Chapter 6

Improve Individual Architectural Decisions


In this chapter, we expand the work in Chapter 5 by proposing an approach (REGAIN) based on the Repertory Grid technique, for capturing architectural decisions made by individual architects. We also present a study to ensure that REGAIN meets the needs of industrial architects. We interviewed sixteen architects, who indicated REGAIN advantages, such as systematic decision making support. Also, architects indicated improvement opportunities, in particular tool support and the possibility to prioritize concerns that are used to evaluate decision alternatives. To address the need for prioritization, we conducted an additional study to evaluate two approaches for prioritizing concerns: pairwise-comparisons and the hundred-dollar approach. We conducted an experiment with thirty graduate students to compare the two prioritization approaches. Based on the results of the experiment, we added the hundred-dollar approach to REGAIN.
6.1 Introduction

In Chapter 5, we argued that existing approaches in software architecture overlook the challenges of capturing tacit architectural knowledge. Overlooking this challenge limits the practical usefulness of approaches for capturing architectural knowledge. To address the challenge of capturing tacit architectural knowledge, we draw inspiration from the knowledge engineering discipline, because this discipline focuses on knowledge-related approaches, so it may offer ideas with high potential for tackling the problem of capturing tacit architectural knowledge.

One such idea, which we presented in Chapter 5, is to use the Repertory Grid technique - a powerful interviewing technique for tacit knowledge acquisition (Boose, 1984; Gaines and Shaw, 1993). In Chapter 5, we proposed an early version of an approach which uses the Repertory Grid technique for capturing architectural decisions. In this chapter, we call this approach REGAIN (REpertory Grid for capturing ArchIitectural decisioNs), and we present its details in Section 6.2.

The version of REGAIN in Chapter 5 lacked industrial validation. Therefore, this chapter investigates and improves REGAIN in industrial practice, to ensure that REGAIN meets the needs of industrial architects for capturing tacit architectural knowledge. In detail, this chapter makes two main contributions:

1. It reports feedback from practitioners on using REGAIN for capturing real-world architectural decisions
2. It presents an improvement of REGAIN by adding a prioritization approach that has been selected based on empirical data from an experiment.

These contributions help address the challenge of capturing tacit architectural knowledge in industrial practice, and encourage using ideas from the knowledge engineering discipline to tackle the problem of architectural knowledge vaporization.

Figure 6.1 shows the three research phases that we followed, and the corresponding sections of this chapter where the phases are presented. Phase 1 (in Section 6.2) presents the initial REGAIN approach, background
6.2. Phase 1 – Initial REGAIN Approach

In this section, we present REGAIN and a short summary of our evaluations of the Repertory Grid technique from Chapter 5.

6.2.1 Theoretical Foundations for REGAIN

REGAIN is based on two major theoretical foundations: the core model of architectural knowledge in (de Boer et al., 2007), and the Repertory Grid Technique (presented in Chapter 5).
In Chapter 5, we mapped concepts from the core model of architectural knowledge (de Boer et al., 2007) to concepts from the Repertory Grid technique. In Figure 6.2, we update the mapping for REGAIN. Compared to the original core model, we renamed the rank action to the evaluate action, and the ranking element with the rating element. When making an architectural decision, there is a topic, \( n \) alternatives, \( m \) concerns, and \( n \times m \) ratings of alternatives against concerns. For example, a decision on the topic of choosing a database can include three databases (such as MySQL, MS SQL, and PostgreSQL), five concerns (such as scalability, availability of documentation, performance, familiarity, and costs), and fifteen ratings for each database against each concern (e.g. using a five-point scale). The actual decision is the alternative that satisfies best the concerns. The rationale for the decision is captured in the evaluations of alternatives against concerns (de Boer et al., 2007) (i.e. the ratings indicate which alternative satisfies best the concerns).

The concepts in the core model in (de Boer et al., 2007) have equivalent concepts in other established models for architectural knowledge: the IEEE 1471-2000 standard (1471-2000, 2000), Tyree’s template (Tyree and Akerman, 2005), and Kruchten’s ontology (Kruchten, 2004). The equivalency among concepts is detailed in (de Boer et al., 2007), and we offer below the following three examples.

1. The concept of concern in the core model (de Boer et al., 2007) is equivalent to the concepts of requirement and risk in Kruchten’s
6.2. Phase 1 – Initial REGAIN Approach

ontology (Kruchten, 2004), and to the concepts of assumption and constraint in Tyree’s template (Tyree and Akerman, 2005).

2. Since architecting is an iterative activity, a decision loop appears: starting from an initial decision topic, the architect makes an architectural decision, which, in turn, may uncover new concerns and decision topics, for which subsequent related decisions need to be made. The decision loop is equivalent to the relationships among decisions in Kruchten’s ontology (Kruchten, 2004). For example, the ‘must use J2EE’ decision introduces a constraint relationship with the ‘use the GlassFish application server’ decision in Kruchten’s ontology (Kruchten, 2004), while in the core model (de Boer et al., 2007) the ‘must use J2EE’ decision introduces the ‘which application server to use?’ decision topic, for which GlassFish is one of the considered alternatives. Overall, the decision loop enables traceability among decisions.

3. The decision rationale concept in the IEEE 1471-2000 standard (1471-2000, 2000) is equivalent to the evaluations of alternatives against concerns in the core model in (de Boer et al., 2007). The evaluations explain why the architect selected a certain alternative.

6.2.2 The REGAIN Approach

The REGAIN approach consists of the following five steps.

In step one of REGAIN, the architect identifies the architectural decision topic (e.g. choice of a framework for developing a new web application).

In step two, the architect generates a list of alternatives, relevant for the decision topic (e.g. a list of web frameworks). Alternatives may be reused from previous similar decision. Optionally, a hypothetical ideal alternative may be included (e.g. ideal web framework). The ideal alternative would represent the most preferred ratings for each concern in an ‘ideal’ world, without any tradeoffs. Comparing alternatives against the ideal alternative provides a basic utility function: alternatives with ratings more similar to the ideal alternative offer more utility than alternatives with less similar ratings.

In step three, the architect provides concerns that are used to evaluate alternatives identified in step two. The architect can reuse concerns from
previous decisions (e.g. through keywords-based suggestions offered by tool support). Otherwise, the architect provides concerns using triadic elicitation. As mentioned earlier, triadic elicitation means that the architect selects three random alternatives and then asks how two alternatives are similar, but different from the third. For example, for the decision topic on a Python web framework, an architect might consider Web2py and Zope similar in terms of popularity (i.e. both are used by fewer popular existing websites) and Django different with regard to popularity (i.e. there are more popular existing websites implemented with Django). Thus, the architect captures popularity as a concern in terms of a contrast (fewer and more popular existing websites). Repeated comparisons among other triads of alternatives produce more contrasts to express the concerns.

For eliciting concerns, in Chapter 5 we explored using two approaches: sentence completion and triadic elicitation. For REGAIN, we chose to use the triadic elicitation approach, since it is better suited than sentence completion at making tacit knowledge explicit in a systematic manner. Jankowicz (Jankowicz, 2001) indicates three reasons for this. First, triadic elicitation encourages expressing concerns as concrete contrasts (or poles) between the alternatives, instead of only labels for concerns (e.g. popularity). Second, triadic elicitation encourages describing contrasts in a precise and operational manner. For example, an architect can interpret popularity in terms of how many search results are returned for the names of the frameworks, so the architect should express his definition of popularity as a contrast and indicate as precisely as possible the values for the left and right poles of the contrast. Third, triadic elicitation encourages making explicit subjective concerns, such as architects’ intuitions or gut feelings about the alternatives.

In step four, the architect judges each alternative against every concern and captures the judgment in a rating. A typical rating scale uses integers from one to five, to help the architect indicate how an alternative satisfies a concern (e.g. assigning one for an alternative being less popular, or assigning five for being very popular).

In step five, the architect analyzes the resulting grid, using content and structure analysis (Jankowicz, 2003). Content analysis refers to evaluating the decision topic (e.g. is this topic relevant), alternatives (e.g. should other alternatives be added to the grid), concerns (e.g. can some concerns be added
6.2. Phase 1 – Initial REGAIN Approach

or removed), and ratings (e.g. are there missing ratings, or do ratings need to be changed). The content analysis helps the architect understand if all components (e.g. alternatives, concerns) of the decision were included in the grid. Structure analysis of a grid uses cluster (Shaw, 1980) and principal components (Jankowicz, 2003) analyses. Both involve statistical operations for analyzing relationships between decision alternatives and concerns. This helps the architect make the decision. The result of cluster analysis is part of the output of REGAIN.

6.2.3 REGAIN Output

The output of REGAIN consists of two main artifacts. First, the grid (or matrix) contains alternatives, concerns, and ratings. Second, two dendrograms (i.e. trees with relationship similarities between items) are produced from the cluster analysis of alternatives and concerns. The cluster analysis uses a nearest neighbor hierarchical clustering algorithm, with the city-block distance (Shaw, 1980).

Figure 6.3 presents the grid and dendrograms (i.e. clusters) for an architectural decision collected in Chapter 5. The decision topic was technology for data visualization. The REGAIN session produced seven alternatives (lower part of Figure 6.3) and ten concerns (upper part of Figure 6.3). As part of the analysis, alternatives and concerns are reordered and grouped based on their similarity. The level of similarity is calculated using the distances between ratings. Alternatives and concerns are grouped in clusters, based on their similarity levels. A similarity level of 100% between two alternatives means that the two alternatives have the same ratings for all concerns. Based on the city-block distance, in Figure 6.3, SVG (i.e. Scalable Vector Graphics) is the most similar alternative to the ideal technology (around 75%, which is the highest similarity level between any pair of alternatives). For the architectural decision in this grid, the practitioner chose the SVG alternative, which is the closest alternative to the ideal technology.
6. Improve Individual Architectural Decisions

We reported two studies with evaluations of the initial REGAIN approach (i.e. the Repertory Grid technique) in Chapter 5, as shown in Figure 6.1. The first study in Chapter 5 aimed at exploring the feasibility of the initial REGAIN approach. To this end, we conducted an exploratory study with seven graduate and undergraduate students, who used REGAIN to capture architectural decisions from their previous academic and industrial projects. After students had captured decisions, we interviewed them to obtain feedback on advantages and disadvantages of REGAIN. We found out that the approach helps students make architectural decisions in a systematic manner, and that it encourages students to reflect on their decisions. Also, the approach produces a concise documentation of architectural decisions, which reduces architectural knowledge vaporization.

In the second study from Chapter 5, we investigated if the REGAIN approach reduces architectural knowledge vaporization. In the study, we asked twenty graduate students to capture architectural decisions with the initial REGAIN approach (similar as in the first study). Next, we compared the output of capturing decisions with REGAIN with the architectural documentation on the same decisions that students created as part of a software architecture course project. For the course project, students received a template for documenting the architecture of a software system. The template included sections for documenting architectural decisions, to indicate decision topic and rationale. Although students spent much time on the course project architectural documentation, we found out that REGAIN was more efficient (i.e. it took less
6.3. Phase 2 – Investigate Industrial Applicability of REGAIN

6.3.1 Research Method, Data Collection and Analysis

The research goal of Phase 2 was to obtain feedback on REGAIN from industry, for the purpose of understanding advantages, disadvantages, and potential improvements to REGAIN, from the viewpoint of practitioners (i.e. architects in the industry), in the context of architects’ efforts for capturing architectural decisions. To achieve the stated research goal, we formulated the following research questions.

RQ1. What are the advantages and disadvantages of REGAIN?

RQ2. What are the improvement opportunities for REGAIN?

To answer these research questions, we conducted an interview study with architects from the industry. We selected architects with a wide range of experience levels, which is representative for the situation in the industry, where architects have various experience levels. We contacted architects in our network and architects from open source projects (e.g. SOFA Statistics project at www.sofastatistics.com). Upon a positive response, we conducted individual interview sessions with them, to capture their architectural decisions with REGAIN.

Each session took typically two hours, and followed three steps:
6. Improve Individual Architectural Decisions

1. We presented the REGAIN approach, including examples of grids, and we answered questions from the practitioners.

2. We asked each practitioner to use the REGAIN approach to capture one to three architectural decisions that practitioners made in their recent work. We did not ask architects to make new decisions. As tool support for REGAIN, we used a web tool called WebGrid (Gaines and Shaw, 2007).

3. Participants filled out a post-questionnaire, and offered additional feedback in semi-structured interviews.

Regarding ethical concerns, we asked practitioners to omit confidential information from the sessions. We indicated that results were to be used only for research purposes and that participants could withdraw at any time from the sessions or refuse to answer questions. We made audio recordings of the sessions, with the permission of each practitioner. Most sessions were conducted face to face, except for two sessions that took place via Skype.

Overall, we interviewed 16 practitioners from 14 different organizations. The Appendix for Chapter 6 has details on the practitioners and the sessions. Overall, practitioners had an average of nine years of experience in the software engineering industry. In total, practitioners used the REGAIN approach to capture 24 architectural decisions that the practitioners made in their industry projects from various application domains. As part of the sessions, we showed practitioners the REGAIN output for their decisions. This way, practitioners could get a clear idea on what to expect from using the REGAIN approach on their real-world decisions, and practitioners were able to offer feedback on the REGAIN approach.

To answer RQ1 and RQ2, we analyzed the answers from the post-questionnaire and the transcripts of the audio recordings from the interviews. We used descriptive statistics on the answers in the post-questionnaire. Two of the authors of this study performed content analysis on the transcripts and assigned one of the following three codes to pieces of content: advantage (RQ1), disadvantage (RQ1), or improvement opportunity (RQ2) for the REGAIN approach.
6.3. Phase 2 – Investigate Industrial Applicability of REGAIN

6.3.2 RQ1 – REGAIN Advantages and Disadvantages

6.3.2.1. Post-questionnaire Analysis

Table 6.1 shows the statements in the post-questionnaire. Statements ID1 to ID5 relate to advantages of REGAIN, in terms of helping communicate the rationales of decisions (ID1 and ID5), offering structured explanation and documentation of the rationale (ID2, ID3), and supporting structured architectural decision-making (ID4). Statement ID6 allowed architects to indicate the effort required for using REGAIN. Architects indicated their level of agreement with each statement using 1 (strongly disagree), 2 (disagree), 3 (neutral), 4 (agree), or 5 (strongly agree). We notice that architects agreed with the statements about communicating rationale (ID1 and ID5), structured explanation (ID2), documentation (ID3), and making (ID4) of architectural decisions. Finally, practitioners perceived a reasonable amount of spent effort (ID6).

In Table 6.1, we compare the answers of the industry architects with the answers from the students in a previous study in Chapter 5, in which students filled out the same post-questionnaire, after using the REGAIN approach.
6. Improve Individual Architectural Decisions

Table 6.1. The values indicate the average and standard deviation of the indicated ratings for each statement, for architects (A) and students (S).

<table>
<thead>
<tr>
<th>Statement</th>
<th>A</th>
<th>S</th>
<th>Diff</th>
<th>p</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The output of the session helps explain to a colleague why I took a certain decision.</td>
<td>4.29</td>
<td>4.14</td>
<td>0.15</td>
<td>97%</td>
</tr>
<tr>
<td>2. The session provides a structured way of explaining architectural decisions.</td>
<td>3.86</td>
<td>4.14</td>
<td>-0.28</td>
<td>33.6%</td>
</tr>
<tr>
<td>3. The session provides a structured way of documenting architectural decisions.</td>
<td>4.07</td>
<td>3.57</td>
<td>0.50</td>
<td>8.4%</td>
</tr>
<tr>
<td>4. The session provides a structured way of making architectural decisions.</td>
<td>3.64</td>
<td>4.00</td>
<td>-0.36</td>
<td>38.6%</td>
</tr>
<tr>
<td>5. When trying to understand why another architect took a certain decision, I could use such an output.</td>
<td>4.00</td>
<td>4.29</td>
<td>-0.29</td>
<td>44.6%</td>
</tr>
<tr>
<td>6. I found the session too tiring for the output.</td>
<td>2.21</td>
<td>2.43</td>
<td>-0.22</td>
<td>64.2%</td>
</tr>
</tbody>
</table>

We notice that there are only small differences on averages and standard deviations between the answers from students and the answers from industry architects (i.e. fifth column in Table 6.1). We investigate further the differences, by checking if there are any statistical significant differences between students and industry architects. The two samples (i.e. students and practitioners) are independent from each other, as they consist of different persons. We cannot assume that the samples are drawn from a normal distribution, so we use a non-parametric test: the Mann-Whitney U test. The null hypotheses are that there are no differences between the two samples on the statements ID1 to ID6. We use SPSS to apply the Mann-Whitney U test, and we obtain the p-values in Table 6.1. Since all values are higher than 5%, we cannot reject any of the null hypotheses. Therefore, we consider safe to accept that there are no statistically significant differences between the answers from students and the answers from practitioners on the six statements in Table 6.1.
6.3. Phase 2 – Investigate Industrial Applicability of REGAIN

Overall, the results of the post-questionnaire indicate two points. First, the REGAIN approach provides value to practitioners, in terms of costs (i.e. not too tiring) and benefits (i.e. systematic approach). Second, the REGAIN approach provides value to both inexperienced architects (i.e. students) and industry architects.

6.3.2.2. Transcripts Content Analysis

Following the content analysis of the interviews transcriptions, we identified the following codes for advantages.

1. **Decision-making support** – This code refers to helping architects in their decision-making. This code was indicated by 6 out of 16 participants. One participant remarked: ‘now I have a more clear view about the choices that have been made and the analysis provided a more clear direction on where to move in the future.’ Another participant said: ‘Without such a tool, I don't know how to distinguish between the different possibilities, in a more evident way. This tool provides evident results of what is really the ideal for this situation.’

2. **Decision rationale** – This code refers to offering rationale for the architectural decisions, and it was indicated by five participants. One participant said: ‘Also to explain your decision to others, I think it would give a good impression, and easy to understand. If someone sees it, maybe he doesn't know the project.’

3. **Reasoning support** – This code refers to the architects obtaining a clearer perspective on their decision, and it was indicated by four participants. One of the participants said: ‘from my personal view, now I have a clearer view about choices I made.’ Another participant said: ‘The nice thing about this tool is it's very simple. If people need a lot of time for criteria, that's because they don't have the criteria clear in their own mind, that's not a drawback of the tool, it's offering you a mirror about your own conceptions and criteria in your design. It could be a painful experience for somebody who is being interviewed. But that's not a drawback of the tool, it's exposing what you have in your mind.’
6. Improve Individual Architectural Decisions

4. **Systematic** — This code refers to how well structured the approach is, and it was indicated by two participants. One of the participants said: ‘I like the structured way of thinking about the decision, the questions you asked: are there any additional issues? Are these similar? Are there communalities? I like the approach very much, I think it’s very helpful, it also gives the person you ask a clear focus. It really helps me think about how the decision was taken, what we did, what were the most important constraints, it’s something like a feedback so I can reflect myself.’

Following the content analysis, we identified the following codes for disadvantages.

1. **Insufficient tool support** — This code was indicated by five participants, who complained on various issues of the tool that we used during the sessions. For example, one participant said: ‘I think the tool should have more workflow support, you know the next step, but someone using it for the first time might not. The tool should be more user-friendly.’ Another participant remarked: ‘Having some open source decision making tool would be very helpful.’

2. **Subjectivity** — This code refers to the fact that REGAIN encourages capturing subjective concerns about the alternatives. While useful for making knowledge explicit, subjective concerns might have limited reusability, since they express personal perspectives that might not be applicable in different situations. This code was indicated by three participants. One of the participants said: ‘my criteria might be subjective, to my bias, my experience might not be complete.’ Another participant commented: ‘the subjectiveness of the constructs is the big problem for reusability.’

3. **Effort** — This code refers to the effort involved for using REGAIN, and this code was indicated by three participants. For example, one participant said: ‘people need some training of how this is working. I suggest an e-learning thing to use it.’

An additional result to the answer for RQ1 emerged from the content analysis: the application context of REGAIN. We obtained the following.
6.3. Phase 2 – Investigate Industrial Applicability of REGAIN

1. **Project size** – One participant found the approach useful for large projects, rather than for small or medium-sized. Another participant considered the approach useful especially for expensive decisions.

2. **Domain** – One participant considered the approach useful for enterprise architectural decisions, in addition to software architectural decisions.

3. **Time** – Regarding when to use the approach, one participant suggested using the approach for ongoing projects, so that decision makers can benefit from using the approach. Four participants suggested using the approach multiple times during the decision lifetime. One of the participant stated: ‘It’s very nice to do this repetitively, not only for the final choice. For example, after one or two months to apply again these criteria to see where you are moving, alter your changes or criteria, and do it again in two weeks or more, to have an indication on where you are moving compared to your initial goals.’ Two participants suggested using the approach for the early architectural decisions. For example, one participant said: ‘It can help a lot in decision making processes. This is very important at the very initial stages of making choices. This is a very good thing to do at the starting point of a project.’

There are similarities between the feedback from practitioners and the feedback from students in our previous study in Chapter 5. Both practitioners and students indicated the same advantages: systematic, insight, reflective, and decision-making support. In addition, practitioners and students indicated the same disadvantages: learning curve, tool support, and time consuming. Moreover, practitioners and students indicated similar application context for REGAIN: large projects and early architectural decisions. Overall, the findings from the interviews with practitioners confirm our previous findings from Chapter 5.

6.3.3 RQ2 – REGAIN Improvement Opportunities

Following the content analysis, we identified the following codes for REGAIN improvement opportunities.
1. **Concerns prioritization** – This code refers to specifying priorities for concerns. Ten participants indicated this code. One of them said: ‘Adding in the weights would make it even more precise. That would make it very useful, this is extremely important to me, because in a specific project weights can vary a lot.’

2. **Group decisions** – This code refers to using REGAIN for group architectural decisions. Ten participants indicated this code. One of them said: ‘I should find a way to do this together [with others], but that will become a mess: sitting around one screen. Maybe you should have an online version where each fills it in, then results are pulled, you get an outcome like this, but averaged for the whole team.’

3. **Decision reuse** – This code refers to reusing previous items from REGAIN outputs (e.g. alternatives, concerns) for documenting new architectural decisions. Five participants indicated this code. One of them stated: ‘I think it would be easier to reuse it, because you know what kind of criteria you must take into account when making such decisions, and different projects might have few other criteria. That way you can definitely reuse it: this kind of decisions are made in every project, every java project: what kind of UI are we using? So, some of these alternatives would be reusable.’

4. **Interpretation assistance** – This code refers to offering guidelines for interpreting the output of cluster analysis, so that architects can interpret the output with the help of the REGAIN tool, without help from another person. Five participants indicated this code. One of them stated: ‘perhaps there should be some idea about what the differences, now it's between 64 and 70%: what does that mean? Is that big or small? This can help the reader: if it's less than 10% it's quite similar. Now, it doesn’t say anything to me now.’

5. **Confidence factors for ratings** – This code refers to indicating confidence factors to ratings, which would express the architect’s level of confidence in each assigned rating (e.g. if the architect has a vague idea that a certain alternative has high popularity, then the architect assigns a rating and a low confidence score for the rating). Capturing confidence factors helps the architect document his level of knowledge.
6.3. Phase 2 – Investigate Industrial Applicability of REGAIN

about the various alternatives. In the study, four participants indicated this code. One of them stated: ‘many criteria cannot be assessed from the very beginning of a project's lifecycle; they might reveal themselves at the testing phase. One could have a criterion and give a rating according to estimations, but the real rating could be assigned only when there are some test results in hand.’

6. Optional extra explanations – Two participants indicated this code, which refers to having the possibility of adding optionally more details to explain the meaning of some concerns or ratings. One participant said: ‘What I meant was adding a bit more explanations about what the two poles mean. Clicking on text gives more explanations. Maybe give the user an option so that when scoring, to type why he chose that number. I would like that purely optional, so that people don't need to type a lot. Sometimes, some decisions need to be elaborated more.’

7. Sensitivity analysis – This code refers to the possibility of analyzing the impact on REGAIN output when changing ratings for a certain concern. Two participants indicated this code. One of them said: ‘You should do something on price. What is the influence on the concern? Some price sensitivity thing would be useful.’

8. More types of rating scales – One participant indicated the need to have more types of rating scales, depending on the nature of the concerns.

9. Decision dependency – One participant indicated the need to consider dependencies among architectural decisions, which refers to offering support for the analysis of not only one architectural decision, but for analyzing sets of decisions that depend on each other (e.g. choice of web framework and operating system).

6.3.4 Discussion

The two previous studies with students in Chapter 5 indicated that REGAIN helps reduce architectural knowledge vaporization. The next step was to investigate the industrial applicability of REGAIN, with the goal of obtaining feedback from practitioners. We formulated research questions on
advantages/disadvantages and improvement opportunities for REGAIN, and conducted an interview study to answer the research questions.

The interview study brings empirical evidence on the advantages and disadvantages that practitioners can expect when using REGAIN to make and capture architectural decisions. The results indicate that REGAIN is a systematic approach that offers practitioners important advantages: decision-making support, documentation of decision rationale, and reasoning support. However, REGAIN has some disadvantages: insufficient tool support, subjectivity and required effort. Still, insufficient tool support seems to be the most important disadvantage, since five practitioners indicated it in the study. Another disadvantage, subjectivity (indicated by three practitioners) is a minor disadvantage as it affects only partially the future reuse of captured decisions, since subjective concerns might be less applicable across multiple architects. The effort disadvantage is the expected cost for using any systematic approach. Furthermore, this study shows improvement opportunities for REGAIN, which reduce the disadvantages, and further help practitioners in their activities.

An interesting result is that this study indicates similarity between feedback from students in Chapter 5 and feedback from practitioners. The similarity suggests that students might be an acceptable proxy for practitioners, in the sense that results from empirical evaluations with students might be applicable to practitioners. Following this study, we agree with Tichy’s perspective that students differ marginally from practitioners, except for knowledge of domain-specific areas, large scale systems, and organizations (Tichy, 2000). Tichy’s perspective encourages broader usage of students in empirical evaluations, which makes much sense for the software architecture community, because studies with industry architects are very difficult to conduct, due to architects’ busy schedules. If more researchers confirm the applicability of results from studies with students to practitioners, within the limits indicated by Tichy, then the software architecture community will be able to produce results with high statistical significance, by getting access to more study participants (i.e. students, instead of practitioners).

The following research directions emerge from this study. First, we identified improvements to REGAIN, such as concerns prioritization, group decisions, support for decision reuse, and addressing decision dependencies. Since
concerns prioritization was the most demanded improvement (see Section 6.3.3) by practitioners, we present our efforts to add this improvement in Section 6.4. We will address the other improvements in future work. Finally, insufficient tool support was the most mentioned disadvantage of REGAIN (see Section 6.3.2.2...2), therefore we implemented user-friendly, open source tool support for REGAIN, which we present in Chapter 8.

6.4 Phase 3 - Investigate Prioritization Approaches

Architects indicated the need to prioritize concerns for REGAIN, as presented in Section 6.3. Furthermore, priorities of concerns are part of architectural knowledge. For example, if the priority of security is higher than the priority of performance, then this difference in priorities explains why a more secure, but slower alternative for a database system was selected. If these priorities are not captured, then such architectural knowledge is lost, resulting in higher evolution costs. Therefore, it is very important to answer the following research question:

RQ3. Which concerns prioritization approach to use for REGAIN?

To identify previous work on prioritization of concerns for REGAIN, we surveyed existing literature on the Repertory Grid technique. We found that some Repertory Grid tools include features to capture priorities of concerns (Boose et al., 1990a; Gaines and Shaw, 2007). Shaw and McKnight describe a basic prioritization approach: ranking concerns from one to ten based on their importance (Shaw and McKnight, 1981). We could not identify any other study on prioritization approaches for the Repertory Grid technique, in particular for prioritizing concerns for architectural decisions.

We chose to investigate prioritization approaches that are already established and accepted in the software architecture community. In Section 6.6, we mention other available prioritization approaches (i.e. ordinal scale approaches) that we considered, but decided not to use with REGAIN. A recent survey on architectural decision-making techniques (Falessi et al., 2011) indicates that two techniques (Gilb, 2005; Kazman and Klein, 2001) (one of which is CBAM (Kazman and Klein, 2001)) use the hundred-dollar approach for prioritization, while three other decision-making techniques (Al-Naeem et al., 2005; Andrews et al., 2005; Svahnberg et al., 2003) use the pairwise-
comparisons approach. In addition, other researchers also use pairwise-comparisons for prioritization in architectural decision-making (Jabali et al., 2011). Since these two approaches are already established and accepted in the software architecture community, we started to investigate which of them to use with REGAIN.

The hundred-dollar approach requires decision makers to assign a value between 0 and 100 to items that need to be prioritized, so that the sum of all values is 100. The pairwise-comparisons approach (or analytic hierarchy process) requires decision makers to compare all possible pairs of concerns. This helps identify prioritization inconsistencies (e.g. $A$ is more important than $B$, $B$ is more important than $C$, but $A$ is as important as $C$) (Saaty, 1990). Since the hundred-dollar and pairwise-comparisons approaches are already used by other architectural decision-making techniques, this indicates that these two prioritization approaches are familiar and effective to architects, at least to some extent. So, we considered using one of them for improving the REGAIN approach, and we set out to find which one to select.

No study compares the hundred-dollar and pairwise-comparisons approaches for making architectural decisions, so we did not know which one to use with REGAIN. For choosing between the two prioritization approaches, we considered several factors. First, the performance (in terms of time) of a prioritization approach is important, because architects have busy schedules and they prefer prioritization approaches that take less time. Second, the perception of architects (e.g. learnability) about a prioritization approach influences their decision to use or not to use the approach. Third, architects want to capture accurate priorities, which reflect their perspectives. To summarize: The objective of the study in Phase 3 is to analyze the hundred-dollar and pairwise comparisons approaches for the purpose of comparison with respect to their performance, users’ perception, and impact on REGAIN analysis output, from the viewpoint of architects, in the context of deciding which prioritization approach to add to the REGAIN approach. To achieve this research objective, we performed an experiment with the two prioritization approaches.

Next, we describe participants in the experiment, experimental materials, tasks performed by participants, hypotheses, experimental parameters, design, as well as the results of the experiment.
6.4. Phase 3 - Investigate Prioritization Approaches  

6.4.1 Participants

Participants in the experiment were graduate students who took the Software Architecture course at the University of Groningen in the Netherlands. As part of the course, students worked in groups of six over a period of ten weeks to architect a home automation system that would interact with the Smart Grid to trade electrical power, and manage energy consumption of home appliances. For the experiment, we prepared tasks for which students had enough training, to eliminate the need for professionals-only knowledge. Tichy argues that computer science graduate students differ marginally from professionals, in the sense that professionals might be more knowledgeable of domain-specific areas, large scale systems, and organizations (Tichy, 2000). Therefore, if a study does not require such extra knowledge from graduate students, then study results are applicable to professionals. Thus, we minimized knowledge requirements about domain-specific areas, large systems and organizations, to make our study results applicable to professionals.

We addressed the ethical aspects of conducting the experiment as part of a course (Berander, 2004) by using a checklist for integrating research and teaching goals (Carver et al., 2009; Galster et al., 2012):

1. **Ensure adequate integration of the study into the course topics** - The software architecture course stressed the importance of architectural decisions. During the experiment, students learnt about using the REGAIN approach to capture architectural decisions.

2. **Integrate the study timeline with the course schedule** - The course schedule included an optional seminar session in week four, in which we conducted our experiment. In week four, students already had made a few architectural decisions for their course project. However, they had to make and capture more architectural decisions. Thus, they could benefit from REGAIN in their course project.

3. **Obtain subjects’ permission for their participation in the study** - We announced the optional seminar session at the beginning of the course. We informed students that the seminar would address advanced software architecture topics. By showing up for the study, students indicated implicitly their consent to participate in the study.
We specified clearly that participation or performance in the seminar had no impact on grades.

4. **Plan follow-up activities** - To increase the educational value for students, we scheduled debriefing sessions with groups of participants, within three weeks after the study. We prepared individual packages, so that students could see their own results. In each half-hour debriefing session, we presented the study details, preliminary results, and we answered questions from students.

Of the 36 graduate students enrolled in the software architecture course, 30 participated voluntarily in our experiment. Twenty-three participants were Dutch, two Indonesian, one Icelander, one Indian, one German, and one Chinese. All participants had bachelor’s degrees, in computer science or informatics (20), industrial engineering management (8), industrial automation (1) or Mathematics (1). Participants had an average of 3.6 years (standard deviation of 2.2) of practical experience in software development, and an average of 1.2 years (standard deviation of 1.3) in software architecture.

### 6.4.2 Experimental Materials and Tasks

We prepared the following experimental materials, which are available online at (Tofan, 2012).

1. Instruction about how to perform experimental tasks, including examples.

2. Two grids that described two architectural decisions that students had to make for their course project: choice of user-interface and choice of storage. The grids included decision topics, decision alternatives and concerns for each of the two topics, so that participants would not have to use the whole REGAIN approach during the experiment, due to time constraints. We derived decision topics, decision alternatives and concerns for the two grids from project reports from previous years. In the Appendix, we show the grid for the user-interface architectural decisions.

3. Four prioritization forms: two types of prioritization forms for both grids. For the hundred-dollar approach, participants had to fill out amounts for each concern in a grid, depending on their perceived
importance, with the constraint that sum is 100. For pairwise-comparisons, participants had to fill out comparisons among all pairs of concerns (e.g. moderately more, extremely less important). Both types of forms also asked participants to fill in the time when they started the prioritization task and when they completed it.

4. A post-questionnaire on the background of participants, questions about how participants perceived each prioritization approach, and a section for general feedback on the experiment.

Figure 6.4 shows the steps for executing the experiment, which took place in one session. All students gathered in the same room. In the first step, we presented the plan for the session, we gave a brief introduction on the key concepts for the session, and an overview of tasks to be performed by the students. In step two, half of the participants went to a different room, accompanied by one research assistant. In step three, all participants rated alternatives against concerns for the two architectural decisions, from their personal perspectives, using their knowledge on their project. In steps four and five, participants prioritized concerns for both decisions with the hundred-dollar and pairwise comparisons approaches. We collected the paper forms for step four, before handing over the forms needed for step five. In the final step, participants filled in the post-questionnaire.
6. Improve Individual Architectural Decisions

Throughout the session, we encouraged participants to ask for clarifications about their tasks. Furthermore, we placed no time restriction on tasks. This ensured that participants understood their tasks and were comfortable with their tasks.

6.4.3 Hypotheses and Variables

We present the hypotheses on the prioritization approaches, regarding their impact on performance, users’ perceptions, and REGAIN output.

6.4.3.1. Performance

We consider two dimensions for evaluating performance of prioritization techniques: required time to perform prioritization using an approach, and scalability of the approach. Scalability concerns the variation of required time for a prioritization approach with the number of concerns to be prioritized. We define the following scalability ratio metric to measure scalability.

\[ \text{RS} = \frac{t_a}{t_b} \times \frac{b}{a} \]  

(1)
6.4. Phase 3 - Investigate Prioritization Approaches

For this metric, \(a\) and \(b\) are the number of prioritized concerns, and \(a\) is smaller than \(b\). A participant needs a time period \(t_a\) to prioritize \(a\) concerns and a time period of \(t_b\) to prioritize \(b\) concerns. \(rs\) indicates the variation of the time required to prioritize concerns when the number of concerns increases. For example, if \(rs\) equals 1, then doubling the number of concerns means that at least twice as much time is needed to prioritize them (linear scalability). If \(rs\) is smaller than 1, \(rs\) indicates negative scalability. If \(rs\) is larger than 1, then doubling the number of concerns needs at least less than half the time (positive scalability).

To evaluate the performance of the hundred-dollar and pairwise comparisons approaches, we test the following null hypotheses.

\(H_{a0}\): Participants need the same amount of time for prioritization.

\(H_{b0}\): The hundred-dollar and pairwise comparisons approaches have the same scalability ratios.

6.4.3.2. Users’ Perceptions

We operationalize users’ perceptions in terms of ease of use, ease to learn and attractiveness of a prioritization approach. The post-questionnaire and the results are presented in the analysis section, in Table 6.3. We test the following null hypotheses for users’ perceptions.

\(H_{c0}\): Participants perceive the hundred-dollar ($100) as equally easy to use compared to the pairwise comparisons (PWC) approach.

\(H_{d0}\): Participants perceive the hundred-dollar ($100) as equally easy to learn compared to the pairwise comparisons (PWC) approach.

\(H_{e0}\): Participants perceive the hundred-dollar ($100) as equally attractive compared to the pairwise comparisons (PWC) approach.

6.4.3.3. REGAIN Output

Priorities of concerns influence REGAIN output, which is the result of cluster analysis on the grid with the architectural decision. Figure 6.5 shows an example of the impact of prioritization approaches on the REGAIN output. The example uses two grids with identical decision alternatives (A1 to A5), concerns ((C1, -C1) to (C4, -C4)) and ratings. However, the concerns in the
two grids have different priorities. All concerns in the left grid have a priority of 10. In contrast, the concerns in the right grid have various priorities (i.e. 10, 15, 30, and 45). In Figure 6.5, the lower triangular matrix shows percentages of similarities among decision alternatives A1 to A5 (left grid), as calculated by the WebGrid tool. The upper triangular matrix shows percentages of similarities among A1 to A5 for the right grid. For example, the similarity between A2 and A3 is 68.8 for the left grid, and 76.2 for the right grid.

Figure 6.5. Percentages of similarities between pairs of alternatives vary by modifying priorities of concerns.

We compare clusters based on the distance between the similarity matrices from which the clusters are derived. The distance between the similarity matrices is calculated as follows:

$$\text{Distance}_{SP,PWC} = \sum_{i=1}^{N} \sum_{j=1}^{N} \left| SP_{i,j} - PWC_{i,j} \right|$$  \hspace{1cm} (2)

In Eq. (2), N is the number of alternatives. SP is the similarity matrix of alternatives, calculated assuming that all concerns have the same priority. PWC is the similarity matrix of alternatives, calculated with the priorities obtained from using the pairwise comparisons approach. Using the above metric, we define the following hypothesis.

$$H_0: \text{Distance}_{SP,PWC} = \text{Distance}_{SP,100} = \text{Distance}_{PWC,100}$$

$$H_1: \text{Distance}_{SP,PWC} \neq \text{Distance}_{SP,100} \neq \text{Distance}_{PWC,100} \neq \text{Distance}_{SP,PWC}$$

6.4.3.4. Summary

Overall, in this experiment we investigate how prioritization approaches influence performance, users’ perceptions, and REGAIN output. We summarize the independent and dependent variables in Table 6.2. The first
column shows variables and corresponding hypotheses. The last column shows how the hypotheses will be tested.

<table>
<thead>
<tr>
<th>Variable</th>
<th>H</th>
<th>Type</th>
<th>Scale</th>
<th>Unit</th>
<th>Range</th>
<th>Testing procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prioritization approach</td>
<td>NA</td>
<td>I</td>
<td>Nominal</td>
<td>NA</td>
<td>PWC, $100</td>
<td>NA</td>
</tr>
<tr>
<td>Needed time</td>
<td>(H_a)</td>
<td>D</td>
<td>Ratio</td>
<td>s</td>
<td>&gt; 0</td>
<td>Mann-Whitney test</td>
</tr>
<tr>
<td>Scalability</td>
<td>(H_b)</td>
<td>D</td>
<td>Interval</td>
<td>NA</td>
<td>&gt; 0</td>
<td>Mann-Whitney test</td>
</tr>
<tr>
<td>Ease to use</td>
<td>(H_c)</td>
<td>D</td>
<td>Nominal</td>
<td>NA</td>
<td>PWC, $100</td>
<td>Binomial test</td>
</tr>
<tr>
<td>Ease to learn</td>
<td>(H_d)</td>
<td>D</td>
<td>Nominal</td>
<td>NA</td>
<td>PWC, $100</td>
<td>Binomial test</td>
</tr>
<tr>
<td>Attractiveness</td>
<td>(H_e)</td>
<td>D</td>
<td>Nominal</td>
<td>NA</td>
<td>PWC, $100</td>
<td>Binomial test</td>
</tr>
<tr>
<td>Distance between similarity</td>
<td>(H_f)</td>
<td>D</td>
<td>Ratio</td>
<td>%</td>
<td>&gt;=0</td>
<td>Kruskal-Wallis, Mann-Whitney tests</td>
</tr>
<tr>
<td>matrices</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

6.4.4 Experiment Design and Results

This experiment used a randomized, paired comparison design, with one factor and two treatments (Juristo and Moreno, 2001). The factor was the prioritization approach. Treatments were the hundred-dollar and pairwise-comparisons approaches. Since each subject used both prioritization approaches, the order effect was a validity threat, as results might be confounded by using one prioritization approach before the other. Also, participants might experience fatigue or boredom after using an approach. Therefore, we used counterbalancing in our design, by randomly assigning participants to two similar groups. The two groups used the prioritization
approaches in different orders, as shown in Figure 6.4: participants in Group 1 used pairwise-comparisons, and afterwards they used the hundred-dollar approach. However, participants in Group 2 used the approaches in the reverse order.

To increase validity, we exclude data points from subjects who have a consistency ratio larger than 20% in the pairwise comparisons approach. The rationale is that such data points indicate inconsistencies in the pairwise comparisons (e.g. an inconsistency exists if x is more important than y, y more important than z, but z is more important than x) (Karlsson and Ryan, 1997).

6.4.4.1. Results on Performance

Regarding required time, the histograms in Figure 6.6 present the frequency of time intervals needed by participants to apply each prioritization approach, for both decisions. We note that the user-interface decision (left side) had a maximum time of around five minutes, while the storage decision (right side) had a maximum time of around eleven minutes. Furthermore, the histograms indicate the distribution of the time intervals. For example, the leftmost histogram indicates that most participants (i.e. eleven) needed between 33 and 66 seconds to prioritize concerns for the user-interface decision using the hundred-dollar approach.
6.4. Phase 3 - Investigate Prioritization Approaches

We use the Mann-Whitney U Test to test the null hypothesis that both approaches need the same amount of time ($H_{00}$) for both decisions. The p-value for both tests is 0.000, so we reject the null hypothesis and we accept the alternative hypothesis that the required time for $100$ differs from the required time for PWC. Based on histograms in Figure 6.6, we conclude that PWC needs more time than $100$.

Regarding scalability of the prioritization approaches, we calculate $rs$ for the two prioritization approaches as outlined in Eq. (1), using the amounts of time for the UI and Storage decisions ($t_a$ and $t_b$), which have four and eight concerns ($a = 4$, $b = 8$). We remove two outliers with ratios of 6 and 18, and we display boxplots of $rs$ in Figure 6.7. Most ratios for the hundred-dollar test indicate positive scalability, while ratios for pairwise-comparisons indicate negative scalability.
To gain more detailed insights into these findings and to check for neutral scalability, we used the Mann-Whitney U test. However, neither hundred-dollar test, nor pairwise comparisons scale neutrally ($p = 0.043$ and $p = 0.00$). Therefore, we accept the alternative hypotheses: hundred-dollar test scales positively, and the pairwise-comparisons approach scales negatively.

### 6.4.4.2. Results on Users’ Perceptions

Table 6.3 lists questions from the post-questionnaire related to the perception of users, including the frequency of answers to each of these questions. Answers were either $100 or PWC. To test the hypotheses in Table 6.3, we ran binomial tests on the numbers for $100 and PWC. We obtained the $p$-values in the last column in Table 6.3.
Table 6.3. Number of answers in the post-questionnaire.

<table>
<thead>
<tr>
<th>Hypothesis</th>
<th>Post-questionnaire item</th>
<th>$100</th>
<th>PWC</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ease to use</td>
<td>Which of the two approaches was easier to use?</td>
<td>23</td>
<td>7</td>
<td>0.003</td>
</tr>
<tr>
<td>Ease to learn</td>
<td>Which of the two approaches was easier to learn?</td>
<td>22</td>
<td>7</td>
<td>0.008</td>
</tr>
<tr>
<td>Attractiveness</td>
<td>Which of the two approaches was more fun to use?</td>
<td>28</td>
<td>2</td>
<td>&lt;0.01</td>
</tr>
<tr>
<td>Attractiveness</td>
<td>If choosing between the two approaches, which would you prefer to use in your decision making?</td>
<td>24</td>
<td>6</td>
<td>&lt;0.01</td>
</tr>
</tbody>
</table>

Since all p-values are smaller than 0.05, we accept the alternative hypotheses: participants perceive the hundred-dollar test as easier to use, easier to learn and more attractive than the pairwise comparisons approach.

6.4.4.3. Results on REGAIN Output

The boxplots in Figure 6.8 summarize the data for the 18 valid data points, for the UI and storage decisions. The vertical axis represents the city block distances among each pair of prioritization approaches.

The Kruskal-Wallis test for the UI decision indicates a statistically significant difference between distances ($H(2) = 13.635, p = 0.001$), with a mean rank of 38.64 for SP-PWC, 22.75 for $100$-PWC, and 21.11 for SP-$100$. Therefore, we reject the null hypothesis that distances between similarity matrices are equal ($H_0$). Repeated Mann-Whitney tests indicate differences between SP-PWC vs. $100$-PWC ($p = 0.006$), and SP-PWC vs. SP-$100$ ($p = 0.000$), but no difference between $100$-PWC vs. SP-$100$ ($p = 0.962$).

For the storage decision, the Kruskal-Wallis test indicates a statistically significant difference between the distances ($H(2) = 16.395, p = 0.000$), with a mean rank of 23.82 for SP-$100$, 41.53 for SP-PWC, and 21.66 for $100$-PWC. Repeated Mann-Whitney tests indicate differences between SP-$100$ vs. SP-PWC ($p = 0.002$), and SP-PWC vs. $100$-PWC ($p = 0.000$). No difference exists between SP-$100$ vs. $100$-PWC ($p = 0.815$).
6. Improve Individual Architectural Decisions

Figure 6.8. Distances between similarity matrices for the $100, PWC, and same priorities (SP) approaches for the two decisions.

This suggests that $100 offers a compromise between using SP and PWC. Different prioritization approaches produce priority values which impact the hierarchical clusters. To evaluate the impact, we compared similarity matrices for grids that use same priorities, pairwise-comparisons, and hundred-dollar prioritization approaches. SP and PWC result in most different matrices. The distance between matrices from using $100 and SP does not differ from the distance between matrices from using $100 and PWC.

6.4.5 Discussion

Results from the interview study with practitioners (in Section 6.3) indicated the need to prioritize concerns for the initial REGAIN approach. Also, concerns’ priorities are part of architectural knowledge, and implicit priorities increase the risk of architectural knowledge vaporization. Hundred-dollar and pairwise-comparisons are already used in other decision-making approaches (Falessi et al., 2011). Therefore, we conducted an experiment to compare the hundred-dollar and pairwise-comparisons approaches, to understand which prioritization approach to add to REGAIN.

In the experiment, we investigated performance, users’ perceptions, and impact on REGAIN output of the two approaches. On performance, we found out that the hundred-dollar test needs less time than the pairwise-comparisons,
and that the hundred-dollar scales better than the pairwise-comparisons approach. Users perceived the hundred-dollar as more attractive, easier to use and to learn than pairwise-comparisons. Finally, we found that prioritization approaches influence REGAIN output, and that the hundred-dollar approach offers a middle ground between the pairwise-comparisons approach and using same priorities for all concerns.

The results on performance and users’ perceptions can be explained by the steps of the two approaches. Pairwise-comparisons require comparisons among each possible pair of concerns, so it has an $O(n^2)$ order of complexity. In contrast, hundred-dollar requires direct indication of priorities. Therefore, if a person has a clear idea about priorities of concerns involved in an architectural decision, then the hundred-dollar approach captures priorities more efficiently than pairwise-comparisons, which also influences users’ perceptions. However, if a person does not have a clear idea about the priorities, then pairwise-comparisons might be more efficient, because it offers smaller steps than the hundred-dollar approach. In such situation, the small steps of the pairwise-comparisons approach might provide more value than the hundred-dollar approach.

Following this experiment, we recommend practitioners with little experience in capturing concerns’ priorities for their decisions to use the hundred-dollar approach, because first-time users’ perceptions favor it over the pairwise-comparisons approach. Also, we recommend practitioners who have a clear idea of the concerns’ priorities to use the hundred-dollar approach, because of its better performance. However, we recommend that practitioners who need help clarifying priorities to use the pairwise-comparisons approach, because it offers smaller steps and a consistency ratio.

In practice, the number of items (i.e. concerns) to be prioritized can be larger than the number of items we used in this experiment. This raises questions on the scalability of the hundred-dollar approach for many items to be prioritized. For example, Regnell et al. (Regnell et al., 2001) present a study with practitioners who prioritized 58 items. In addition, Berander and Jönsson (Berander and Jönsson, 2006) mention that prioritizing 200 items happens in practice. To cope with the large number of items, researchers propose two approaches. The first approach is to increase the number of points (or dollars), such as using $100,000 for prioritizing the 58 items in (Regnell et al., 2001).
The second approach is to define various levels of abstraction that enable high-level items to be refined into more detailed items, thus creating hierarchies of items in which the hundred-dollar prioritization takes place at each level of the hierarchy. This extension of the hundred-dollar approach is called hierarchical cumulative voting. Berander and Jönsson (Berander and Jönsson, 2006) present hierarchical cumulative voting in detail. Thus, according to (Berander and Jönsson, 2006; Regnell et al., 2001), the hundred-dollar approach scales better than the pairwise-comparisons approach even for large number of items to be prioritized.

There are tradeoffs for choosing which prioritization approach to add to the REGAIN approach. Although the hundred-dollar approach has better performance and users’ perceptions, the hundred-dollar approach does not address potential prioritization judgment errors, as it does not help identify prioritization inconsistencies. This means that the hundred-dollar approach might be less reliable than the pairwise-comparisons approach. However, the experiment results on REGAIN output suggest that the hundred-dollar approach offers results similar to the results from the pairwise-comparisons approach, which indicates the reliability issue is not critical. Therefore, we consider that the hundred-dollar approach offers a good tradeoff for REGAIN concerns’ prioritization.

These findings are in line with results from the requirements engineering domain. Karlsson et al. found out that the pairwise-comparisons approach is the most trustworthy out of six prioritization approaches, because of its inclusion of a consistency check, which is critical in addressing potential human judgmental errors (Karlsson et al., 1998). Berander and Svahnberg noticed that requirements engineers might prefer the hundred-dollar approach over the pairwise-comparisons approach, because the former is more straightforward (Berander and Svahnberg, 2009).

The findings from this experiment are important for several reasons. First, we obtained empirical evidence for the comparison between the hundred-dollar and pairwise-comparisons approaches. The evidence increases our understanding of prioritization approaches for architectural decisions, and removes speculations on the characteristics of the two approaches. Second, since REGAIN is based on the Repertory Grid technique, this experiment encourages other researchers to add a concerns prioritization step when using
6.5. Validity Threats

We discuss internal, construct, conclusion, and external validity threats for the interview study in Section 6.3 and the experiment in Section 6.4, using the recommendations from Jedlitschka et al. (Jedlitschka et al., 2008).

6.5.1 Interview Study Validity Threats

Internal validity are not applicable for this study, since we do not try to establish a causality relationship.

Construct validity threats may originate from how the study design reflects the studied constructs (e.g. advantages and disadvantages of the REGAIN approach) (Wohlin et al., 2012). To avoid such threats, we allocated enough time to explain the REGAIN approach to the practitioners. We used it on a toy example to make sure that practitioners understood the approach. Also, we selected practitioners who took architectural decisions in the industry, to ensure that they could provide relevant feedback. Moreover, we avoided using a single type of measures (the post-questionnaire), by also using semi-structured interviews, so that practitioners could elaborate on their perspectives.

Another source of construct validity threats is the behavior of the participants and researchers (Wohlin et al., 2012). For example, some participants might change their behavior during a study. We used recommendations from (Hove and Anda, 2005) on creating a non-judgmental and comfortable atmosphere during the interviews, so that participants would not change their behavior.
Still, researchers might influence subconsciously study results because of their own expectations, which might change researchers’ behavior (e.g. extra enthusiasm) (Wohlin et al., 2012). To mitigate this problem, we asked another researcher to conduct the sessions with the architects. This additional researcher was not involved in the previous studies on the REGAIN approach, and had no interest in obtaining either positive or negative feedback on the approach.

**Conclusion validity** - in this study, we used descriptive statistics. Thus, we did not run into issues on low statistical power. Regarding measurement, we used a post-questionnaire validated in a previous study, to ensure reliability of measures, and we used a non-parametric statistical test, which has broader assumptions on the distribution of the data, compared to parametric statistical tests. Furthermore, two researchers were involved in the content analysis of the interviews transcriptions, who agreed on the data interpretation.

**External validity** - to address this validity threat, we recruited architects who worked in a wide variety of domains (see Table 10.2 in the Appendix). Also, the architects used the REGAIN approach on real-world architectural decisions from their industrial practice. To further generalize the results, we needed to make sure that enough architects participated in the study. Unfortunately, architects have busy schedules and limited time to participate in studies, which lead to difficulties in recruiting architects for this study. However, we reached data saturation from the sessions with the sixteen architects, in the sense that the last two sessions brought no extra insights compared to the previous sessions.

### 6.5.2 Experiment Validity Threats

We addressed **internal validity** threats as follows. To avoid the instrumentation validity threat, we piloted the experiment on three persons with software architecting background. We used their feedback to improve the quality of the experimental package. For example, we increased the readability of the paper forms. Another instrumentation validity threat is the use of self-reporting for time needed by participants for the prioritization tasks. We addressed it by reminding participants to record the start and end time for each task during the experiment. Social threats like demoralization of participants influence negatively study results. To avoid that, we made sure that
participants received enough directions for the tasks. Furthermore, we asked participants to rate some statements about the session. Results are summarized in Table 6.4. Overall, subjects considered that directions were clear and they had clear ideas about their tasks. Most subjects were neutral or positive on enjoying the session. We consider these results suggest a good level of subjects’ participation and engagement in the exercise, with a positive impact on the internal validity of the experiment.

Table 6.4. Number of session-related answers in the post-questionnaire.

<table>
<thead>
<tr>
<th>Statement</th>
<th>Strongly disagree</th>
<th>Disagree</th>
<th>Neutral</th>
<th>Agree</th>
<th>Strongly Agree</th>
</tr>
</thead>
<tbody>
<tr>
<td>The directions for the exercise were clear</td>
<td>1</td>
<td>-</td>
<td>1</td>
<td>20</td>
<td>8</td>
</tr>
<tr>
<td>I had a clear idea on what I had to do in the exercise</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>20</td>
<td>7</td>
</tr>
<tr>
<td>I enjoyed doing the exercise</td>
<td>1</td>
<td>2</td>
<td>12</td>
<td>11</td>
<td>4</td>
</tr>
</tbody>
</table>

We addressed the mortality validity threat by integrating the study with the software architecture course, so that subjects joined it voluntarily for the educational value. No participant dropped out from the experiment, although we had a few cases of incomplete data. To avoid selection threats, we randomized the distribution of participants to the two groups. Furthermore, we checked that subjects have similar background on decision-making, e.g. exclude persons with background on prioritization approaches or the repertory grid technique.

We addressed construct validity by defining concrete metrics for operationalizing constructs in our experiment, such as the scalability ratio. Another risk was treatments interaction, because each participant used two prioritization approaches. We addressed it partially by randomly assigning participants to two groups that received the treatments in different orders. Social threats for construct validity refer to the altered behavior of subjects when participating in an experiment. We reduced social threats by integrating the study with the course, and eliminating any impact on participants’ grades.
Additionally, participants did not know our hypotheses, which could have altered their behavior.

We addressed conclusion validity by using non-parametric tests, which make fewer assumptions about the distribution of data, such as Mann-Whitney. However, this experiment included around 30 data points, which may negatively impact the statistical power of tests. Furthermore, subjective measures such as perceptions (e.g., ease to use) tend to be less reliable than objective ones (e.g., time). Thus, we piloted the post-questionnaire to ensure that participants understand the items in the post-questionnaire.

We addressed external validity threats by ensuring that the experimental tasks did not require a specific level of software architecture experience, and no professional-only knowledge was needed for the experimental tasks. Furthermore, the participants had some practical experience in software development (average of 3.6 years) and software architecture (average of 1.2 years). Kitchenham et al. regard students as relatively close to the population of interest, because they are the next generation of software professionals (Kitchenham et al., 2002). Finally, increasing commitment level of participants in a study benefits applicability of study results to professionals (Berander, 2004). Since we integrated the experiment timeline with the course schedule, participants in this experiment had a high commitment level. Therefore, we consider the results of the experiments are applicable to software architects of various levels of practical experience.

### 6.6 Related Work

The main benefit of REGAIN is reducing the vaporization of architectural knowledge, through capturing (or acquisition) of architectural knowledge. REGAIN is based on the Repertory Grid technique, which has been used successfully for the acquisition of other types of design knowledge outside the software domain. Boose et al. (Boose et al., 1990b) used the Repertory Grid technique to capture design knowledge for different types of products from the Boeing company. Hassenzahl and Wessler (Hassenzahl and Wessler, 2000) used the technique to capture design knowledge of industrial products at the Siemens company. Overall, these studies indicate the value of the Repertory Grid technique for capturing design knowledge, because the technique
6.6. **Related Work**

encourages capturing the personal perspectives of the designers, and such perspectives are a very important of design knowledge.

Software architecture related applications of the Repertory Grid technique include the following studies. De Boer and van Vliet (de Boer and van Vliet, 2008) used the Repertory Grid technique to compare the mental models of auditors about existing documentation against the semantic structure of the documentation. This was to discover architectural knowledge in documentation. Shaw and Gaines (Shaw and Gaines, 1996) argue for closer collaboration between the knowledge engineering and software engineering community. Therefore, they present the Repertory Grid technique that helps capturing knowledge on requirements (Shaw and Gaines, 1996). Niu and Easterbrook (Niu and Easterbrook, 2007) use the Repertory Grid technique to capture the knowledge of stakeholders about requirements, and to clarify inconsistencies in how stakeholders use terminology for stating requirements.

Decision-making applications of the Repertory Grid technique include the following studies. Boose (Boose, 1989) presents decision support tools that are based on the Repertory Grid technique, with applicability in various domains. Scheubrein and Zionts (Scheubrein and Zionts, 2006) propose a decision-making approach based on the Repertory Grid technique, with tool support in the form of an Excel spreadsheet. Castro-Schez et al. (Castro-Schez et al., 2005) propose an extension of the Repertory Grid technique for decision-making support for managers.

There is much interest from the software architecture community in architectural decision-making and capturing techniques. Falessi et al. (Falessi et al., 2011) indicate three main types of techniques for decision-making:

1. Naturalistic decision making (or keeping the first available alternative)
2. Selecting among a finite number of alternatives (or multi-attribute techniques for decision-making)
3. Selecting among an infinite amount of alternatives (covered by optimization research, such as multi-objective techniques for decision-making)

In practice, the most common type of techniques is selecting among a finite number of alternatives (Falessi et al., 2011). Falessi et al. (Falessi et al., 2011) compare fifteen such architectural decision-making techniques (e.g. CBAM
Improve Individual Architectural Decisions

(Kazman and Klein, 2001), and notice that each technique involves tradeoffs, as there is no perfect technique. Techniques for capturing architectural decisions include decision views (Kruchten et al., 2009; van Heesch et al., 2012), templates (Tyree and Akerman, 2005), and tools (e.g. (Capilla et al., 2010)). Tang et al. compare five tools for capturing architectural decisions (e.g. Archium, PAKME) (Tang et al., 2010).

Prioritization approaches received much attention in the software engineering community, since practitioners need to prioritize concerns, stakeholders, features, and requirements. Prioritization approaches can be grouped in ordinal scale and ratio scale approaches (Berander and Jönsson, 2006).

For the ordinal scale prioritization approaches, there is an order among prioritized items (e.g. by assigning importance levels), but no arithmetic operations are possible with the values for priorities. Such a prioritization approach is included in RFC 2119 (Bradner, 1997), which recommends specific keywords (i.e. must/should/may) for prioritization. In addition, IEEE 830-1998 (830-1998, 1998) recommends prioritizing by assigning one of following keywords: essential/conditional/desirable. The updated version of IEEE 830-1998 is IEEE 29148-2011 (29148, 2011), which recommends three types of prioritization: either using shall/will/should/may keywords, one-to-five scale, or high/medium/low keywords.

For the ratio scale prioritization approaches, all arithmetic operations are possible, ratios among values and interval sizes are relevant. This means that values obtained from using a ratio scale prioritization approach are more precise than values obtained from ordinal scale prioritization approaches (Berander and Jönsson, 2006). Pairwise-comparisons and hundred-dollar are the most used ratio scale prioritization approaches (Berander and Jönsson, 2006).

There are important differences between ordinal and ratio scale prioritization approaches, which influenced our choice on which type of approach to use for REGAIN. Although ordinal scale approaches are standardized (830-1998, 1998; 29148, 2011; Bradner, 1997) and very common in practice (Berander and Jönsson, 2006), such approaches carry the risk of knowledge vaporization. Results from previous studies (Berander and Jönsson, 2006; Karlsson, 1996; Karlsson et al., 1998) indicate that using a ratio scale approach captures more
knowledge than using an ordinal scale prioritization approach, such as fine grained differences in the priorities. Since REGAIN aims at capturing knowledge on architectural decisions, ratio scale approaches are better suited for REGAIN, instead of ordinal scale prioritization approaches.

6.7 Conclusions

In this chapter, we address two problems. First, REGAIN or the Repertory Grid Technique had not been used with practitioners to make and capture real-world architectural decisions. Second, potential improvements of REGAIN were not known.

To solve these problems, we conducted an interview study with 16 practitioners. From the interview study, we found advantages, disadvantages, and improvement opportunities for REGAIN. A critical improvement was to prioritize concerns for REGAIN. To select a prioritization approach, we conducted an experiment with 30 graduate students in which we compared the hundred-dollar and pairwise-comparisons approaches.

The results of the experiment indicate that the hundred-dollar approach has better scalability and needs less time than the pairwise-comparisons approach. Also, users’ perceptions favored the hundred-dollar approach: participants considered that the hundred-dollar approach was easier to use, learn and more attractive than the pairwise-comparisons approach. Moreover, results suggest that prioritization approaches impact REGAIN output, and that the hundred-dollar approach provides a ‘middle ground’ between using the same priorities and the pairwise-comparisons approach.

Following these studies with practitioners and students, we implemented tool support for REGAIN, which we report in Chapter 8. Such dedicated tool support reduces the effort from practitioners to use REGAIN. Furthermore, in the next chapter, we propose and validate a group architectural decision-making process that extends REGAIN.

6.8 Acknowledgments

We thank Konstantinos Tselios for his help in conducting the studies, and participants for their efforts to contribute in the studies.
6. Improve Individual Architectural Decisions