ABSTRACT
Visualization of architectural design decision (ADD) and its rationale, as a kind of traceability information, is supposed to facilitate the understanding of architecture design and the reasoning behind the design rationale, which is supposed to improve the architecting process and gets better architecture design results. But the lack of empirical evaluation that supports this statement prevents industrial practitioners from using ADD visualization in their daily architecting activities. In this paper, we conducted a controlled experiment to investigate how visualization of ADD can improve the understanding of architecture design by using Compendium, an open source tool for rationale visualization. The preliminary results show that visualization of ADD and its rationale does not increase the total time for reading architecture document (including visualization diagrams of ADDs) plus designing task, and it improves the understanding of architects on existing architecture design with better new design results.

Categories and Subject Descriptors
D.2.11 [Software Engineering]: Software Architecture - Languages, I.6.8 [Computing Methodologies]: Simulation and Modeling - Visual

General Terms
Design, Human Factors.

Keywords
Software architecture, design rationale, architectural design decision, design decision visualization, controlled experiment.

1. INTRODUCTION
Visualization in computer graphics is a technique for creating images, diagrams, or animations to communicate a message, which is normally not easy to describe and understand in text. In a similar way, visualization of architectural design decision (ADD) is a technique for creating diagrams to communicate an architecture design, which contains not only the design artifacts themselves, but also the reasoning behind the design rationale, i.e., “why a particular design solution is the way it is” [1]. Understanding correctly and unambiguously the design rationale underlying an architecture design is critical for architects to modify existing design or to make new design according to e.g., new requirements in architecting [2].

There are many existing tools that support ADD visualization, like Archium, Knowledge Architect, Kruchten’s ADD Ontology Tool, Ontology-Driven Visualization Tool, PAKME, and AREL, etc. (a comprehensive survey can be found in [4]). To the best of our knowledge, design rationale (e.g., arguments, pros, cons, and decision drivers) visualization support is missing in most of existing ADD tools [3]. Furthermore, ADD visualization tools are normally developed by individuals or organizations based on their specific ADD models, and only support the visualization of ADDs that follow the proprietary ADD model, which hinders the communication, usage, and share of ADDs across organizational boundaries.

To provide ADD visualization support, especially design rationale visualization, in architecting process, we proposed to use Compendium\(^1\), an open source tool that provides notations for rationale visualization, to visualize ADD [3]. Meanwhile, to support general ADD visualization, we define the mapping relationships from ADD major elements [5] to IBIS (Issue-Based Information System) concepts [6] that Compendium notations are based on, so that ADDs in various ADD models can be readily visualized in Compendium [15]. We also demonstrated several examples in our previous work [3] about how to use ADD visualization in Compendium to assist architecting process, including architectural analysis, synthesis, and evaluation.

In architecture design, an ADD provides the traceability and dependency information from problem space to solution space,
i.e., from requirements to architecture design [16]. Traceability between requirements and ADD can result in better understanding of why an ADD was taken, and traceability between ADD and architecture design can lead to better understanding of how an ADD affects the system implementation [14]. Visualization of ADD and its rationale is a kind of traceability visualization that is supposed to facilitate the understanding from requirements to architecture design and the reasoning behind the design rationale.

Compendium, as a promising tool for general ADD rationale visualization, still lacks of empirical evaluation on its usefulness in architecting activities, which prevents industrial practitioners from using ADD visualization in architecting. In order to investigate the statement that “visualization of ADD and its rationale can facilitate the understanding of architecture design and the reasoning behind the design rationale, which consequently improves the architecting process and gets better design results. [3]”, we conducted a controlled experiment to investigate this statement quantitatively in two aspects: the time used to complete a given architecture design task, and the correctness of understanding architecture design.

The rest of this paper is organized as follows: Section 2 briefly summarizes related work on ADD visualization and rationale visualization in architecture design and requirements engineering. Section 3 motivates the research questions and derives research hypotheses. Section 4 describes the experiment design and the experiment materials with the threats to the validity of the experiment results. Section 5 reports the experiment results and accepted hypotheses. We conclude this paper with future work directions in Section 6.

2. RELATED WORK
This section summarizes related work on ADD visualization and rationale visualization in software development, which has significant impact on the rationale visualization of ADD using Compendium.

2.1 ADD Visualization
As we mentioned in the previous section, there are many existing tools that support ADD visualization, but few of them explicitly support rationale visualization of ADDs. These tools are based on proprietary ADD models, and they are competent for their specific ADDs visualization, but are incompetent as a general ADD rationale visualization tool for communicating architecture design and design rationale across system boundaries. Our previous work in [3] presented a detailed description and comparison of these tools for ADD visualization, and the motivation for using Compendium, a general ADD rationale visualization tool.

2.2 Rationale Visualization
Visualization technique has been widely employed in software development, and the typical example is modeling in software design, for example, UML that provides a set of graphic notations to create visual models of software-intensive systems. Rationale was historically proposed in the context of software design as means of presenting the “why” of a design (decision), and is often documented in text following certain rationale templates or conceptual models (e.g., IBIS [6], QOC, DRL). Effective design reasoning is an important part and skill in software design [20]. While rationale visualization, as an emerging technique to visualize the reasoning process behind the design rationale, has been used in software development, including requirements engineering and architecture design.

Lopez et al. proposed to use Semantic Web techniques to visualize and compare architecture rationale in order to facilitate the review, comparison, and reuse of rationale [7]. This rationale visualization approach focuses on automated reuse of design rationale with the support of an underlying Non-functional requirements and Design Rationale (NDR) ontology, while our approach using Compendium pays special attention to the usability of creating ADD visualization, and easy understanding and communication of architecture design and design rationale by architects.

Rationale visualization also contributes to requirements engineering, which is a closely-related activity to architecture design in the lifecycle of software development. Sellier and Mannion proposed to visualize the inter-dependencies between requirement selection decisions in order to understand the impact and reasoning of variable requirements in product line systems [8]. Ugai et al. developed a tool to visualize stakeholders’ interest of concern, which is the original source and rationale of why a requirement is proposed. This concern visualization helps to detect the missing links between requirements and stakeholders, and prompts to unify the requirements for the same type of stakeholders [9]. These approaches provide meaningful insights about how to visualize ADDs and use ADD visualization in architecting process.

3. RESEARCH QUESTIONS
In this paper, we investigate the usefulness of ADD visualization with Compendium for architecture design in two aspects, i.e., the time used to complete architecture design, and the correctness of understanding architecture design.

We present the objective of this controlled experiment according to the template suggested in [10]: Analyze Compendium as an ADD visualization tool for the purpose of improving (with respect to) the understandability of architecture design from the point of view of (experimental Subject) in the context of (experiment Material). In this controlled experiment, we plan to compare the understandability one has of an architecture design when using Compendium for visualizing ADDs and their rationale as opposed to without using ADD visualization.

To this end, we need a way to quantify the understandability someone has of an architecture design. Because understanding architecture design is crucial for an architect’s ability to perform other architecting activities, we can indirectly measure the

2 The term design rationale is most often used in the literature.
understandability one architect has of an architecture design by evaluating how well he/she performs architecting activities in architecting process. Based on this assumption on the relationship between understandability and architecting activities, the following research questions are formulated:

- **Q1**: Does using Compendium for visualizing ADDs and their rationale reduce the time that is needed to understand architecture design in architecting process against without using it?
- **Q2**: Does using Compendium for visualizing ADDs and their rationale increase the correctness of understanding architecture design in architecting process as opposed to not using it?

Associated with these two research questions, there are two null hypotheses formulated as follows:

- **H1**: Using Compendium for visualizing ADDs and their rationale does not affect the time needed to understand architecture design in architecting process.
- **H2**: Using Compendium for visualizing ADDs and their rationale does not affect the correctness of understanding architecture design in architecting process.

The alternative hypotheses that we use in this controlled experiment are the following:

- **H1**: Using Compendium for visualizing ADDs and their rationale increases the time needed to understand architecture design, but reduces the time needed on architecture design task in architecting process.
- **H2**: Using Compendium for visualizing ADDs and their rationale improves the correctness of understanding architecture design in architecting process.

### 3.1 Variables

Following the standard practice of empirical software engineering [10], the independent variable in this experiment is the (none) use of Compendium during conducting the controlled experiment. The experiment also consists of two dependent variables: the time spent by an experimental subject (e.g., a software architect) on understanding architecture design in architecting process, and the correctness of this understanding quantified by applying the evaluation criteria (see Section 5.3) on the experiment design results submitted by the subjects. The details of the experiment design and process are presented in the next section.

### 4. EMPIRICAL STUDY

This section presents the experiment design and process, employed experiment material, experiment participants (i.e., subjects), and the threats to the validity of experiment.

#### 4.1 Experiment Design and Process

The usefulness of ADD visualization is quantified in two aspects: the time used to complete architecture design, and the correctness of understanding architecture design. Meanwhile, to make this experiment performable, we asked the subjects to complete the experiment tasks in a limited time, i.e., the architecture design results should be submitted by subjects within one and a half hour. Considering these issues, the steps of this experiment are specified below:

1. Divide the experimental subjects randomly into two groups (Group A and B);
2. Select an appropriate sub-system architecture design and related architecture description from an architecture document, and the selected sub-system design is understandable in about 45 minutes;
3. Provide both groups a new requirement about the sub-system, and ask the subjects to make new design to satisfy this new requirement based on their understanding of the existing design;
4. Group A is presented the architecture design document with ADD visualization in Compendium to make the new design according to the new requirement (Note that, we didn’t introduce a tutorial session on Compendium to the subjects because we thought that the Compendium notations are straightforwardly understandable. We only add some labels to the Compendium notations in ADD visualization, e.g., Design Issue, Positive Factor, see the experiment material for Group A in Section 4.2 for a detailed understanding);
5. Group B is presented the architecture design document without ADD visualization in Compendium to make the new design according to the new requirement;
6. Ask the two groups to submit the design results within one and half hour, with a suggestion of using 45 minutes for reading the document, and 45 minutes for completing the design task;
7. Record the actual time spent by subjects on reading architecture document and designing architecture;
8. Evaluate quantitatively the correctness of understanding architecture design by the design results submitted by the two groups (see Section 5.3) and finally analyze the experiment results using t-test (see Section 5.1).

<table>
<thead>
<tr>
<th>Group A (using ADD visualization)</th>
<th>Group B (without using ADD visualization)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step</strong></td>
<td><strong>Step</strong></td>
</tr>
<tr>
<td>Understand the Existing Design in the Architecture Document</td>
<td>Understand the Existing Design in the Architecture Document</td>
</tr>
<tr>
<td>Make New Design according to a New Requirement</td>
<td>Make New Design according to a New Requirement</td>
</tr>
<tr>
<td><strong>Time</strong></td>
<td><strong>Time</strong></td>
</tr>
<tr>
<td>about 45 min</td>
<td>about 45 min</td>
</tr>
</tbody>
</table>

**Figure 1. Experiment Design and Process**

- **Figure 1.** Experiment Design and Process
Figure 1 illustrates the experiment design with a suggested time-interval (45 minutes) for executing the individual tasks.

4.2 Experiment Material

The experiment material (i.e., the sub-system architecture design and related architecture description) is selected from the architecture document of Cyber Video System (CVC), which is a system that provides digital media service over a satellite connection to consumers (e.g., movies). The reason that we select this system as the experiment material is that: this system is easily understandable by the subjects without the prerequisite of any specific domain knowledge, and the scale and complexity of the architecture design are appropriate.

The experiment documents for Group A

The experiment documents for Group A (architecture design document with ADD visualization in Compendium) and Group B (architecture design document without ADD visualization in Compendium) are both available online for readers reference. The only difference of the software architecture (SA) documents used for Groups A and B is the ADD visualization part by Compendium. The SA document for Group A has this part, and Group B has not.

The experiment results for analysis (i.e., the architecture design results by the subjects) in various formats (including Office Visio, Word, and other boxology formats) by Group A and B are also available online.

4.3 Experiment Tool

Compendium tool is a semantic hypertext concept mapping tool that supports Issue-Based Information System (IBIS), an argumentation based approach for design rationale representation [17]. Compendium consists of six main notations. Table 1 shows these notations. Traceability in ADD means that the origins and targets of an ADD and its rationale are traceable. Compendium provides traceability support among requirements (Questions), Decisions, and alternatives (Answers) of a decision. In Compendium tool, the traceability links from design rationale (i.e., ADD) to requirements and architecture design are all converted and provided as links. These links are identified by the architects who use Compendium tool and they are manually recorded by architects. There is currently no link mechanism/support from the ADD visualization in Compendium to requirements in requirements tools (e.g., Rational DOORS) or components in architectural modeling tools (e.g., Rational Software Architect). But it is possible to provide such traceability function in the next step since Compendium is an open source tool implemented in Java.

<table>
<thead>
<tr>
<th>Table 1. Compendium Tool Notations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Notations</td>
</tr>
<tr>
<td>Question (📸)</td>
</tr>
<tr>
<td>Decision (ߒ)</td>
</tr>
<tr>
<td>Answer ( (${})</td>
</tr>
<tr>
<td>Pro Argument (🗣)</td>
</tr>
<tr>
<td>Con Argument (🗣)</td>
</tr>
<tr>
<td>Justification (🗣)</td>
</tr>
</tbody>
</table>

Figure 2 presents an example (the example that we used in our controlled experiment) of the visualization of an ADD and its related entities in Compendium. The design problem (Question (📸)) to be addressed is “Choosing the security mechanism of the Content Storage on CVC side”. The Decision (Thunk) “Media Content in CVC Content Store is kept encrypted” gets selected with two Alternatives (🗣): “Online encrypted” and “Offline encrypted”. Every Alternative has positive and negative effects to the design problem, which are represented by Pro (📸) and Con (🗣) Argument. With these notations, the rationale of an ADD can be easily visualized using Compendium in a straightforward way. For example, the positive effect (🗣) of Alternative (🗣) “Offline encrypted” is that “Content security is increased in CVC Content Store”, while the negative effect (🗣) of this Alternative is that “Each movie has its own key”. Users can easily follow the visualization links between rationale entities through an ADD in Compendium, and update their decisions (e.g., when design context changes).

4.4 Experimental Subjects

There are 10 subjects who participated in this controlled experiment: 6 of them are from academia (researchers at universities), and 4 of them are from industry (developers and architects in software companies). The 10 subjects have an average of 5.6 years experience on software design and development. The 10 subjects are grouped into Group A and B evenly across the groups based on their expertise and background (i.e., from academia or industry). Table 2 shows the characteristics of the experimental subjects.

4.5 Threats to Validity

Due to the limitations of this controlled experiment, there are several threats to the validity of the experiment results. We classify threats into internal validity, which describes the cause-effect relationship between the independent and dependent variables; external validity, which refers to the ability to generalize the results of the experiments in different contexts; and conclusion validity, which refers to the degree to which the conclusion that is being achieved from the experiment is reasonable [18].

---

3 This document can be downloaded at http://www.shark-workshop.org/temp/GroupA.pdf
4 This document can be downloaded at http://www.shark-workshop.org/temp/GroupB.pdf
5 This document can be downloaded at http://www.shark-workshop.org/temp/DesignResults.zip
4.5.1 Internal Validity

One of the usual internal validity threats about the experimental subjects is that the knowledge of subjects may differ between the test group and controlled group, and possibly affect the quality of experiment results. To alleviate this threat, the subjects were evenly distributed across the groups based on their expertise and background (e.g., from academia and industry, see Table 2). The subjects participated in this study have enough competency to do the experimental task because most of them work on software (architecture) design and development, including academic researchers and industrial practitioners. They also don’t know the goal of this experiment before performing the experiment.

Researchers’ bias as a validity threat can affect the results of the study. This threat exists in our study because the answers of the subjects (see Section 5.3) were graded by three researchers (i.e., the authors of this paper). Double-blind experiment can be used to eliminate the possibility of this threat, but it is difficult to introduce double-blind experiment in the evaluation of answers of subjects in this experiment because we (as the researcher) have to assign the experiment tasks to the experimental subjects by ourselves. But to alleviate this threat, an answer evaluation model (the criteria introduced in Section 5.3) was designed in advance that clearly states the required elements and corresponding points for each element. Thus the researchers have graded the answers of subjects according to this answer model.

4.5.2 External Validity

The number of subjects (10) is limited due to the criteria for selecting the experimental subjects and the resources we have: we only select experienced software developers and architects. The considerable effort for performing this experiment is another difficulty to include more subjects since this experiment is not related with their job. We plan to repeat this experiment with more subjects in our future work by introducing ADD visualization using Compendium in some real projects in software companies.

The number of architecture evaluators (3, i.e., the authors of this paper) for evaluating the design results is quite limited since experienced evaluators are scarce resources. The involvement of more evaluators can reduce the bias in the evaluation of architecture design results. We try to alleviate this threat by intensive discussions among evaluators to get consensus during evaluation.

The size of the architecture design task used in this experiment is relative small due to the time limitation of the experiment. We plan to extend the task of this experiment to a real architecture project (for example, a software architecture course project or an architecture project in a software company) that covers the integrated architecting process (i.e., architectural analysis, synthesis, and evaluation) and includes the collaboration and communication between architects and involved stakeholders, in order to investigate the systematic usage of ADD visualization in architecting.

4.5.3 Conclusion Validity

In this experiment, we use t-test (see Section 5.1), a parametric test requiring that each of the two populations being compared should follow a normal distribution. This condition can be tested using, e.g., Shapiro–Wilk test [19], and Shapiro–Wilk test succeeded for time and correctness experimental results, which means that t-test can be used.

5. EXPERIMENT RESULTS

This section summarizes the results of the controlled experiment. We first introduce the t-test statistical method in Sections 5.1, and discuss the time and correctness results respectively in Section 5.2 and 5.3.

5.1 T-test

T-test is a statistical hypothesis test method that assesses if there is a significant difference between the means of two groups [12]. In statistical tests, a p-value represents the probability that a hypothesis test is significantly different from the null hypothesis. The p-value shows how likely it is that a test group is significantly different from a controlled group. To analyze the controlled experiment results, we applied the t-test at a significance level of 95% (\( \alpha = 0.05 \)) for statistical evaluation, which means that statistical significance is attained in cases where the p-value is found to be lower than 0.05 or t-test is bigger than t-critical. The t-critical value is the cutoff between retaining and rejecting the null hypothesis. If the t-test value is bigger than the t-critical value, the null hypothesis is rejected; otherwise, the null hypothesis is retained. For this study, df (degree of freedom: the number of subjects in both groups minus 2) is 8 and the t-critical value is 2.31 if the alpha level is 0.05.
5.2 Time Results

We start by testing the null hypothesis $H_{10}$, which states that using Compendium for visualizing ADDs and their rationale does not affect the time needed to understand architecture design in architecting process.

The time aspect in this controlled experiment is divided into the time spent on reading the architecture document (time on reading document) and time on making the new design (time on design task).

Tables 3 and 4 show that Group A (using ADD visualization) spent more time on reading the document and less time on design task in comparison of group B respectively. We suppose that the additional time spent by the subjects of Group A is used for reading and understanding the design rationale diagrams (i.e., ADDs) visualized by Compendium.
Table 3. Experiment Results on Reading Document Time by Group A and B

<table>
<thead>
<tr>
<th>Group</th>
<th>n</th>
<th>Mean Time on Reading Document</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group A (using ADD visualization)</td>
<td>5</td>
<td>32 min</td>
</tr>
<tr>
<td>Group B (without using ADD visualization)</td>
<td>5</td>
<td>28.8 min</td>
</tr>
</tbody>
</table>

Table 4. Experiment Results on Design Task Time by Group A and B

<table>
<thead>
<tr>
<th>Group</th>
<th>n</th>
<th>Mean Time on Design Task</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group A (using ADD visualization)</td>
<td>5</td>
<td>23.6 min</td>
</tr>
<tr>
<td>Group B (without using ADD visualization)</td>
<td>5</td>
<td>35.4 min</td>
</tr>
</tbody>
</table>

The time duration spent for reading SA document and design task is not fully separated (like oil and water) because the subjects can sometimes go back to read the document again when they design, for example, to confirm/recall something which is not clear, or to verify new design with existing design. In such situation, the time results of “reading” and “design” provided by the subjects may not be accurate, because it is difficult to count how much time they spent on re-reading the document during the design activity. Therefore, we consider the total time (inducing reading SA document and design task) to investigate whether there is a significant difference in the total time spent between the Group A and B.

Table 5 indicates that on average the Group A (using ADD visualization) spent 13.94% less time for the total time compared to Group B. The t-test (p-value > 0.05 or t-test is less than t-critical) shows that there is not a significant difference in time spent on total time between the Group A and B. Since the analysis result by t-test is not significant difference, we accept null hypothesis H10 and it means that visualization of ADD and its rationale does not affect the necessity needed to understand architecture design in architecting process. Tang et al. have conducted similar experiment to investigate the application of a design reasoning process in two groups: test group (equipped with design reasoning) and control group (without design reasoning) [13]. Their results show that both groups took a similar amount of time to finish their tasks during software design, which is similar to our results of time aspect.

The conclusion is that using visualization of ADD and its rationale in architecting doesn’t increase the total time for reading and designing task.

Table 5. Descriptive Statistics of the Experiment Results on Total Time (including “reading” and “design”) by Group A and B

<table>
<thead>
<tr>
<th>Group</th>
<th>n</th>
<th>Mean Time</th>
<th>Diff.</th>
<th>Std. Dev.</th>
<th>p-val.</th>
<th>t-test</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group A</td>
<td>5</td>
<td>55.6 min</td>
<td>-13.94%</td>
<td>11.67</td>
<td>.425</td>
<td>.84</td>
</tr>
<tr>
<td>Group B</td>
<td>5</td>
<td>64.2 min</td>
<td></td>
<td>19.60</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

5.3 Correctness Results

This subsection presents the test of null hypothesis H20, which states that using Compendium for visualizing ADDs and their rationale does not affect the correctness of understanding architecture design in architecting process.

To evaluate the correctness of understanding architecture design, we consider 3 criteria that indirectly evaluate the design results by subjects as follows:

1) Can the design result satisfy the new requirement? This criterion is divided into 4 sub-criteria that address various aspects of the new requirement. Each of them has 0.25 point. For example, one sub-criterion is that “Checking whether the request ID is a user with Certification Authority”. A new design that can satisfy the new requirements is based on the correct understanding of existing design.

2) Has the design result any conflict with existing design? A design conflict refers to the conflict between the new architecture design and existing design. For example, in the existing design for requirement “Store Movie”, the movie content is encrypted before being stored in Movie Management System (MMS). If a new design for requirement “Handle Movie Request” doesn’t consider the decryption of movie content before sending to the movie requester, then there is a design conflict between the new design (for requirement “Handle Movie Request”) and existing design (for requirement “Store Movie”). If we identify a design conflict between new design and existing design, we give a -0.25 point to the score of the design result, since design conflict is a sign that demonstrates that the subject didn’t understand the existing design correctly.

3) Does the design result reuse any part of existing design? Effective and appropriate reuse of existing design artifacts is an important sign to show that the subject has a correct understanding of the existing design. If the subject uses a new component, we add 0.00 point to the score of the design result, and if the subject reuses an existing component appropriately, then we add 0.25 point to the score of the design result.

The correctness hypothesis is investigated using t-test based on these evaluation criteria and evaluation results. Table 6 shows that the mean score of the design results submitted by the subjects of Group A were 44% better than the design results of Group B, averaging 1.95 points compared to 1.35 points of Group B. Since the difference (p-value < 0.05 or t-test is bigger than t-critical) is clearly in favor of the ADD visualization group (Group A), it follows that hypothesis H2 can be rejected, and the alternative hypothesis H3 is accepted, which states that using Compendium for visualizing ADDs and their rationale can improve the correctness of understanding architecture design in architecting process.

Table 6. Descriptive Statistics of the Experimental Results on Correctness by Group A and B

<table>
<thead>
<tr>
<th>Group</th>
<th>n</th>
<th>Mean Score</th>
<th>Diff.</th>
<th>Std. Dev.</th>
<th>p-val.</th>
<th>t-test</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group A</td>
<td>5</td>
<td>1.95</td>
<td>+44%</td>
<td>.32</td>
<td>.035</td>
<td>2.53</td>
</tr>
<tr>
<td>Group B</td>
<td>5</td>
<td>1.35</td>
<td></td>
<td>.41</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
6. CONCLUSIONS AND FUTURE WORK

In software architecture community, ADD has been recognized as a first-class element in software architecture [2][11], but it still requires consolidated effort on how to employ ADD in daily architecting activities. Visualization of ADD provides an intuitive way for architects and concerned stakeholders to communicate, use, and share ADDs, and consequently facilitates the understanding of the architecture design that is based on the ADDs. The controlled experiment we conducted in this work indicates that the test group, which was equipped with ADD visualization technique, produced better architecture design results than the controlled group, and ADD visualization does not increase the total time for reading SA document and design task.

We outline our ongoing and future work in the following directions: (1) Conduct this controlled experiment with more experimental subjects and design results evaluators in order to address the limitations discussed in Section 4.5 and achieve more convincing experiment results. (2) Conduct a controlled experiment in an integrated architecting process, which covers all the architecting activities, including architectural analysis, synthesis, and evaluation. The purpose of this experiment is to investigate quantitatively the usefulness and effectiveness of ADD rationale visualization in a real architecting context (e.g., practical projects from software companies). (3) The prerequisite of using ADD visualization is to create or recover ADDs and the traceability links from and to an ADD from architecture documents, which requires considerable effort (including manual work with the support of ADD visualization tool). We plan to investigate the effort of creating, recovering, and visualizing ADD using Compendium for the cost-benefit analysis of using ADD visualization in architecting.

ACKNOWLEDGMENT

This work has been partially supported by the Natural Science Foundation of China (NSFC) under Grant No.60903034, QuASAK: Quality Assurance in Software architecting process using Software Architectural Knowledge. We would also like to thank all the participants of this controlled experiment, and comments for this work by Paris Avgeriou and Anton Jansen.

7. REFERENCES


