reported inspection meetings to be very effective [16][17], while more recent studies reported contradictory results [18][29][30][31]. Software inspection researchers have identified several potential benefits of meetings:

1. *Synergy*: The synergy effect assumes that meeting dynamics help find new defects. However, Votta [31] observed in his study that individual inspectors had already recorded 8 out of 9 defects that came out of the team meeting and thus only very little synergy can be achieved. Bianchi *et al.* [11] report that meeting losses (i.e., real defects found during individual preparation but then dismissed in the meeting as irrelevant or false positives) significantly outweigh meeting gains (i.e., defects newly found during the team meeting). Johnson and Tjahjono [18][19] observed a substantial degree of synergy (30% to 40% of new defects detected).

2. *Identification of False Positives*: Land *et al.* [23] report that team meetings have a clear advantage over individual defect detection in discriminating between true defects and false positives.
3. *Soft Benefits* are meeting benefits apart from synergy and false positive reduction, including the sharing of review experiences, the dissemination of product-related knowledge and the creation of collective ownership for the review outcome among reviewers [18][19]. However, in software inspections the reduction of false positives and the soft benefits do not seem to justify the meeting costs.

We believe that all these aspects are quite important to be explored in the context of software architecture evaluation, especially, any potential impact of the experiences of the participants, team size, and the use of nominal teams.

2.3 Participant experience

Experience of inspectors and reviewers is a success-critical factor in QA. Typically, extensively experienced engineers will achieve better results, e.g., a higher number of more important and critical defects (inspection) and important scenarios (architecture review). Active guidance of reviewers / inspectors during the review process will support individuals and lead to an increased number of identified defects/scenarios, higher effectiveness and efficiency [33]. Nevertheless, review performance depends on individual experience. Applying teams will achieve synergy effects, deliver additional defects/scenarios (real teams or nominal teams) but may also loose defects/scenarios (real teams) [12] that were reported by individuals. While there are several reports on participant experience and team effects in QA, e.g., in software inspection, an investigation on the likely effect size of teams on scenario generation in architecture evaluation remains open.

3. RESEARCH HYPOTHESIS AND VARIABLES

The context for this experiment is a scenario development workshop (such as Quality Attribute Workshop (QAW) [6][27]), where stakeholders develop scenarios to precisely specify quality attributes (i.e., non-functional requirements). These scenarios are used to assess the capability of an architecture with regards to the desired quality attributes characterized by these scenarios [8]. According to Bengtsson and Bosch, we applied a two-stage process of scenario development for architecture evaluation [10]: 1. each individual develops scenarios alone, and 2. individuals develop team scenarios based on the individual scenarios and discussions in team meetings. We expect that those individuals who (a) are more experienced and (b) apply a guided technique for scenario generation should deliver a larger proportion of the most relevant scenarios. Our approach to measuring the quality of developed scenarios is designed to capture this aspect of scenario quality.

In SA the importance of identified scenarios is success-critical. Thus, we can apply a scoring experience, which represents the individual scenario importance. Scenario prioritization can be either performed by experts (expert prioritization), which will result in additional effort for categorization and classification purposes, or scenario priorities can be based on the frequency of occurrence of each scenario in the individual as well as team scenario profiles (frequency-based approach [5]). For empirical evaluation, we investigate the performance of individuals and teams of varies sizes (nominal teams) to find scenarios in three categories: critical (very important, class A) scenarios, important (class B) scenarios, and less important (class C) scenarios. Critical scenarios have a major impact on business value while less important scenarios can be classified as “nice-to-have”. Scenario prioritization can be an effort-consuming activity, if performed by experts. Thus, we apply two different scenario prioritization approaches to increase scenario prioritization performance: (a) Scenario classification based on scoring (SC-S) as above-mentioned frequency-based approach, and (b) scenario classification of experts (SC-E) based on domain knowledge and business impact:

a) *SC-S: Scenario prioritization* rates the importance of scenarios based on the overall number of scenarios, sorted by frequency from a (large) group of participants. The top 20% of scenarios were classified as critical (very important) scenarios, 40% were important, and 40% were less important; as we assume that the top 20% of scenarios have the highest impact on business value.
b) *SC-E: Expert classification of scenarios* applies domain knowledge and the impact on business values to identify the most important scenarios. A member of the experiment team conducted this expert classification.

Scoring is important to focus the evaluation work with scenarios on the most worthwhile scenarios, e.g., for intensive discussion. Thus, it seems important to investigate, whether different scoring schemas reliably identify the most important scenarios. Table 1 gives an overview of two different scenario classification schemes and the corresponding ranges with respect to class assignment.

TABLE 1: SCENARIO CATEGORIZATION SCHEMA

	Scenario Scoring (SC-S)		Expert Scoring (SC-E)	
	No. of Scenarios	Share of Scenarios	No. of Scenarios	Share of Scenarios
Critical (Class A)	22	21%	19	18%
Important (Class B)	41	39%	60	58%
Less Important (Class C)	41	39%	25	24%
Total	104	100%	104	100%

The individuals use either a structured and guided approach (also called top-down) to develop scenarios instructed by so-called change categories (treatment group) or an ad-hoc, unguided approach (also called bottom-up). The latter represents the control group. (see [1] for detailed information about the software change categories used to

guide the scenario development process of the treatment group in this research). Performance can be measured for individuals as well as for nominal teams in each of the treatment groups. There can be either real teams, which conduct a team meeting or nominal teams, which do not meet. This study applies nominal teams. The scenarios list for a nominal team is directly derived from the individual scenario lists of the members of the nominal team.

Variable Definition

- *Independent variables* of the study are: a) a list of domain-specific categories of software changes provided to the participants, b) prior qualification of participants, and c) size of non-communicating teams. One half of the participants received these scenario categories (treatment group) the other half of the participants did not use these scenario categories (control group). Participant experience was collected prior to the individual scenario development phase, based on their software development experience applying a background questionnaire. Team size was varied during data analysis.
- The *dependent variable* is the quality of scenario profiles developed by the participants individually, and in nominal teams (i.e., a non-communicating team) of 2 to 8 person teams. We measure individual (and team) performance based on the number of identified scenarios per scenario class and classification schema (see table 1 for details). In this paper we follow three main research goals: In a first step we want to assess the *impact of individual experiences of participants on generated scenarios during architecture* evaluation workshops. In a second step we focus on *different scenario scoring approaches* and in a third step we want to investigate the impact of *team-size of nominal teams*.

We propose the following null hypotheses for the reported research:

- *H01*: Extensive and less experienced participants will identify a similar number of scenarios.
- *H02*: The number of identified critical scenarios is similar for expert ranked and frequency related scores.
- *H03*: The number of identified scenarios is similar for all nominal teams independent of team size.

The alternative hypotheses for research are:

- *H11: Impact of participant experience*: Participants with extensive experience find more and more important scenarios than participants with less experience.
- *H12: Scenario scoring schemes*: The scenario scoring schemes provide different results for critical results, i.e., results need to be discussed separately for both scoring methods.
- *H13: Team Size*: Larger teams find more and more important scenarios.

For data collection regarding these research hypotheses, we conducted a controlled experiment in an academic environment as described in the following sections.

4. EXPERIMENT DESCRIPTION

This section provides an overview on the experiment design, conduct, and threats to validity.

4.1 Experimental Design

The experiment design was a randomized balanced design, which used the same experimental materials for both treatments and assigned the subjects randomly to each treatment [34]. The assignment of individuals to the treatment groups and real teams was randomized using a sort card method of randomization; there were 12 participants in each of the treatment groups (and 4 teams of 3 persons in the team part) of the experiment.

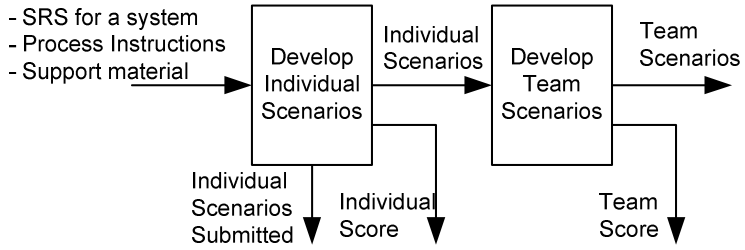


FIGURE 1: CONTROL FLOW DIAGRAM OF THE EXPERIMENT PROCESS.

Figure 1 shows the major steps of the experiment process: Based on system requirements (functional and non-functional) and process instructions the participants delivered a list of developed scenarios in the first step. After developing the individual scenarios, each participant joins his/her respective team to develop a group scenario profile (i.e., a set of scenarios) based on their individual scenarios. Based on the discussion within the teams net gains (additional scenarios) and net losses (removing scenarios) are possible due to interaction. In this paper we applied nominal teams (i.e. non-communicating teams) of different sizes [3] rather real teams. Hence, for this research the second part of the process – shown in Figure 1 – is for data analysis purposes only, where we build team scenario profiles of varying sizes of team (2 to 8) based on the scenarios of the individuals assigned to each formed team.

4.2 Experimental Context and Subjects

The 24 participants of this study were recruited from a software architecture course offered at the University of New South Wales, Australia. The experiment was part of a scenario development workshop, which was one of the assessment tasks. The students were briefed about the objective and procedure of the study. They had the option of withholding their results from research. Written permission was sought from the participants to use their data in this study. Most of the students were post-graduate students with the exception of 3 fourth-year undergraduate students. There were only 5 female students out of 24 participants in this study. All of the participants were either working or had worked as information technology (IT) professionals with an average working experience of 4.5 years in the IT industry and were of an average age of 27 years. Table 2 shows the distribution of participants regarding their individual experience and their group assignment (full randomization of group assignment).

TABLE 2: EXPERIENCE AND GROUP ASSIGNMENT OF INDIVIDUALS

Individual	Control Group		Treatment Group		Total	
Experience	No	Share	No	Share	No	Share
Extensive	5	21%	7	29%	12	50%
Less Experience	7	29%	5	21%	12	50%
Total	12	50%	12	50%	24	100%

4.3 Experimental material

Two lectures (2 hours each) were dedicated to the topics directly related to the experimental study, i.e., quality attributes, software architecture evaluation, and approaches to brainstorm and structure general and concrete scenarios in order to characterize quality attributes. During the course, there was also one class exercise to brainstorm and structure scenarios for a system the students were familiar with. One week before the study, all the participants received detailed information about the system, LiveNet [14], for which they were supposed to develop software change scenarios. A short document describing various features of LiveNet was also provided a week before the study. Before the study all the participants attended a 30 minutes refresher session briefly covering the lectures' topics as mentioned above. However, our study did not require the participants to have any experience in developing scenarios for evaluating software architecture, which is also the case in industrial scenario development workshops where stakeholders normally receive minimum training in creating scenarios. The most important elements in experiment materials and instruments are: the systems requirements specification; software change categories; and consistently measuring the quality of the resulting scenarios.

4.3.1 Software requirements specifications

This study used the Software Requirement Specification (SRS) for a web-based collaborative tool, LiveNet [14]. LiveNet provides a generic workflow engine and features to support collaboration among geographically distributed members of a team, e.g., synchronous chat, discussion forum, document repository, notification, roles, planning tools, and task assignment tool. LiveNet enables users to create workspaces and define elements of a particular workspace. LiveNet also supports emergent business processes. We prepared a simplified version of an SRS and a description of the system to provide the participants with a clear picture of the system and its requirements.

4.3.2 Approaches for scoring the scenarios and justification for it.

We applied two different scenario categorization schemes to investigate the impact of scenario importance and to identify the most effective categorization and classification approach: (a) scenario prioritization based on brainstorming frequency (SC-S) and (b) expert classification of scenarios (SC-E). SC-S uses the scenario identification frequency by all participants, i.e., counting the number of scenarios identified by individuals and real teams per individual scenario. Result is a prioritized list of scenarios according to their frequency. We applied a 20%/40%/40% distribution, to focus on the top 20% scenarios (critical and very important) with the highest impact on business value, 40% of important scenarios and 40% less important scenarios (nice-to-have scenarios). This scoring approach is easily applicable based on the individual lists of identified scenarios. On the other hand, expert prioritized (SC-E) scenarios (without predefined share of scenario classes) might deliver different results, e.g., because of domain knowledge and might be more practical. Nevertheless, this scoring approach requires additional effort by experts. Table 1 shows scenario category distributions of both scenario classification schemes used for scoring the scenarios for individuals and nominal teams.

4.4 Experiment Execution

We recruited the participants for this study from postgraduate course in software architecture as mentioned in section 4.2. Since we needed to have real teams for following the two-stage process and for determining the frequency of occurrence of each scenario reported by individual and team, for the second stage of the developing scenarios, we decided to form teams of 3 participants. Prior to the experiment, we form 8 teams based on the expected number of the participants. Each team was assigned a name and three participants were assigned to each team. Each team was allocated to one of the two experimental conditions (treatment and control). The composition of teams and the assignment of the participants to experiment conditions was randomized using a card sort randomization method. After the briefing session of 30 minutes, the participants were given a simplified version of requirements for LiveNet. The participants in the treatment group also received a document describing the five categories of most commonly occurring changes in a collaborative application like LiveNet. They were

encouraged to use the categories to stimulate their thinking about the types of changes that may be expected to occur over the coming three years during scenario development exercise.

The participants were asked to develop software change scenarios individually for 35 minutes. When 35 minutes of time had passed the profile of individuals were collected, photocopied and returned to them. All the participants were asked to join their respective teams to develop team scenarios for 40 minutes. Once 40 minutes of time had elapsed, the team (real teams') scenario profiles were collected. The experiment finished with a debriefing session in which the participants also filled a post-session questionnaire. Information collected during the experiment was tracked by an identification code present on the individual scenario profiles, team scenario profiles, and post-session questionnaire. Thus, the data is not anonymous. We collected four sets of data: the demographic data about each participant, the individual scenario profiles, the team scenario profiles and the questionnaire filled by treatment group participants.

4.5 Validity considerations

Every empirical study has to deal with several threats to internal and external validity; in this sub-section we discuss the major threats in our context and the countermeasures we applied.

4.5.1 Threats to internal validity

Internal validity is the degree to which the values of dependent variables can only be attributed to the experimental variables [34]. In order to avoid bias in allocating participants to the treatment groups, we randomized the assignment by using a sort card method. We wrote the names of the participants and groups on plain cards. After shuffling the cards, we assigned one card to each group (treatment and control) without seeing the individual's or group's name on the card.

Another threat to the internal validity of our experiment is the scoring schemes used for scenarios. We applied two scoring schemes. SC-S (scenario scoring) is based on the frequency of identified scenarios. Because there is a high dependency on individual results (and experience of participants) the results may differ according to the contexts (e.g., industrial/academic context, participant experience variation). To address this issue, we collected the individual experience of participants prior to the study using a questionnaire. The second scoring schema uses a categorization and prioritization of scenarios by an expert (not any of the authors), who was familiar with the basic requirements of LiveNet and (b) an expert in requirements and scenario elicitation.

4.5.2 Threats to external validity

External validity is the degree to which the results can be generalized, i.e. transferable to other similar situations. In particular, it is important to consider whether the participants are representative of the stakeholders who would undertake architecture evaluation in the industry, and whether the experimental materials and process are representatives of the process and materials used in industrial architecture evaluations. In an industrial evaluation, stakeholders may have a variety of different backgrounds, e.g., software engineering, marketing, management, sales. This was not the case in our experiment. All the participants had educational and professional backgrounds in either computer science or software engineering. Nevertheless, the difference of experience might not be large enough to make a significant impact. This means that our results are more likely to generalize to stakeholders with a technical background than stakeholders with a non-technical background.

Secondly, stakeholders in an industrial situation are more likely to have considerable experience of the application being evaluated, whereas the participants in our experiment only had limited knowledge of LiveNet. That means our results are most likely to apply to stakeholders with limited experience of application being evaluated. The participants had limited experience of software architecture evaluation and of developing scenarios for quality attributes. As far as we are aware, organizations normally do not provide extensive training to their employees for software architecture evaluation or developing quality-sensitive scenarios [1]. Thus, the experience of the experimental participants is likely to be similar to that of stakeholders performing an industrial evaluation. The software requirements specifications used in the experiment is relatively short and simple compared with a typical industrial one. However, in industry stakeholders would be given both more requirements and a more time to develop their scenarios. Finally, there may be a threat to the external validity if the scenario development process used in our study is not representative of the industrial practices for developing scenario profiles for software architecture evaluation. However, the scenario development process in this experiment was similar to the one used for in brainstorming workshops for gathering quality sensitive scenarios like QAW [6].

5. RESULTS

The following sub-sections provide the findings from analysing the data collected during the experiment.

5.1 Data Analysis procedure

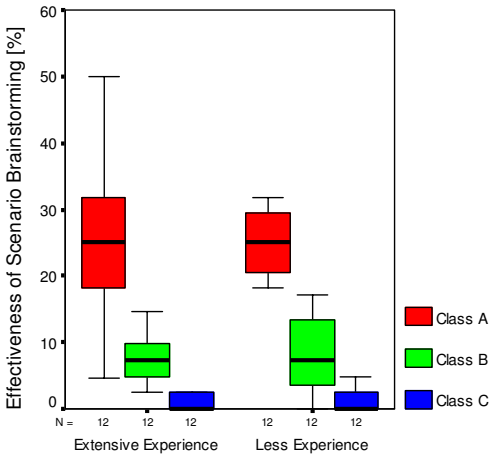
The data analysis takes as inputs the scenario profiles coming out of the experiments steps pictured in Figure 1. Additionally, we captured project experience as indicator for participant experience. As preparation for statistical evaluation, the identified scenarios were matched and divided into three scenario categories: (class A) critical and very important scenarios, (class B) important scenarios, and (class C) less important and minor scenarios. To investigate the impact of frequency and expert scenario classification, we applied two scenario classification schemes: (a) scoring based on the frequency of identified scenarios following a 20/40/40 distribution of scenarios (SC-S) and (b) scoring based on expert prioritization (SC-E). Regarding the impact of different sizes of nominal teams we conducted full combinations of all participants of a group (treatment group (TG) and control group (CG)).

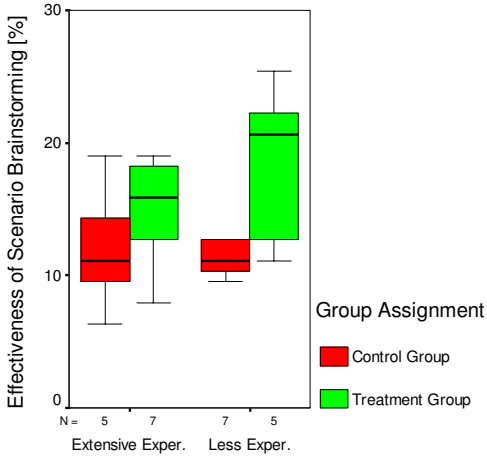
We did not mix groups. We applied the Mann-Whitney-Test at a significance level of 95% (2-tailed) to test the hypotheses [2]. In the following sub-sections we provide an overview of key descriptive statistical data of the effects of independent variables.

5.2 Individual Scenario Brainstorming Effectiveness and Participant Experience

Effectiveness is defined as the number of scenarios identified by an individual in relation to the overall number of scenarios per scenario category and individuals. Figure 2 presents the results of individual scenario brainstorming effectiveness for the frequency-based scenario scoring approach (SC-S). Figure 2a shows the distribution of effectiveness regarding the scenario classes A, B, and C (see table 1) and experience classes (see table 2). We observed the highest effectiveness for critical / very important scenarios and lowest effectiveness for less important scenarios. However, the average effectiveness for higher and less experienced participants is similar. Applying the Mann-Whitney-Test at a significance level of 95% (2-tailed), we could not observe significant differences.

Figure 2b presents the results of the different treatments and participant experience: control group without guidance by scenario categories and treatment group including given scenario categories. Analyzing the results we observed benefits for the treatment group for both, extensive and lessexperienced participants. Regarding mean value of the control group we observed similar results regarding high and low experienced participants. In contrast, the comparison of the mean values of the treatment group showed advantages for less qualified participants. The scenario generation method (scenario categories provided (treatment group) / not provided (control group)) seems to be more important than individual experience in the context of the study. We could not observe significant differences between high and low experienced participants (control group, p-value: 0.869(-) and treatment group, p-value: 0.287(-)). Nevertheless, we observed that the top-down scenario generation method (treatment group) supported participants with less experience better than participants with higher experience. The results confirm our hypothesis H01 that higher and less experienced participants identify a similar number of scenarios.





	Extensive Experience		Less Experience	
	Mean	SD	Mean	SD
Class A	25.4	11.70	25.0	11.55
Class B	7.5	3.67	8.3	5.83
Class C	1.6	2.82	1.0	1.63

2a: Experience and Scenario Effectiveness.

	Extensive Experience		Less Experience	
	Mean	SD	Mean	SD
Control	12.1	4.84	11.1	1.83
Treatment	15.0	4.08	18.4	6.21

2b: Experience/Treatment and Scenario Effectiveness.

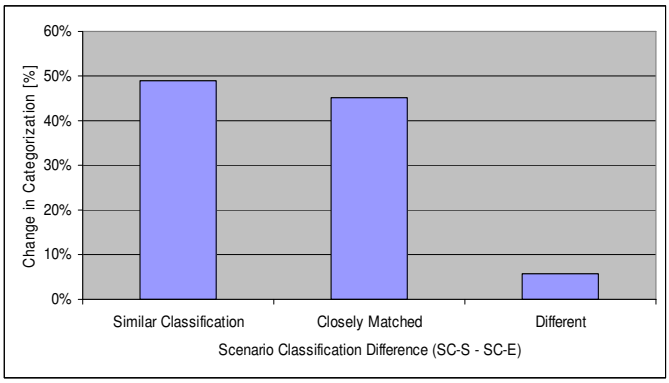
FIGURE 2: INDIVIDUAL EFFECTIVENESS OF SCENARIO BRAINSTORMING.

5.3 Comparison of Scenario Scoring and Expert Scoring

To identify an effective and efficient scenario classification schema according to the importance of scenarios, we introduced two scenario categorization techniques: (a) frequency based (SC-S) and (b) expert based scenario (SC-E) prioritization (see section 3). For evaluation purposes we focus on more important scenarios (i.e., critical (class A) and important scenarios (class B) as these have a major impact on product quality and they should be addressed first in the architecture evaluation process. Table 1 provides the number of assigned scenarios per scenario class and classification schema. The results of this evaluation identified a similar number of important scenarios (A+B scenarios) independent of the participant experience. Note, that the distribution of categories varies (20:40:40 for Score and 20:60:20 for Experts), because we did not provide a range for expert classification.

A first analysis of scenario class assignment in both scoring schemes (SC-S, SC-E) showed that more than 94% of the scenarios are similar, (49%) or very closely matched, i.e., difference of one scenario class (45%). In a more detailed analysis we summarized different/equal scenario classifications of both scoring approaches (SC-S and SC-E) per identified scenario in figure 3. *Similar classification* includes no change of the classification in SC-S and SC-E), *Closely matched* summarizes a minor different classification in one of the scoring schemes. i.e., A-B, B-C and vice versa. *Different* means a major modification of scenario classification from A to C or vice versa. In a third step we investigated, how many critical (most important) scenarios were classified identically in both scoring schemes: To investigate the deviation of critical (class A) scenarios, we calculated the overall number of scenarios, which were classified as “critical” in at least one scoring schema (SC-S or SC-E). The result showed a coverage

73% of class A scenarios of SC-S and 63% of SC-S. 37% of critical scenarios were classified similar in SC-S and SC-E. Applying a chi-square test on the individual distributions showed a p-value of 0.011 for frequency base scoring and 0.144 for expert scoring. The p-value for non-critical scenarios was <0.001. Following these results, the scoring approach based on frequency (SC-S) seems to be an appropriate alternative for expert scoring. Nevertheless, a more detailed investigation is necessary to verify these results.



	Extensive Experience		Less Experience	
	Mean	SD	Mean	SD
Score	13.8	4.46	14.2	5.48
Expert	9.7	3.34	9.9	2.90

3a. Mean and Std.Dev.

	Score	Expert
p-value	0.930(-)	1.000(-)

3b. P-Value (Mann-Whitney-Test at 95% (2-tailed).

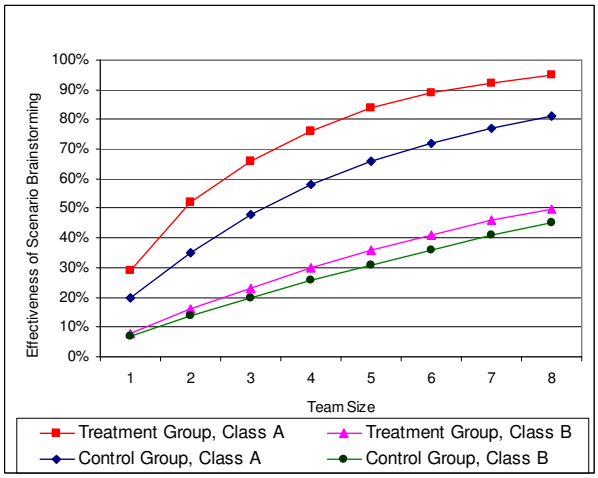
FIGURE 3: CHANGE IN SCENARIO CLASSIFICATION.

An analysis on individual effectiveness in the scoring schemes (i.e., SC-S and SC-E), we found: SC-S results in a higher score than expert scoring due to the added score for frequent occurring scenarios in SC-S. However, the difference comes mostly from the calculation differences and only to a very small extent from the differences in distributions of scenario classes A, B, and C. For the evaluation of scenario generation effectiveness the two scoring schemas provide very similar results and thus we can present all evaluation results according to the SC-S scoring schema. The results confirm H02 that the number of critical scenarios is comparable for both scoring schemes.

5.4 Impact of Team-Size on Brainstorming Effectiveness

Based on individual scenario brainstorming activities, experience of teams ("involvement of more brains") can further support scenario identification. Nominal teams, i.e., non-communicating teams, add the scenarios from all participants within a nominal team. Thus the higher the variety in the team and the lower the overlap of similar scenarios is the better for team performance. An important issue is determining the effect of adding more participants to a team in order to better understand the potential costs and benefits of smaller and larger teams.

For data analysis we used the individual scenario lists and applied full combination for every team size from 1 (individual) to 8 team members to investigate increasing effectiveness. As the scenario generation approach was found to be more important than individual participant experience in this study (see section 5.1), we investigated in detail the influence of team size and scenario generation approach on the effectiveness to find critical (class A) and important (class B) scenarios.



Mean	Team Size / Number of Team Members							
	1	2	3	4	5	6	7	8
TG, Class A	29%	52%	66%	76%	84%	89%	92%	95%
CG, Class A	20%	35%	48%	58%	66%	72%	77%	81%
TG, Class B	8%	16%	23%	30%	36%	41%	46%	50%
CG, Class B	7%	14%	20%	26%	31%	36%	41%	45%

4a. Mean Value.

Std.Dev	Team Size / Number of Team Members							
	1	2	3	4	5	6	7	8
TG, Class A	12%	11%	10%	9%	8%	7%	6%	5%
CG, Class A	8%	10%	10%	9%	8%	7%	7%	6%
TG, Class B	4%	6%	6%	6%	5%	5%	5%	4%
CG, Class B	4%	6%	7%	7%	7%	7%	6%	6%

4b. Standard Deviation.

FIGURE 4: SCENARIO BRAINSTORMING EFFECTIVENESS AND TEAM SIZE.

Figure 4 presents the results: above 3 participants, every additional team member increases scenario brainstorming effectiveness less than 10% increase (higher rates below 3 team members). Thus, a 2-3 person team seems economically advisable. Nevertheless, larger teams might be useful in case of more critical requirements and application domains, since there is a continuous increasing effectiveness up to >90% for the treatment group for class A (critical and most important scenarios). We applied the Mann-Whitney at a significance level of 95% (2-tailed) to test our hypothesis regarding the effectiveness of an increasing team size for the treatment and control group and critical (class A) and important (class B) scenarios. We observed significant difference for an increasing team size the treatment and control groups and scenario classes A and B

(p-value < 0.001(s) each). Thus, the results do not support the hypothesis H0.3 that the number of identified scenarios is similar independent of the team size.

The difference between the control group (no guidance) and treatment group (guidance with scenario categories) of class A (critical) scenarios is some 15% higher for the treatment group for teams >2 members. The advantage of the treatment group for class B (important) scenarios is below 10% for all team sizes (from 2 to 8 members). It should be noted that the participants using the top-down method (treatment group, guidance with scenario categories) is 10-20% higher for critical scenarios than for teams applying the bottom up method (control group without guidance). Overall in this study the impact of using a better technique in a team was as effective as having a considerably larger team without a guiding technique.

6. DISCUSSION AND FURTHER WORK

This section summarizes the hypotheses related to the research questions and spot on directions for further work.

Impact of participant experience. Based on the claims made by Parnas *et al.*[34] that the presence of wrong people in design review sessions is one of the major problems with conventional design review approaches, and Clements' suggestions [15] for having actively involved stakeholders for architecture evaluation that means getting them fully prepared for the session, we postulated that the stakeholders need to have an appropriate experience level for successfully conducting architecture evaluation including developing high quality scenarios. We expected that participants with higher experience find more and especially more important scenarios than participants with less experience: To achieve comparability between different scenario classes we introduced scenario brainstorming effectiveness, i.e., the number of identified individual scenarios per scenario class. In the context of this study, both experience groups showed the highest effectiveness for critical and very important scenarios and lowest effectiveness for less important scenarios. Furthermore, the effectiveness for higher and less experienced participants was found to be similar. An explanation may be that the differences of experience in the study context were not large enough to make a significant impact (i.e., rather homogeneous participant groups). Thus we suggest for future empirical studies to consider participants with more variety of backgrounds (e.g., different architectural backgrounds, usage-oriented rather than technical backgrounds). Additionally, we investigated the impact of individual experience with respect to different treatments (guided vs. unguided techniques for developing scenarios [1]). We observed on average considerably stronger benefits for the treatment group for both higher and less experienced participants. The comparison of the mean values showed advantages from the guided technique for less qualified participants. Thus an initial result was that participants with less experience benefit more from the experience "built into" a guided technique than workshop participants who are more experienced.

Scenario scoring schemes. Scenario scoring is a well-established approach for classification and prioritization of scenarios and requirements in a software project to focus on the most important scenarios with the highest impact on business value. We introduced two different scoring approaches: (a) scenario prioritization based on the frequency of each scenario brainstormed by individuals and real teams (SC-S) and (b) expert classification of scenarios (SC-E) to find out how well the results of both scoring schemes match. The SC-S schema is based on the frequency of scenarios elicited by this study's participants who had varying level of experience in generating quality attribute scenarios; while the SC-E schema is based on one highly experienced individual who rated the generated scenarios independent of the researchers (i.e., authors).

The scenario scoring schemes provide similar results for data analysis regarding the critical (most important) scenarios, which is a reassuring result for the reliability of both scoring methods. An interesting observation of SC-E was that 60% of all scenarios were rated "important" which may not provide enough guidance for practical work. Thus further work will be to investigate rating of groups of experts with the goal of providing reliable and more focused prioritization of scenarios.

Team size. Based on individual scenario brainstorming activities, experience of teams (indicated in the size of a real and/or nominal team) can deliver additional scenarios. The cost and benefits of different sizes of teams for evaluating architecture is an important issue [2], which requires empirical studies.

In this study, we used nominal teams of different sizes to study this aspect, the results show that larger teams found both more and more important scenarios, i.e., they were significantly more effective in finding class A than class B scenarios. The results showed that for more than 3 team members, every additional team member increases scenario effectiveness for less than 10%. Larger teams found on average more scenarios on each level of importance (different scenario classes). The results showed that 2-3 team members per team seem to be economically advisable as the smaller increase of additional scenarios by involving additional team members may not be cost efficiently. Nevertheless, in critical requirements, scenarios, and applications domains, it might be reasonable to apply architecture evaluation with up to 6 persons. Regarding larger teams, diverse background of experience is advisable to improve the chances for more and better scenarios. Such role-based scenario brainstorming techniques might be an aspect for future discussions [7].

Further work is to investigate the impact of different scenario scoring methods in more detail, particularly for important scenarios (experts rated about 60% of the scenarios as important). Another future direction is the improvement of team work, i.e. how to make team work in scenario elicitation more efficient and effective. Tools for the support of scenario generation, discussion and voting (similar to the EasyWinWin-based [13] tool support for inspection meeting) might be a promising approach for application in architecture evaluation as they have been successfully used to improve both the effectiveness and efficiency of contribution elicitation, even in very large teams in software inspections [13].

REFERENCES

[1] Ali-Babar M. and Biffl S. (2006): *Eliciting Better Quality Architecture Evaluation Scenarios: A Controlled Experiment on Top-Down vs. Bottom-Up*. Proc. of the Int. Symposium on Empirical Software Engineering.

[2] Biffl S., Winkler D., and Ali-Babar M. (2008): *An Empirical Study of Scenarios Gained and Lost in Architecture Evaluation Meetings*, submitted to the 2nd Int. Symp. on Empirical Software Engineering and Measurement.

[3] Ali-Babar M. and Kitchenham B. (2007): *The Impact of Group Size on Software Architecture Evaluation: A Controlled Experiment*, 1st Int. Symp. on Empirical Software Eng. and Measurement (ESEM), Madrid, Spain.

[4] Ali-Babar M., Zhu L., and Jeffery R. (2004): *A Framework for Classifying and Comparing Software Architecture Evaluation Methods*. 15th Australian Software Engineering Conference.

[5] Ali-Babar M, Kitchenham B, Jeffery R. (2008): *Comparing distributed and face-to-face meetings for software architecture evaluation: A controlled Experiment*, Empirical Software Engineering Journal, 13(1): pp. 39-62.

[6] Barbacci M.R, et al. (2003): *Quality Attribute Workshops (QAWs)*, Tech Report CMU/SEI-2003-TR-016, SEI, Carnegie Mellon University, USA.

[7] Basili V.R., Green S., Laitenberger O., Lanubile F., Shull F., Soerumgaard S., and Zelkowitz M. (1996): *The Empirical Investigation of Perspective-Based Reading*. EMSE J, 1, 2, pp. 133-164.

[8] Bass L., Clements P. and Kazman, R. (2003): *Software Architecture in Practice*. Addison-Wesley.

[9] Bass L., Klein M. and Moreno G. (2001): *Applicability of General Scenarios to the Architecture Tradeoff Analysis Method*. Tech. Report CMU/SEI-2000-TR-014, Software Engineering Institute, Carnegie Mellon Univ.

[10] Bengtsson P. and Bosch J. (2000): *An Experiment on Creating Scenario Profiles for Software Change*. *Annals of Software Engineering*, *9*, pp. 59-78.

[11] Bianchi A., Lanubile F., Visaggio G. (2001): *A Controlled Experiment to Assess the Effectiveness of Inspection Meetings*, Proc. Metrics 01, London.

[12] Biffl S., and Halling M. (2001): *Investigating the defect detection effectiveness and cost benefit of nominal inspection teams*, IEEE Transactions on Software Engineering, Vol. 23, Issue 5, p385-397.

[13] Biffl S., Grünbacher P., and Halling M. (2006): *A Family of Experiments to Investigate the Effects of Groupware for Software Inspection*, Journal of Automated Software Engineering, Vol. 13/3, p373-394.

[14] Biuk-Aghai R.P. and Hawryszkiewyez I.T. (1999): *Analysis of Virtual Workspaces*. *Proceedings of the Database Applications in Non-Traditional Environments*.

[15] Clements, P., Kazman, R., Klein, M. (2002): *Evaluating Software Architectures: Methods and Case Studies*, Addison-Wesley.

[16] Fagan, M. (1976): *Design and Code Inspections To Reduce Errors In Program Development*, IBM Systems J., vol. 15, no. 3, pp. 182-211

[17] Gilb T., and Graham D. (1993): *Software Inspection*, Addison-Wesley.

[18] Johnson P. M., Tjahjono D. (1997): *Assessing software review meetings: A controlled experimental study using CSRS*, Proc. ICSE 97, Boston.

[19] Johnson P.M., Tjahjono D. (1998): *Does Every Inspection Really Need a Meeting*, Emp. Software Eng.

[20] Laitenberger, O., DeBaud, J.-M. (2000): *An encompassing life cycle centric survey of software inspection*, Journal of Systems and Software 50(1): 5-31.

[21] Kazman R., Barbacci M., Klein M., and Carriere S.J. (1999): *Experience with Performing Architecture Tradeoff Analysis*. *Proc. of the 21th International Conference on Software Engineering*, ACM Press.

[22] Kazman R., Carriere S.J. and Woods S.G. (2000): *Toward a Discipline of Scenario-based Architectural Engineering*. *Annals of Software Engineering, Kluwer Academic Publishers*, *9* (1-4), pp. 5-33.

[23] Land L.P.W., Jeffery R. and Sauer C. (1997): *Validating the Defect Detection Performance Advantage of Group Designs for Software Reviews*, Proc. ESEC / SIGSOFT FSE.

[24] Lassing N., Bengtsson P., Bosch J. and Vliet, H.V. (2002): *Experience with ALMA: Architecture-Level Modifiability Analysis*. *Journal of Systems and Software*, *61* (1), pp. 47-57.

[25] Lassing N., Rijsenbrij D. and van Vliet, H. (2003): *How Well can we Predict Changes at Architecture Design Time? Journal of Systems and Software*, *65* (2), pp. 141-153.

[26] Lassing N., Rijsenbrij D. and van Vliet, H. (1999): *On Software Architecture Analysis of Flexibility, Complexity of Changes: Size isn't Everything*. *Proc. of 2nd Nordic Software Architecture Workshop*.

[27] Ozkaya I., Bass L., Nord R.L., and Sangwang R.S. (2008): *Making Practical Use of Quality Attribute Information*, IEEE Software, pp.25-33.s

[28] Parnas D. L. and Weiss D. M. (1985): *Active design review: principles and practices*. Proceedings of the 8th International Conference on Software Engineering, pages 215-22.

[29] Porter, A. A. and Johnson, P. M. (1997): *Assessing Software Review Meetings: Results of a Comparative Analysis of Two Experimental Studies*, IEEE Transactions on Software Engineering, Vol. 23, No. 3.

[30] Seaman C.B. and Basili V.R. (1998): *Communication and Organization: An empirical study of Discussion in Inspection Meetings*, IEEE Transactions on Software Engineering, Vol. 24, No. 6.

[31] Votta L. (1993): *Does every Inspection need a Meeting?* ACM Software Eng. Notes, vol.18, no.5, pp.107-114.

[32] Williams L.G. and Smith C.U. (2002): *PASA: A Method for the Performance Assessment of Software Architecture*. *Proc. of the 3rd Workshop on Software Performance*.

[33] Winkler D., Biffl S. and Thurnher B. (2005): *Investigating the Impact of Active Guidance on Design Inspection*, 6th Int. Conf. on Product Focused Software Process Improvement (PROFES), Oulu, Finland.

[34] Wohlin C., Runeson P., Höst H., Ohlsson M.C., Regnell B. and Wesslén A. (2000): *Experimentation in Software Engineering - An Introduction*, Kluwer Int. Series in Software Engineering, Kluwer Acad. Publishers.