Improving the Quality of Architecture Design Through Peer-Reviews and Recombination

Mojtaba Shahin(✉) and Muhammad Ali Babar

CREST – The Centre for Research on Engineering Software Technologies,
The University of Adelaide, Adelaide, Australia
{mojtaba.shahin,ali.babar}@adelaide.edu.au

Abstract. Software architecture reviews help improve the quality of architecture design decisions. Traditional reviews are considered expensive and time-consuming. We assert that organizations can consider leveraging peer-reviews and recombination (i.e., promoting design improvement through sharing design ideas) activities to improve the quality of architectures and getting staff trained. This paper reports a case study aimed at exploring the potential impact of combining peer-review and recombination on the quality of architecture design and design decisions made by novice architects, who usually have limited practical experience of architecture design. The findings show that the use of peer-review and recombination can improve both the quality of architecture design and documented decisions. From the decision-making perspective, this study also identifies the main types of challenges that the participants faced during architectural decision making and reasoning. These findings can be leveraged to focus on the types of training novice architects may need to effectively and efficiently address the types of challenges identified in this study.

Keywords: Software architecture design · Design quality · Peer-review · Recombination · Architectural design decision

1 Introduction

With the increasing size and complexity of software-intensive systems, the role of software architecture (SA) as a means of understanding and managing large-scale software intensive systems is considered very critical. The high level design description of a large system can help a system's stakeholders to understand and reason about the designed architecture with regards to architecturally significant requirements (ASRs) of a software-intensive system [23]. Software architecting is a knowledge-intensive activity, in which a large amount of knowledge is being continuously consumed and produced. A poor quality architecture can lead to project failure that usually costs an organization dearly. Software development organizations pay significant attention and allocate resources to design an appropriate architecture that can help achieve the functional and quality requirements expected of a system by all the stakeholders. That is why organizations focus on building their competencies in designing and evaluating architectures before committing substantial resources to build-

D. Weyns et al. (Eds.): ECSA 2015, LNCS 9278, pp. 70–86, 2015.
DOI: 10.1007/978-3-319-23727-5_6

ing a system [12, 13, 15]. Software architecture reviews are usually performed informally by architects themselves or formally by quality assurance teams [13].

An architecture review is considered as an effective way to ensure the quality of software architecture design [12, 15]. However, the current architecture review methods and processes have not been widely adopted by industry due to a large number of limitations [12, 14]. Historically, (formal) architecture review processes rely on time-consuming, tedious and expensive face-to-face meetings [12, 14]. Given the increasing trend to leveraging crowdsourcing in knowledge-intensive activities, we assert that software architecture community should explore the potential role of crowdsourcing as an alternative method in designing and evaluating software architectures and getting novice architects to gain the required knowledge, skills, and experience by soliciting the contributions from the online communities [16]. We argue that two concepts peer-review and recombination can be leveraged simultaneously to improve the quality of architecture design. Peer-review is a reciprocal process, in which people working in groups comment on the work of peers and provide feedback on the reviewed work [9]. Peer-reviews have been applied to several disciplines for identifying the potential defects and improve the quality of the final product [9]. It is demonstrated that crowdsourcing can help reduce development cost, faster time to market and increase the quality through soliciting diverse expertise and creativities from a large workforce [16].. In the recombination process, a specific type of crowdsourcing, the designers should share their designs to others and then they are encouraged to use the ideas from the shared designs if appropriate for revising their own design [3]. The recombination can be interpreted as an indirect collaboration [25]

The main goal of this study is to investigate the role of peer-review process in combination with recombination on the quality of architecture design and design decision. We were also interested in identifying and classifying the types of challenges that the participants faced when asked to deign an architecture for a non-trivial system to be developed using state-of-the-art technologies of mobile cloud computing. We have conducted a case study involving students in an academic context. The findings provide preliminary evidence to support our assertion that combining peer-review and recombination can help improve the quality of architecture design and design decisions as the participants took inspirations and borrowed ideas from designs of their peers and got engaged in intense design reasoning discussions. We have also identified four main categories of challenges that the participants of our study faced. The findings are expected to encourage further studies of leveraging crowdsourcing in software architecture design and guide the future training programs that can prepare the future architects to effectively and efficiently address the types of architecture design challenges faced by the participants of our study.

The rest of the paper is organized as follows: section 2 gives a summary of background and motivation. Section 3 provides the details of the case study. The quantitative and qualitative results of the study are described in Section 4. The section 5 reports a discussion on findings. Finally, we present our conclusions with future work in Section 6.

2 Background and Motivation

Whilst there has been significant research on improving the quality of software architecture through architecture evaluation (i.e., reviews) [12], there has been little work on exploring the impact of peer-reviews on software design quality. Other design disciplines have devoted significant amount of efforts to investigate how feedback, (self) critique, and peer-review can improve design [1, 2]. Dow et al. [1] studied the impact of feedback on the quality of web advertisement designs when created in parallel and serially. They found that the parallel feedback on design led to better quality and more divergence in design. Dow et al. [2] showed that designing and sharing multiple designs for group discussion increases the quality of design rather than sharing the best design for discussion. Moreover, they also found that sharing and discussing multiple designs can also lead participants to explore more concepts.

Mao et al. conducted a survey of using crowdsourcing to support software engineering activities [24]. The results of the survey reveal that although crowdsourcing has been widely employed for supporting coding and maintenance activities, it has been rarely used for software design. TopCoder[1], as one of a few commercial crowdsourcing platforms, supports crowdsourcing software design in which competitors are allowed to provide software design specification based on given user requirements [24]. However, very little research exists on how architecture design, review and evolution can be performed by multiple designers' solutions (i.e., crowd) [3, 24]. To the best of our knowledge, there has been only one paper [3], recently published, that reports a study similar to our line of research. LaToza et al have investigated the role of "recombination" in software design by Crowd [3]. In the "Recombination" process, designers are encouraged to share their designs with others and take ideas and inspiration through such sharing of design for improving their own. LaToza et al studied the impact of "Recombination" on the quality of two types of software design, user experience design and architecture design, through a design competition in which the participants (i.e., graduate students) were asked to share their initial design. The authors organized two separate studies of user experience design and software architecture design. Each of the participants was asked to produce an initial version and a revised version design. For the revised design, the participants were encouraged to take inspiration from other designs and the lessons learned from the crowd (i.e., other participants). The study concluded that the quality of software design can be improved through competitions and "Recombination" as almost all the participants borrowed at least one idea from other participants, who are considered "Crowd" in the study. One of the most interesting findings of the study was that even the strong designers used the ideas from the weak designs and improved their designs.

We came across the work of LaToza et al. [3], while analyzing our data. That study increases our confidence in the importance of exploring the potential benefits of crowdsourcing in design and training architects how to leverage the power of peer review and recombination for improving the quality of software design. Our study investigates the roles of peer-reviews and recombination together on the quality of

[1] http://www.topcoder.com/

architecture design. We are especially interested in comparing the quality of design decisions documented by novice architects before and after peer review and recombination. Thirdly, our study design promoted extensive discussions involving technical arguments in favor and against the reported design and counter justifications. These discussions provided huge amount of qualitative data that helped us to discover the types of challenges novice architects can face when designing architectures.

3 Research Design and Logistical Details

Our long term goal is to empirically build a body of knowledge about the dynamics and potential benefits involved in applying crowdsourcing for improving software architecture design when traditional architecture review have been proven too expensive to be widely adopted [12]. Based on the body of knowledge, we were also interested in training future software architects by identifying and classifying the types of common challenges they face during architecture design. This particular study purported to empirically study and understand the potential impact of crowd level reviews and discussions on the quality of design using peer-reviews and recombination. We identified two research questions for this work.

RQ1. How do peer-review and recombination affect the quality of architecture design? We planned to answer this question by analyzing the quality of the architecture design decisions and architecture designs submitted by each group of the participants before and after the peer reviews and recombination phases. The quality of the architecture designs and decisions has been quantified by applying the evaluation criteria (see sections 4.1.1 and 4.1.2). We also analyze the qualitative data from the discussions and the feedback of the teaching assistant who had observed the whole process and assessed the architecture designs.

RQ2. What challenges do the novice architects experience in architectural decision-making and design reasoning? We envisioned to answer this question by analyzing the discussions on the design decisions made available for review, students' reflections summaries in the submitted design reports, and the feedback of the teaching assistant on the students' performance on design decisions before and after the peer-review and recombination phases and the recurring challenges reported to her.

3.1 Research Method

An empirical study should be carried out using a suitable research method chosen based on the nature of the studied problem and the research questions to be answered. Since there has been scant research on the impact of peer-reviews and recombination on the quality of software design, we decided to carry out an exploratory case study in an academic setting. Case study is considered a suitable research method to investigate a contemporary phenomenon within its real-life context. Our study was an exploratory case study as it mainly deals with the "What" questions. Apart from the guidelines provided by Yin [8], we followed the checklist provided by Kitchenham et al. on case study research [27]. The unit of analysis is group consisting of 4 participants.

3.2 The Participants and The System

This study was carried out through the software architecture design and evaluation activities and submitted artifacts of 31 students who doing a senior level semester long (i.e., 14 weeks) software architecture course in 2014 at the University of Adelaide. Designing and evaluating architecture of a non-trivial software intensive system, healthcare emergency support, were the major assessment tasks (i.e., 50% of the final grade). The design activities were supposed to be carried out by groups of 4 members (one consisted of 3 members). The main training topics included quality attributes, architectural personas, concepts, principles, methods, and best practices of software architecture design, and documentation approaches.

The main goal of the system is to support Australian healthcare workers when responding to emergency situations away from the hospitals. The emergency response team can consist of paramedics, doctors and medical staff located at hospitals. The system is supposed to provide mobile and reliable access to the required information about the patients. The system was to be designed to leverage mobile cloud computing technologies using Service Oriented Architecture (SOA) principles. Security and privacy were identified as the most important quality attributes. The system is expected to be able to integrate with other systems of Australian health care system to become a part of a healthcare ecosystem.

3.3 Case Study Process

The case study process (i.e., shown in Figure 1) consisted of following steps:

1. Each group was given a set of requirements of a distributed emergency healthcare system and asked to design and document software architecture. Each group was supposed to provide following materials at the first phase:
    a. Concrete scenarios and reasons for the key quality attributes.
    b. A set of services to support different features of the system. The decisions made for identifying the services along with the rationale.
    c. Documented design decisions using suitable architectural styles and patterns along with the rationale for the choices made (in a given template). Table 1 shows the decision template along with an example of the documented design decision by group B.
    d. Model the Service Oriented Architecture (SOA) of the system by using SoaML [17] and show the use of patterns. The designed SOA was expected to contain the Service (including composed services) and Component layers.
2. The research team evaluated the quality of software architecture designs and design decisions made by each group in the first phase based on predefined criteria. The submitted architecture designs were evaluated by a senior Teaching Assistant (TA) who had more than 10 years of industry experience. The TA did not know about the study. Later the two authors evaluated the quality of the documented design decisions.

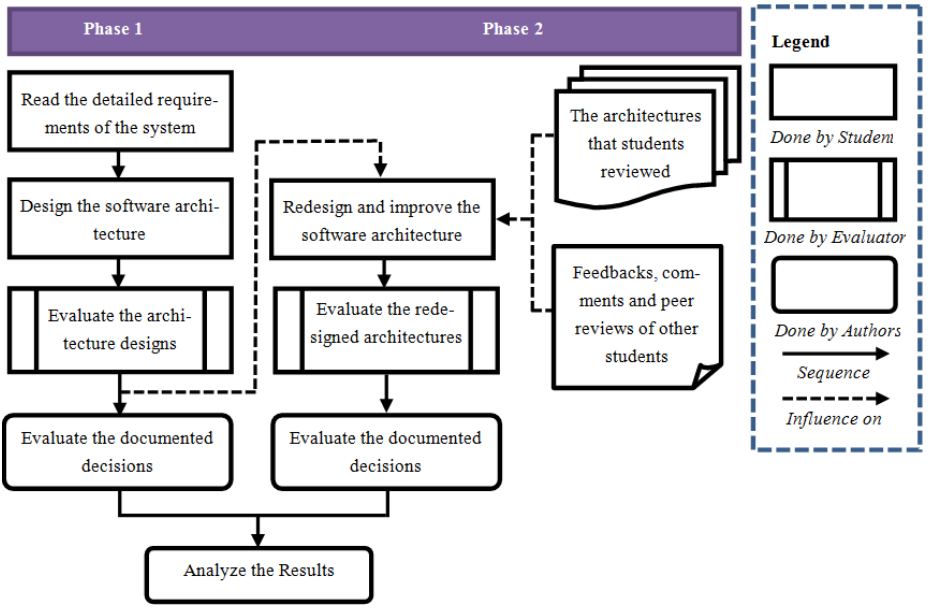

Fig. 1. The steps of case study

Table 1. A design decision captured using the template by group B.

Concern		How to develop the application to adapt to different devices?
Ranking criteria		1. Portability 2. Modifiability
Option(s)	Name	MVC pattern
	Description	MVC pattern divides an interactive application into three components and separates the data from presentation. The model contains the core functionality and data. Views display information to the user. Controller handle user input.
	Status	This option is decided
	Relationship(s)	-
	Evaluation	1. Portability is achieved by making the View an independent part and the system. GUI is not interwoven with functionality, so to adapt to different devices; we only need to change the View. 2. Modifiability is achieved by separating Model, View and Controller. Each part is independent and can be modified without affecting other parts.
	Rationale of decision	This option is decided because it provides a good solution to the quality attributes required.

3. In phase 2, each of the participants had two tasks:
   a. Individual task (i.e., participating in peer-review and recombination): each participant was expected to carefully and critically evaluate an assigned architectural solution submitted by one of the eight groups. The participants were expected to prepare a 1-2 pages summary of his/her evaluation and understating of the key architecture design decisions, their strengths, and weakness, and missing artifacts and post the individual summaries on the assigned Moodle forum that was viewable by all the participants. Each of them was expected to actively discuss and reflect upon the evaluation summaries submitted by other students and provide solid reasons and justification for the strengthens and weaknesses of different design choices and appropriateness of the architectural concepts and methods used. The participants were encouraged to identify and use the ideas from the design of their peers (i.e., the crowd) for revising their group designs for the second phase of the architecture design task. It should be noted that the participants were not given any clues by research team to learn which architecture designs were good candidates for recombination.
   b. Each group was required to improve their design by taking into consideration their own assessment of their architecture or the architecture that they would have found more suitable, and the feedbacks provided to them by other students through Moodle discussion forum and teaching assistant.
4. Like step 3, the TA and the authors again evaluated the quality of architecture designs and documented decisions submitted in the second phase.
5. The authors analyzed the quantitatively data based on all the evaluations of the architectures submitted in two phases for answering the RQ1 and qualitatively analyzed the students' discussions and reflections and feedback by the TA to answer the RQ2.

4 Results

Following sections report the results from the analysis of different types of data gathered for answering the two research questions that motivated this study.

4.1 Findings from Analyzing Quantitative Data

4.1.1 The Quality of Architecture Design

We quantitatively evaluated the quality of architecture design submitted by each group. We asked all the groups to submit two versions of their architecture designs. We expected that the second version submitted in second phase has been influenced by (i) the groups' assessment on their architecture; (ii) the architecture that they would have reviewed and (iii) the feedback and critiques provided to them by their peers through Moodle discussion forum and teaching assistant. For commenting and

critiquing other's designs, we encouraged the participants to follow the sequential critiquing framework [4] that is expected to help improve the quality of the critique. We had an independent evaluator (i.e., TA) to score the quality of architecture designs submitted in phase 1 and 2 separately. It is worth nothing that the evaluator did not know about this study. The submitted architectures were evaluated based on the following criteria; any group could have received 10 points if they satisfied each criterion (i.e., the maximal score can be 60 points)

1. Description of the Personas of doctors, paramedics, and other members of an emergency response team located at a hospital.
2. Three concrete scenarios and associated rationale for 2 quality attributes.
3. Five design decisions and rationale for achieving the identified quality attributes by applying suitable architecture and/or design patterns.
4. SOA models using SoaML to show different services, their interactions.
5. Provide components/service diagrams using any 3 views of the 4+1 views.
6. Provide the details of some services by showing the class diagrams (application of design patterns) and the sequence diagrams.

Table 2. The summary of architecture design scores

Groups	Score of Phase 1	Score of Phase 2	Changes in scores
Group A	47	50	3
Group B	60	62	2
Group C	47	48	1
Group D	49	51	2
Group E	38	41	3
Group F	48	50	2
Group G	58	60	2
Group H	47	49	2
Mean	49.25	51.38	2.13

A comparison of the two architecture designs provided by each group is shown in Table 2 that clearly shows that all of the groups got better score in the second phase. It is interesting to note that the increment obtained after peer review and recombination could be between 0 and 3 points (i.e., 3 as the largest improvement). Each group's score on architecture design improved by approximately 2.2 points on average. We applied Wilcoxon signed-rank test as a nonparametric statistical method to assesses whether or not the improvement was statistically significant [5]. We chose this type of statistical method as the outputs of those participants had to be evaluated twice [10]. The dependable variable is the quality of architecture designs before and after peer-review and recombination process. The application of the Wilcoxon signed-rank test revealed that the quality of the architecture designs submitted in phase 2 was significantly better than the quality of the architecture designs submitted in the first phase at the level of confidence of 95% (see Table 3). These findings provide the evidence that the peer-review and recombination help improve the quality of architecture design. The feedback and critique provided by the participants to each other

helped the participants to fix their decisions, reasoning and design flaws, but also it encouraged them to explore broadly design space and think more critically [1, 2].

Table 3. Descriptive statistics of the results on quality of architecture design

	Phase 1		*Phase 2*		*p-value*
	Mean	SD	Mean	SD	
	49.25	6.92	51.38	6.72	**p=0.011**

4.1.2 The Quality of Design Decisions

Previous section reports the findings from our investigation of the impact of peer review and recombination activities on the quality of all the artefacts submitted as part of architecture design, i.e., all sorts of decisions and documentation. This section reports the findings from particularly investigating whether or not the peer review process has an impact on the quality of design decisions. The participants had been asked to document each of the key design decisions using a given template along with the rationale for that design decision. We noticed that the participants documented the design decisions using the template (i.e., see Table 1, we name them as template-based decision) as well as without using the template (i.e., unstructured decision). In order to evaluate the quality of both types of design decisions, we extended the criteria proposed in [11] by adding three elements, which are expected to reflect architectural decision's quality. The quality of each decision was evaluated by the first author and the doubtful situations were discussed and agreed upon with the second author. Each criterion used a three points Likert scale: "Yes"=1, "No"=0 and "Partially"=0.5. The accumulated quality score for each decision was expected to range from 0 and 7. The following criteria have been employed:

- C1: Is the decision stated clearly?
- C2: Is the rationale of the decision stated clearly?
- C3: Is the documented decision is a viable solution with regard to the described?
- C4: Are multiple design options considered?
- C5: Are the pros and cons of decision compared?
- C6: Does the documented decision reuse any patterns/tactics /reference architectures?
- C7: Are quality attribute, constraints and business goal considered during decision making process?

We found that the number of template-based design decisions increased from 31 to 42 decisions from first phase to second phase and an average quality score per decision improved by 0.5 point. A comparison of the number of documented decisions along with the average quality score in phase 1 and phase 2 is shown in Table 4. It is clear from Table 4 that the average quality score for unstructured decisions increased by 0.3 point. Since we wanted to find out if the improvement in the quality of design decisions before and after the peer-reviewing process was statistically significant, we

used the Wilcoxon signed-rank test. Two paired tests (e.g., Wilcoxon signed-rank test) need the same number of samples in each condition. That was why we randomly selected 31 template-based decisions from phase 2 in order to make the samples equal (i.e., the number of decisions) with phase 1. Tables 5 and 6 contain the p-values for the template-based and unstructured decisions respectively. With a p-value of 0.003, we conclude that applying the peer-review process led to statistically significant improvement in the quality of template-based decisions. The investigation of the used criteria revealed that the design decisions documented by the participants in both phases 1 and 2 satisfied the criteria C1, C3 and C7 to a very large extent. The improved quality of the design decisions in phase 2 compared to phase 1 was mainly due to the increased scores in C2, C5 and C6. It appears that the peer review forced the novice architects (i.e., study's participants) to better justify their decisions in the second step. During the peer review phase, most of the participants were challenged by their peers with such kinds of questions: "*what was your rationale for this given decision*". These questions encouraged the participants to document the positive points of their decisions in the revised version of architecture documents (i.e., C5). However, the novice architects rarely talked about the drawbacks of the choices they made. In phase 2, the students applied more architectural patterns and tactics for satisfying the architectural issues. One possible reason for this trend could be that the participants learnt how to employ architectural patterns for architectural problems and understood that justifying the design decision with architectural patterns is easier. There was no statistical significance in the improvement over criterion C4 between phases 1 and 2.

Table 4. The summary of the number of and quality scores of template-based and unstructured decisions

	Phase 1	*Phase 2*
Number of template-based decisions	31	42
Number of unstructured decisions	11	18
Average quality per template-based decision	4.64	5.19
Average quality per unstructured decision	3.50	3.83

Table 5. Descriptive statistics of the results on quality of template-based decisions

Phase 1		*Phase 2*		*p-value*
Mean	SD	Mean	SD	
4.64	1.03	5.19	0.88	**p=0.003**

Having done the same analysis for the unstructured decision, we could not find a significant deference between the quality of the unstructured decisions before and after the peer reviewing process (i.e., p-value>0.05). Similar to the template-based decisions, the unstructured decisions positively fulfilled the criteria C1 and C3 in both phases 1 and 2. The scores of other criteria except C4 improved from first phase to the second one, however, it was not sufficient to have a statistical significance.

Table 6. Descriptive statistics of the results on quality of unstructured decisions

	Phase 1		*Phase 2*		*p-value*
	Mean	SD	Mean	SD	
	3.50	1.09	3.86	0.83	p=0.29

4.2 Findings from Analyzing Qualitative Data

Our study design and execution also included several means of collecting qualitative data, which is considered an important source of evidence and can supplement the findings from analyzing quantitative data. We gathered qualitative data from dozens of pages of design decisions discussions by students on Moodle forum, students' reflections notes in the submitted design reports, and teaching assistant's feedback. For analyzing the qualitative data, we employed thematic analysis [6], a qualitative data analysis method for identifying, analyzing, and reporting patterns (themes) from the collected data. We decided to analyze the data to identify the challenges that the participants faced and shared with their peers. Following are the main categories of challenges that were reported through discussions.

Service Decisions: One of the key tasks was to decide and reason about the types of services required of the system to be designed. Our analysis revealed that the participants found it quite challenging to determine what services and sub-services were required. It was also difficult for the participants to decide about the levels of abstractions to be used for describing the identified services and justifications for them. This situation made it difficult for others to easily understand the reported service. One of the participants described this challenges: "*This report should give more detail of the connections of each service. One of the important thing is they should give the details of the task manager service. How can this service integrated with other service, and how the services communicate in the system?"*

Design Decisions: Our analysis revealed that the participants faced many problems in making and documenting design decisions as well as in understanding the decisions made by other groups. It was also revealed that the participants did not consider and reasoned about more than one design option when making the design decisions; nor did they document all the decisions made. There was a significant difference between the number of documented decisions in architecture document (AD) and that of we derived from the discussion forum. This situation changed during the peer review process as the participants frequently asked questions like "*what your rationale is for decision X or pattern Y*". This types of questions and peer critiques of the reported design decisions and the rationale provided resulted in detailed and intense discussions about design decisions, rationale and the design options that could have been considered. The simulants for such discussions were questions like "*what is your rationale for this decision*?" and "*how can you ensure that your decision reaches quality X*?" "*how the system can avoid attacks such DDoS attacks*?"

These types of questions made the participants talked about the design options that they thought about, their pros and cons, and why those design options had been rejected. One participant shared that "*We can also add some measures which can help reduce the impact of DoS attacks, such as using 2-factor authentication and rejection of any further requests from a device until it has been confirmed. With the rejection being done on a separate cluster with redundancy to reduce the chance that this service will be taken down from such an attack*." Another participant replied, "*The main concern that I see with using 2-factor authentication is that in the high-pressure environment of an emergency accident scene or in a hospital emergency room, authentication can be time consuming and stressful for emergency staff. As such, it is very important to ensure that the authentication methods chosen are as simple to use as possible. One idea would be to use a staff member's NFC access card.…our group is considering the option of having the device itself as a possible token, where that device is registered to the system for approved access by a specific user*"

Despite the participants complaining about not having sufficient documented information about design decisions shared for peer review, the situation did not improve in the second phase either. Albeit the quality of the overall design and design decisions improved, but most of the documented design decisions lacked sufficient rationale. Compared with the phase 1, some students did mention about the difficulty in ranking different design options and selecting the best solution based on the required quality attributes, constraints, and patterns' forces. We found that participants had more difficulty to justify the decisions related to the whole structure of the system (i.e., architectural decision [20]) than those used to meet component-level design issue (i.e., low-level decision [20]). Our analysis also revealed that the participants found it hard to maps the individually reported design decisions onto overall architectural views. For example, they frequently asked "*where we can find the impact of this decision on architecture design*". It became clear that a general lack of established traceability between the architectural decisions and the overall architecture designs submitted by different groups made it difficult to gain a good understanding of the designed architectures. Moreover, a large majority of the groups did not document execution decisions (i.e., tool, technology, process, organization as per Kruchten's classification of decisions [7]) using decision template. However, some of the participants did mention such decisions in their comments. For example, one participant commented, *"It is sure that we need more work on the multi-platform. As we use MVC design pattern and SOA software architecture, it is just an easy work move to other platform, we will use HTML 5 to design the GUI, So that all the devices can access to the system. We will make the description more clear and specific."*

Quality Attributes Decisions: Our analysis revealed that most of the groups did not provide any details about the trade-offs among quality attributes in the documented design decisions in the phase 1, however, they demonstrated a reasonable understanding of considering tradeoff decisions among quality attributes when prompted. For example, one participant remarked, "*… the design decision of data encryption is significantly correct and necessary here. However, the performance issue that may be caused by encryption is not taken into consideration in this decision*". The response to this comment was: "*You're right that we didn't mention anything about performance*

here. We were aware of the issue, but we'll make sure to put it in writing for phase 2". On another group's submission, a participant commented, "… *you put a central server, which is the cloud in one location, and it only handles the logic process. It needs to retrieve data from data servers distributed in different locations. If this is the case, … it increases the data communication between central server and data servers a lot. Latency may increase here*". The response was, "*As you can see, we did not mention this decision in the design decision section which we definitely should. The problem you raised, the latency, is a problem indeed… we pay more attention to the security part than the latency… we shall document this design decision in the next phase… we will regard it as a trade-off point and further discuss it in detail*".

Pattern Challenges: the analysis of the qualitative data also indicated that the participants were having challenges in understanding the goals and suitability of patterns reported to be used in software architecture design diagrams. One reason for that situation appeared to be missing information about the names of the patterns used. For example, one participant commented, "…*What is the name of the design pattern or patterns being employed in this design?..strictly the Model-View-Controller (MVC) pattern or is it a hybrid of MVC and the repository pattern. This was not clearly stated and looking at their diagrams and architectural design I would make the assumption of the Model-View-Controller pattern but it is not always easy to tell*."

5 Discussion and Limitations

Peer Review and Crowdsourcing in Architecture Design and Review: The findings from our exploratory case study provide preliminary evidence to support the assertion that peer-review and recombination approaches can improve architecture design and decisions quality. Such an improvement can possibly be examined as follows: (i) Once the novice architects participated in the peer review process, they were motivated to justify the rationale of their architecture design and the decisions made; (ii) most of the feedback and critique provided by the participants to each others were constructive; (iii) reviewing and looking at the architecture designs of peers enabled the participants to explore broadly the design space and consider the possible design alternatives. Since designing and reviewing an architecture of a complex software systems heavily rely on knowledge and expertise from different fields as well as experience and intuitions, which is usually beyond the possession of a given organization, we believe that organizations can employ the peer-review and recombination through crowdsourcing for improving the quality of final architecture design. Linus's law (i.e., "given enough eye balls, all bugs are shallow" [18]) has shown the effectiveness of a peer-review process and it has been adopted as an effective practice for quality improvement by Open Source Software communities [19]. Since a peer-review through crowdsourcing can leverage the experiences and expertise of a large number of individuals, we assert that it can result in better architecture design quality. An organization can use the method as an effective approach to reviewing software designs. It is different to traditional architecture review, which involves formal, time-consuming and expensive meetings. It can also reduce the tension raised during the face-to-face

meetings. There needs to be more research to examine the opportunities and perils of a design peer-review process by crowdsourcing.

More Training and (Semi) Automated Decision Making Support: The results have revealed that novice architects, who are supposed to be the next generation of architects, had many problems during decision-making process. Although they were successful in capturing and documenting the design decisions and their rationale to a satisfactory extent, but they experienced many challenges in other steps of the decision making process such as proposing design alternative, evaluating the alternatives, tradeoff analysis between conflicting quality attributes and selecting the best solution. We can assert these challenges may partially stem from lack of enough expertise and experience. These findings are similar to the results reported in [20], which revealed that the personal experience is a major influencing factor in making and documenting architectural decisions. It can be said that there needs to be more focus on providing the future architects with sufficient training and experience in different aspects of designing and evaluating design decisions and providing appropriate support for (semi) automate decision-making to improve the efficiency and effectiveness of the architecture design and evaluation activities. In a recent study, it was found that most of the existing design decision tools just focus on modeling, capturing and documenting decisions without providing sufficient automation support for decision-making [11]. We assert that the areas for automation support can be ranking design options, generating design alternatives, and supporting quality attributes trade-off analysis.

Limitations: One of the key limitations of this study can be the process and evaluators of the architecture design and decisions. Albeit the external evaluator was not aware of the study when evaluating the submitted artifacts, it would be better to apply double-blinded evaluation process to reduce the impact of potential bias in the evaluation of architecture design and decisions. We tried to alleviate this threat by intensive discussions between the two authors to reach a consensus on different evaluations performed on the artifacts used in this study. The other validity threat could be the participants of this study and system (i.e., healthcare emergency support system) being studied. Since there has been a little research on the impact of peer-review and recombination on the quality of software design and decisions, we decided to start this exploratory research by studying students' design and evaluation activities in an academic context as those students can be considered the future generation of software architecture professionals [21]. Researchers have found that students are suitable replacements for industry professionals if performing small tasks of judgment [22]. We plan to extend this research in a number of ways including the possibility of conducting a study with software architects from industry.

The third validity treat is how to ensure the improved quality of architecture designs and decisions in phase 2 was indeed due to peer review and recombination, not simply because of knowledge acquired by the participants (i.e., learning factor) through the study. We agree that the learning factor has the potential to be a confounding factor in our study as well as it is closely intertwined to peer-review and recombination techniques, but we argue that the participants as novice architects did not have the tendency to self-question and self-critique on their own designs and they

preferred to start self-question and self-critique after getting external feedbacks and comments from the teaching assistant and peers (i.e., crowd). We assert that our suggested techniques and particularly peer-review process can challenge "*Law of Least Effort"* [26].

6 Conclusion and Future Work

We have carried out an exploratory case study to investigate how peer-review and recombination affect the quality of architecture design. The quality of designed architectures and documented decisions by software architecture students, as novice architects, before (as a first version) and after the peer-reviews and recombination (as second version) were examined and the results have enabled us to conclude that: (1) the peer-review and recombination activities can potentially improve quality of software architecture design and decisions (i.e., particularly decisions documented by template). (2) The novice architects can face specific types of challenges in design decision making process: (i) determining the required levels of abstractions to be used for describing the identified services and justifying them. (ii) Reporting the rationale for decisions made and proposing and ranking design options. (iii) Performing tradeoff decisions among conflicting quality attributes. (iv) Understanding the goals and suitability of patterns reported to be used software architecture design diagrams. We conclude that these findings can lead to design and execution of better training programs for novice architects to help them to gain the required knowledge and experience in relatively short amount of time as the technological advancement and increasing complexity require software development organization to have highly skilled and experienced architects.

Our ongoing future work can be outlined as follows: (1) we plan to replicate our case study in different settings and different sizes of population and with practitioners to explore if similar findings can be achieved with different contexts attributes. (2) We plan to further investigate the qualitative data from the Moodle forum to find out the types of design decision that were mentioned and discussed in the forum.

References

1. Dow, P.S., Glassco, A., Kass, J., Schwarz, M., Schwartz, D.L., Klemmer, S.R.: Parallel Prototyping Leads to Better Design Results, More Divergence, and Increased Self-efficacy. ACM Transactions on Computer-Human Interaction **17**(4) (2010)
2. Dow, P.S., Fortuna, J., Schwartz, D., Altringer, B., Schwartz, D.L., Klemmer, S.R.: Prototyping dynamics: sharing multiple designs improves exploration, group rapport, and results. In: The SIGCHI Conference on Human Factors in Computing Systems, pp. 2807–2816 (2011)
3. LaToza, T.D., Chen, M., Jiang, L., Zhao, M., van der Hoek, A.: Borrowing from the crowd: a study of recombination in software design competitions. In: 37th International Conference on Software Engineering (2015)

4. Xu, A., Bailey, B.P.: A crowdsourcing model for receiving design critique. In: CHI 2011 Extended Abstracts on Human Factors in Computing Systems, pp. 1183–1188 (2011)
5. Armitage, P., Berry, G.: Statistical Methods in Medical Research, 3rd edn. Blackwell (1994)
6. Braun, V., Clarke, V.: Using Thematic Analysis in Psychology. Qual. Res. Psychol. **3**(2), 77–101 (2006)
7. Kruchten, P.: An ontology of architectural design decisions in software intensive systems. In: 2nd Groningen Workshop on Software Variability, pp. 54–61 (2004)
8. Yin, R.: Case Study research: Design and methods, Sage Publications, Inc. (2003)
9. Nicol, D., Thomson, A., Breslin, C.: Rethinking Feedback Practices in Higher Education: a Peer Review Perspective. Assessment & Evaluation in Higher Education **39**(1), 102–122k (2014)
10. McCrum-Gardner, E.: Which is the Correct Statistical Test to Use? British Journal of Oral and Maxillofacial Surgery **46**(1), 38–41 (2008)
11. Lytra, I., Gaubatz, P., Zdun, U.: Two Controlled Experiments on Model-based Architectural Decision Making. Information and Software Technology **63**, 58–75 (2015)
12. Ali Babar, M., Gorton, I.: Software Architecture Review: The State of. Practice **42**(7), 26–32 (2009)
13. Tang, A., Lau, M.F.: Software Architecture Review by Association. Journal of Systems and Software **88**, 87–101 (2014)
14. Tang, A., Kuo, F.-C., Lau, M.F.: Towards independent software architecture review. In: Morrison, R., Balasubramaniam, D., Falkner, K. (eds.) ECSA 2008. LNCS, vol. 5292, pp. 306–313. Springer, Heidelberg (2008)
15. Maranzano, J.F., Rozsypal, S.A., Zimmerman, G.H., Warnken, G.W., Wirth, P.E., Weiss, D.M.: Architecture Reviews: Practice and Experience. IEEE Software **22**(2), 34–43 (2005)
16. Klaas-Jan Stol, K., Fitzgerald, B.: Two's company, three's a crowd: a case study of crowdsourcing software development. In: 36th International Conference on Software Engineering, pp. 187–198 (2014)
17. Service Oriented Architecture Modeling Language (SoaML) Specification, OMG. http://www.omg.org/spec/SoaML/1.0.1/PDF
18. Raymond, E.S.: The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary. O'Reilly (2001)
19. Wang, J., Shih, P.C., Carroll, J.M.: Revisiting Linus's Law: Benefits and Challenges of Open Source Software Peer Review. International Journal of Human-Computer Studies **77**, 52–65 (2015)
20. Weinreich, R., Groher, I., Miesbauer, C.: An Expert Survey on Kinds, Influence Factors and Documentation of Design Decisions in Practice. Future Generation Computer Systems **47**, 145–160 (2015)
21. Kitchenham, B., Pfleeger, S., Pickard, L., Jones, P., Hoaglin, D., El Emam, K., Rosenberg, J.: Preliminary Guidelines for Empirical Research in Software Engineering. IEEE Transactions on Software Engineering **28**(8), 721–734 (2002)
22. Host, M., Regnell, B., Wohlin, C.: Using Students as Subjects - A Comparative Study of Students and Professionals in Lead-time Impact Assessment. Empirical Software Engineering **5**(3), 201–214 (2000)
23. Bass, L., Clements, P., Kazman, R.: Software Architecture in Practice, 3rd edn. Addison Wesley, Boston (2012)

24. Mao, K., Capra, L., Harman, M., Jia, Y.: A Survey of the Use of Crowdsourcing in Software Engineering. Technical Report RN/15/01, Department of Computer Science, University College London (2015)
25. Jiang, L.: Recombination Contest: Crowdsourcing Software Architecture and Design. Master Thesis, University of Amsterdam (2014)
26. Kahneman, D.: Thinking, Fast and Slow. Penguin (2011)
27. Kitchenham, B., Pickard, L., Pfleeger, S.L.: Case Studies for Method and Tool Evaluation. IEEE Software **12**(4), 53–62 (1995)