Architecture design in global and model-centric software development

Heijstek, W.

Chapter 6

Experimental Analysis of Textual and Graphical Representations for Software Architecture Design

In this chapter the results of a study on the use of software architecture documentation is described. First, the effectiveness of text-dominant versus diagram-dominant architecture descriptions are explored by means of an experiment. Second, developer characteristics that benefit architecture representation understanding are investigated.

This chapter is based on the following publication:

Werner Heijstek, Thomas Kühne and Michel R.V. Chaudron **Experimental Analysis of Textual and Graphical Representations for Software Architecture Design.** *In Proceedings of the 5th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM 2011) pages 167–176, Banff, Alberta, Canada*

6.1 Introduction

Software architecture documentation facilitates stakeholder communication and is instrumental in ensuring that essential design principles are adhered to by the source code. Software architecture documentation *"captures and preserves designer intentions about system structure, thereby providing a defense against design decay as a system ages, and*

it is the key to achieving intellectual control over the enormous complexity of a sophisticated system" (Hofmeister, 2000). However, software architecture and design knowledge management is challenging in the context of GSD (Ali et al., 2010). In such a scenario, complete and unambiguous architecture design documentation constitutes an indispensable complement to informal communication (Curtis et al., 1988, Lee et al., 2006).

However also in co-located development, using an iterative development process, reliable and effective documentation is highly desirable. While co-located teams better support spontaneous and informal communication, there is a danger that code is not implemented according to the principles as laid out in documentation artifacts. If the design documentation is not treated as the ultimate reference and does not succeed in allowing answers to common questions to be derived, there is a danger that the resulting system will be based on inconsistent interpretations and assumptions: *"The best architecture is worthless if the code doesn't follow it"* (Clements and Shaw, 2009). In their landmark study on the state of the practice in software architecture research, Shaw and Clements (2006) formulate some promising areas in which significant opportunities exist for new contributions in software architecture research. Among others, they discuss the need to find the right language to represent architectures and finding ways to assure conformance between architecture and code. Documentation of the architecture is also essential for maintenance activities which typically involve different engineers from the ones who originally developed the system.

In this chapter we report on an experiment we conducted on media type effectiveness for documenting software architecture designs. The outline of this chapter is as follows. Section 7.3 contains an overview of related work. The study objective is outlined in Section 6.3. In Section 6.4, the experimental design is explained and Section 6.5 contains an overview and discussion of the results. The threats to validity are discussed in Section 6.6. Finally, recommendations are given in Section 6.7 and Section 6.8 describes our conclusions and future work.

6.2 Related Work

The related work for this study spans different sub-fields of software engineering. In the following paragraphs we will discuss related work on software architecture representation in practice, quality of documentation, the use of UML for architectural representations, the use of design documentation, multimedia learning and related experimental analysis of software design representations.

6.2.1 Software Architecture Representation in Practice

In their survey of 11 industrial systems Soni et al. (1995) found that a combination of informal and semi-formal techniques was used to describe software architectures.

They found that informal diagrams, tables and natural language text with naming conventions are used to describe many of the software structures not described in functional decomposition diagrams. They note that, "even when a formal notation is used, it is often supplemented with informal and incomplete diagrams, in order to enhance the understanding of the formal model." Soni et al. did not find this surprising as rigorous architecture description techniques were not yet available at the time. However, it is common that the software architecture description of systems is informal and based on "boxes and lines" types of notation (Abowd et al., 1995, Soni et al., 1995). A recent study of 57 industrial software architecture documents (Heijstek and Chaudron, 2011) confirms that software architecture is still described using a variety of media without an apparent systematic approach to media usage. The limitations of this style of representation led to the taxonomic separation between software design and software architecture (Eden and Kazman, 2003). It also led to the development of several methods and frameworks for defining and representing architectures (e.g. Bachmann et al., 2000, IEEE, 2000, Bachmann et al., 2000, Clements et al., 2002, Jansen and Bosch, 2005, Taylor et al., 2009). It is, however, unknown which of these styles are more effective to allow developers to correctly understand the intended architecture. As a result, little is known about how to produce more effective documentation. Bengtsson and Bosch (1999) describe an industrial case in which they were involved in designing the architecture. They note that they, *"found it hard to capture the essence of the architecture."* They also note that only because of the co-located nature of the project, they were able to *"overcome the problems with the written documentation."*

Agile methods (Highsmith and Fowler, 2001) recommend to make "lean" documentation, suggesting that documentation should only include information that is used. But even such code-centric, light-weight methodologies employ a form of architectural documentation (Smith, 2001). Agile methods have been introduced in more rigorous methodologies such as RUP (Hirsch, 2002, Pollice, 2001), partly in an attempt to incorporate increased formality regarding documentation. Developers do seem to prefer less documentation. This is supported by the findings of Forward and Lethbridge (2002) and Lethbridge et al. (2003) who studied the use and usefulness of documentation. The authors find a preference for simple and powerful documentation and conclude that documentation is an important tool for communication, even if it is not up to date. A recent survey by Stettina and Heijstek (2011b) of 79 agile software development professionals in 8 teams in 13 different countries, found that the majority of agile developers find documentation important or even very important, but also that too little documentation is available in their projects.

6.2.2 Use of UML for Architectural Representations

Many current architecture description methods recommend the use of UML diagrams for representing a software architecture. Hofmeister et al. (1999) present results of

their action research study into using UML for representation of a system's architecture. They found that UML worked well for describing important aspects typically described in software architecture documentation (such as the static structure of the architecture) and not so well for constructs (such as protocols and a general sequence of activities). A more recent study of the suitability of using UML to model software architectures (Medvidovic et al., 2002) reports that, *"UML lacks direct support for modeling and exploiting architectural styles, explicit software connectors, and local and global architectural constraints."*

There is certainly no standard way of creating architectural diagrams with the UML. UML allows its users a large degree of freedom (Nugroho and Chaudron, 2008). Systems are generally modeled incompletely and varying levels of detail are applied (Lange, 2006). UML standards are often applied loosely (Lange et al., 2003). All these aspects have been found to negatively contribute to the quality of software (Nugroho and Chaudron, 2009).

Results from an experiment by Tilley and Huang (2003) suggest that the UML's efficacy in support of program understanding is limited by factors such as ill-defined syntax and semantics, spatial layout, and domain knowledge. This chapter contributes to the understanding whether UML diagrams fulfill the expectation to represent precise and effective architecture documentation.

6.2.3 Use of Design Documentation

An observational study by Dekel and Herbsleb (2007) found that software teams improvised representations, incurring orientation difficulties and leading to an increased reliance on memory. Documented designs were therefore not useful without additional contextual information. We have to assume that this might be true to some extent for documented design decisions in practice. Design decisions are often the result of one-on-one meetings (LaToza et al., 2006). A study involving interviews and a survey by Cherubini et al. (2007) found that many modeled design decisions are lost. Studies that find that developers avoid using design documents when possible (Herbsleb and Moitra, 2001, LaToza et al., 2006, Kraut and Streeter, 1995, Müller and Tichy, 2001) may be construed as an indication that we know little about how to produce the best possible software architecture design documentation. In their study of how software developers use diagrams in software documentation Hungerford et al. (2004) found that search patterns that rapidly switched between two different diagrammatic representations are most effective. They note that, *"these findings support the cognitive theory thesis that how an individual processes information impacts processing success."* Empirical studies show that developers mostly understand only "their" specific components of the application (Curtis et al., 1988). Holt (2002) even advocates a "law of maximal ignorance" which he summarizes as follows: *"Don't learn more [about an architecture] than you need to get the job done."* Due to time pressure, Holt notes, developers barely have enough time to get acquainted with a system. Scanniello et al. (2010) experi-

mentally evaluated the use of design documentation that outlines design patterns on maintenance activities performed on source code. They found that the effort and efficiency significantly improved when design pattern were properly documented and provided to the subjects. In their mixed-method study of software engineers in practice, Lethbridge et al. (2003) found that software documentation is frequently out of date, too voluminous, poorly written and unfathomable and that documentation processes ("much mandated documentation") can be inefficient and ineffective. They find that, "[a] considerable fraction of documentation is untrustworthy" but also that *"[A]rchitecture and other abstract documentation information is often valid or at least provides historical guidance that can be useful for maintainers."* More than 40 percent of respondents note that they find that software architecture documentation is rarely, if at all, updated after changes have been made to a software system. Most respondents agreed with the statement: "Documentation is always outdated relative to the current state of a software system." Lethbridge et al. conclude that we need to better understand the various roles of software documentation and more closely match our prescribed processes to fit those roles. We specifically focus on the role of architecture documentation to support implementation work. Observational studies show that developers use documentation as little as 3 percent of their time (Lethbridge et al., 2003). More recently, Stettina and Heijstek (2011b) studied the use of documentation in Agile teams and found that the majority of developers found documentation important to very important and that they also found that too little documentation was available. In addition, they found it difficult to locate this (internal) documentation.

6.2.4 Experimental Analysis of Software Design Representations

Various experiments have been conducted to investigate the efficacy of software design representations. These experiments concentrated on measurement of the strength of a particular representation type to convey certain design properties under certain circumstances. For example, Lange and Chaudron (2006) used an experiment to investigate how developers deal with inconsistencies in UML diagrams. They found that defects often remain undetected and cause misinterpretations. Gemino and Wand (2003) advocate the use of evaluating modeling techniques based on models of learning based on Mayer's cognitive theory of multimedia learning (Mayer, 2009). In a later study, Gemino and Wand (2005) compared two different visualizations of Entity-Relation Diagram (ERD) types (Chen, 1976). They found that, *"clarity within [a] model may be more important than the apparent complexity of [that] model when a model is used for developing domain understanding."* In this study, we compare textual and diagrammatic models as this combination is found to be most commonly used in industrial practice. Another relevant related experiment is the comparison of comprehensibility of UML class diagrams versus ERD in the context of comprehension, maintenance and verification by De Lucia et al. (2010). They found that using UML class diagrams, subjects scored higher on comprehension. In this experiment, we use UML diagrams.

Knodel et al. (2008) conducted an experiment to study the role of graphical elements in architecture representations. They found that specific visualizations, such as neighbor highlighting, and an information overlay panel had a significant impact on developer comprehension. In our experiment, we did not use an interactive architecture visualization tool and could therefore not incorporate their findings to increase the efficacy of our experimental material. Formal notations (such as the Object Constraint Language (OCL, Warmer and Kleppe, 1998)) can also be used to make software architecture documentation more unequivocal. In fact, Briand et al. (2005) found evidence that if developers are well trained OCL might be a more effective annotation for UML models than text. OCL is sparsely used in industrial practice.

6.3 Objectives

In this chapter, we address **RQ2** (Section 1.3). This question aims to find representation methods for software architecture design that are understood by developers in the context of GSD. Tilley (2009) concludes an overview of "findings and lessons learned related to documenting software systems with views from numerous projects spanning 15 years of research and practice" by noting: *"The question of when graphical documentation is more effective than other forms of documentation (e.g. textual), and for which types of users, remains open."* We describe our main objective according to the goal definition template provided in the GQM paradigm (Basili et al., 1994):

> We **ANALYZE** the effectiveness of diagrams and text for representing software architecture designs **FOR THE PURPOSE OF** improving the quality of architecture documentation **FROM THE PERSPECTIVE OF** communication between software architects and developers **IN THE CONTEXT OF** project-based custom software development.

We therefore pose the following research questions:

1. *Are diagrams using a visual notation better suited to communicate software architecture design than textual representations?*
2. *How do software developers comprehend software architecture representations?*
3. *How do developers deal with missing and conflicting information in software architecture representations?*
4. *To what extent do developers make assumptions or fill in gaps in software architecture representations?*

Diagrams are widely used during designing and it appears plausible that their visual representation is an effective medium to communicate software design. Diagrams contain the essential information in an easy to overview, easy to process, two-dimensional

arrangement. Albers (2004) asserts that diagrams should allow developers to find answers quickly because diagrams contain less noise. In particular architectural principles pertaining to the topology of an architecture should be obvious in a diagram whereas they will have to be inferred from text. Moreover, diagrams should transcendent socio-cultural and language idioms because they are largely independent from natural language. For all the aforementioned reasons it can be expected that developers may have a preference for diagrammatic as opposed to textual documents. Yet, to the best of our knowledge no empirical study has ever attempted to confirm or refute these assumptions. In this study, we focus on natural language text because it is most commonly used in industrial practice. The experiment was designed to test the following hypotheses:

H1$_1$ *Diagrams are better than text at conveying software design to software developers.*

H2$_1$ *Diagrams are better than text at conveying topology-related design information to software developers.*

6.4 Experimental Design

Potential methods for testing our hypotheses include user surveys and controlled experiments. We dismissed the first because it generally only applies when more is known with regard to the variables that need to be controlled. Moreover, there is an element of uncertainty as to whether users correctly evaluate which media type is more effective (e.g. due to subjectivity). An "in the field" study, observing software architects while they are performing actual work might yield reliable data. However, it is difficult to get access to companies for such an in-depth analysis.

Hence, we designed an experiment in which we measure media effectiveness by varying the media dominance during a series of design document presentations. We define media effectiveness as the extent to which a medium can convey the modeled design information it contains in such a way that a developer understands it correctly. Therefore, we measured media effectiveness on the basis of how well participants were able to extract the intended design information from two documents that described one architecture design. One document contained text and the other diagrams. The variables under study are summarized in Table 6.1.

During the experiment, we used three methods to collect data:

1. We used two questionnaires to obtain participant-specific information.

2. We filmed participants during a set of tasks to understand when and from which medium they obtained answers to our questions.

Table 6.1: *Experiment Variables*

		variable type	*values*	*source*
Question Characteristics				
1.	QUESTION NATURE	nominal (binary)	$\in$ {topological, non-topological}	experimental design
2.	MEDIA DOMINANCE	nominal (binary)	$\in$ {diagram, text}	experimental design
3.	ANSWER LOCATION	nominal (quaternary)	$\in$ {diagram, text, both, neither}	experimental design
Developer Characteristics				
4.	LINGUISTIC DISTANCE	ratio (continuous)	$\frac{1}{\text{language score}}$ (see Table 6.6)	preliminary questionnaire
5.	EXPERIENCE	ordinal (quaternary)	$\in \{1,2,3,4\}$ (see Par. 6.5.7)	preliminary questionnaire
6.	MODELING SKILL	interval (discrete)	$\in \{1,\ldots,7\}$	preliminary questionnaire
Developer Performance				
7.	MEDIA PREFERENCE	nominal (binary)	$\frac{1}{n}\sum_{i=1}^{n} q_i$_percentage_diagram (q for question)	analysis of video
8.	MEDIA SWITCHES	ratio (continuous)	$\frac{1}{n}\sum_{i=1}^{n} q_i$_switches	analysis of video
9.	TIME PER QUESTION	ratio (continuous)	$\frac{1}{n}\sum_{i=1}^{n} q_i$_total_time	analysis of video
10.	USED ONE MEDIUM	ratio (discrete)	$\sum_{i=1}^{n} q_i$_only_used_one_medium (true if q_i_perc_diagram $= 0 \vee 100$)	analysis of video
11.	ANSWERS CORRECT	ratio (discrete)	$\in \{0,\ldots,13\}$	analysis of video
12.	FELL FOR FALSE FRIENDS	ratio (discrete)	$\in \{0,1,2\}$	analysis of video
Developer Opinion				
13.	PERCEIVED EFFECTIVENESS OF MEDIA	interval (discrete)	$\in \{-6,\ldots,6\}$ perc_eff_diagram – perc_eff_text (both interval $\in \{1,\ldots,7\}$)	post-experimental questionnaire

3. We requested the participants to think out loud while answering questions about the system.

Filming the participants had several benefits. First, we obtained a wealth of information regarding participant behavior during the experiment. We could, for instance, determine which media type was consulted first, last, with what frequency and how often participants switched between media types. The recordings are likely to prove useful in the future for extracting new data from the data set according to different research questions or the measurement of other, newly identified variables.

We documented our experiment design with an experiment protocol in which the experiment environment, process and measurement methods are outlined in detail. This documentation enabled consistency for running the experiment at different locations and will likewise facilitate future replication. For the diagrammatic representation of architectures we chose to use UML, which is the de facto standard for representing software designs.

6.4.1 Experiment Planning

The study was executed between February and August 2010. After an initial literature study, a general study design was created. We then initiated the human ethics approval process and started with the design of materials, questions and the experiment protocol. We ran several test sessions and subsequently refined the protocol, material and questions. The first 15 instances of the experiment were conducted in Wellington. While coding the first videos, we organized the second run in Leiden. Lastly, we focused mainly on the professional developers in the Dutch organizations. After 47 participant sessions (approx. 1.5 hour per session, including preparation), two weeks were spent on video coding and data entry (approximately three hours per participant).

6.4.2 Data Collection Process

We applied quota sampling (Wohlin et al., 2000) to obtain at least one professional developer for every three students who participated in the experiment. For both groups we applied convenience sampling. In Wellington, students participated voluntarily and went into a draw for a prize of NZ$50. In Leiden students participated both voluntarily and through a mandatory part of a course. Subjects from industrial organizations in Wellington participated voluntarily. All students but not all professionals had used UML during their studies. All participants had previous experience with UML. The study (which we referred to as "design study" towards the students) was mentioned during various software engineering lectures in Wellington. In Leiden, M.Sc. students who were enrolled in a research methodology course were required to participate. Student performance on the experiment did not influence their grade. In Leiden, absence from the experiment would be reflected in the course grade, though. Each

(a) Participant Station (b) Recording Environment

Figure 6.1: *Experiment environment*

participant was surveyed in the same way: the participant was welcomed and seated behind a table on which two blank pieces of paper were taped at 35 cm apart (see Figure 6.1(a)). A camera was placed so that it was able to accurately record movements of the participant's head. However, the environment was set up in a way so that the camera non-intrusively appeared as part of general audio and video equipment. This was done in order to prevent participants from getting overly nervous or self-conscious. The experimenter sat behind the participant so that the participant could concentrate on the task. This set-up also discouraged the participant from interacting with the experimenter. The set-up of the experimental environment was described in detail in the experiment protocol to minimize potential differences between the various locations at which the experiment took place. Generally, various participants were planned after one another. After a participant entered the room, he or she[1] was asked to sign a consent form and to fill out a preliminary questionnaire. This questionnaire consisted of 10 questions covering academic and industrial software modeling experience, a self-assessment of modeling skills, recent software architecture experience and demographics such as age, gender, (highest attained) level of education and native language. When designing the questionnaires, we followed standard guidelines (such as those described in Oppenheim, 1966). Next, it was explained that the objective of the study was to understand the use of software architecture design documentation and that a series of questions regarding such documentation would be asked. The participant was told that he could use the presented information in any way he deemed fit and that the experimenter did not know the correct answer to the question. In addition, the participant was requested to verbalize his thought process while using the architecture documentation. The first architecture design

[1]Most participants were male, we will continue referring to participants as males.

document presented was meant to serve as an example question only. The main purpose of this example question was to put the participant at ease and to acquaint him with the experiment protocol. The procedure of the experiment was carried out as follows: Architectural design documents would be placed on the participant's desk. Each architectural design was split up into two documents, one page contained a textual description and a another page contained a diagram. Both papers contained information about the same architecture. After placing the documents at their precisely defined positions, the context of the design would be briefly introduced. For instance, "This architecture describes the architecture for a booking system for flights." This introduction was made because in real software development scenarios, developers are aware of the domain or the high-level objectives of a system. Participants were then verbally asked questions about the design — such as *"Can component X directly authenticate users?."* Participants could then ask for a question to be repeated but no other requests such as clarifications would be accepted. After a participant had answered, the answer was repeated by the experimenter for verification purposes and the next question was asked.

Participants were given no feedback as to the correctness of their answer. The experiment was double-blind as neither participant nor experimenter was aware of the correct answer or in which of the documents the answer had to be found. As a result, the participant would not change his behavior according to his record of correct or incorrect answers during the experiment. Four architectures were used and three questions were asked per architecture — not counting the introductory example.

6.4.3 Material and Question design

Each set of architecture documentation consisted of a pair of sheets of which one contained text and one contained a diagram. After the last question, the participant was handed a second questionnaire, asking him to rate the extent to which he perceived the two media types used as being effective and the degree to which he understood all notational elements. All five text-diagram pairs (four pairs and one example pair) were inspired by industrial software architecture diagrams (SADs) documents in our possession. The documents were altered in such way as to:

- enable a uniform notation across cases,
- enable separation from larger design documents and
- enable translation to English where needed.

We focused on the ability of participants to extract design information from both grammatically and syntactically correct diagrams and texts. We corrected ambiguous constructs in original documentation and attempted to attain an overall coherent visual style plus lucid textual descriptions. All diagrams represented structural views of the system. We used UML 2 component and deployment diagrams. An example of one of

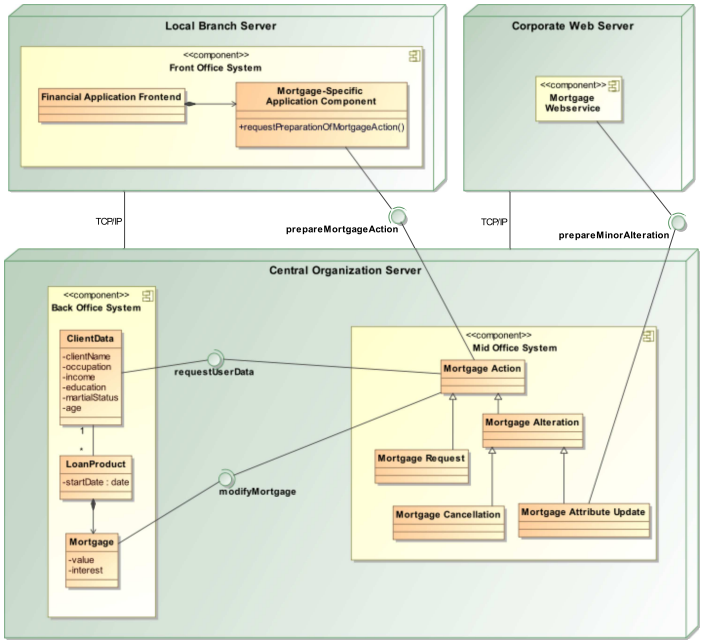
the (verbose) diagrams is depicted in Figure 6.2(a). The text consisted of a description of the architectural component in natural language text. An example can be found in Figure 6.2(b). This text was assembled to be in concordance with industrial SADs. The non-verbose version of the text contained fewer details regarding the model and was about half as long as the verbose version. Questions and design documents were carefully tuned to each other in order to allow a multitude of subsequent analyses. Examples of questions are: *"Is system x the only component that may modify attribute y?"* and *"Through what node does system x connect to system y?."* The questions can be found in Table 6.4.

We define a set T that holds all design information described in the text sheet and a set D that holds all design information described in the diagram sheet. All design information is then described by $T \cup D$. We designed the questions so that the answer to some questions can be found in the intersection $T \cap D$, some answers can only be found in the complement $T \setminus D$ and some answers can only be found in the complement $D \setminus T$. Finally, some questions are not to be found in either set ($\sim (T \cup D)$). The distribution of the answers for our questions is described in Table 6.4.

When designing questions, we limited ourselves to questions which could be answered with information available either in text or via diagrams. We designed the architecture representations and the questions so not to rely on detailed knowledge of UML semantics. The experiment design process involved determining and listing the most important design information conveyed in the design, creating a set of questions relating to this information and validating the questions by means of trials. Overly complicated or ambiguous questions were disregarded or rephrased. We created a total of nine architecture pairs before selecting the final four, which were selected based on ease of understanding. Half the question set consisted of open questions. Answering open questions is comparatively more difficult because a) more information has to be gathered and b) it is not as clear as to whether further consultation of another medium is required once a partial answer has been formulated. Open questions provide more insight into how easily a participant is satisfied with partial information.

With a view to our hypothesis $H2_1$, five questions were designed to address design information of a topological nature. Per architecture, we only asked three questions. We left more difficult questions for last as they required increased use of the media and might create a learning effect for subsequent easier questions.

Each architecture was described using one page of text and one page with diagrams. We created two versions of each architecture description. One version contained a verbose diagram and non-verbose (content reduced) text and the second version contained verbose text and a non-verbose (content reduced) diagram. We refer to the former as a diagram-dominant representation and the latter as a text-dominant representation. We created non-verbose versions of diagrams and text by removing elements or sentences from the complete ones. Each participant was presented two diagram-dominant representations and two text-dominant representations making sure that the two verbose diagrams were not placed on the same side of the desk each



(a) verbose diagram

The system described in this diagram provides support for creating new mortgages and alteration of existing mortgages.

The design aims to separate the complexities of the business logic from the Financial Application Frontend by bundling all mortgage-related services on a central Mid Office System. This system provides services for the setup of all `mortgage actions'.

The Front Office Component hosts a Financial Application Frontend which contains a Mortgage-specific Application Component. Due to concerns regarding decreased Back Office availability, mortgage action requests may have a maximum size of 300 kilobytes.

The Mortgage Webservice provides an additional method to update mortgage attributes. This service only connects to an interface provided by the Mortgage Attribute Update specialization.

(b) non-verbose text

Figure 6.2: *Example design*

Table 6.2: *Media Dominance*

architecture		α	β	γ	δ
medium		**t d**	**t d**	**t d**	**t d**
experiment version	*A*	V n	V n	n V	n V
	B	n V	n V	V n	V n

t = text / **d** = diagram
V = Verbose / **n** = non-verbose

time. Ergo, out of every four participants, every first participant received the ordering pair α (non-verbose diagram on his left side), pair β (non-verbose diagram, right), pair γ (verbose diagram, left), pair δ (verbose diagram, right). The questions were the same for diagram- and text-dominant representations of the architecture. The distribution of media dominance is summarized in Table 6.2.

6.4.4 Ordering Process

We eliminated a potential bias caused by a possible tendency of participants to start reading the information on a particular side (e.g. the left hand side for Western participants) by changing the position of the diagram for every pair presented. In order to prevent participant preferences for a media type, based on the verbosity of the medium, medium-verbosity was also balanced. Furthermore, the ordering of the architecture pairs used was changed for every participant so that results for any particular architecture pair (particularly the first and last pairs particularly pairs α and δ) would not be influenced by effects due to the participant learning, getting tired or getting bored.

6.4.5 Data Coding

We obtained information regarding the amount of time subjects looked at media by manual coding of the video recording. We counted switches between media, which media the participants looks at first and last. An excerpt of the manner in which the videos were coded is depicted in Table 6.3.

To ensure consistency of extraction of data from the video recordings, we used a set of guidelines for coding. For example, the first timing measurement started at the moment the question was spoken and timing stops when the participant mentions the core element of his answer. To ensure data-entry consistency, we used scripts to transfer timing information from spreadsheets to the database. We then performed consistency checks on the final data employed by means of a set of semi-formal consistency checks such as

$$\forall switches\,[\mathtt{mod}(switches) = 0 \Rightarrow medium_{start} = medium_{end}]$$

Table 6.3: *Example of video coding log for a single question*

question number	diagram or text (or answer)	video timing (mark[1])	relative timing (seconds)	looked at diagr. (seconds)	looked at text (seconds)
γ2	d	594.9			
	t	610.5	15.6	15.6	
	d	615.2	4.7		4.7
	answer	619.1	3.9	3.9	
answer given: "no"					

[1] from beginning of video file
Derived metrics:

- *total time looked at diagram:* 19.5 seconds (80.58% of total time)
- *total time looked at text:* 4.7 seconds (19.42% of total time)
- *switches between media:* 2
- *started at:* diagram
- *ended at:* diagram

which were executed by means of a set of SQL queries. In addition, we recalculated all derived values (e.g. percentages) in the database.

6.5 Results and Discussion

In this section, we discuss the results of the experiment. A total of 47 subjects participated of which 35 were male and 12 female. The average age was 27, ranging between 21 and 41. The participants came from two universities and four different organizations:

- 12 BSc. and BSc-hons. students, 1 MSc. student and 1 Ph.D. student at the School of Engineering and Computer Science (ECS) at Victoria University Wellington;
- 1 BSc. student, 20 MSc. students and 1 Ph.D. candidate at the Leiden Institute of Advanced Computer Science (LIACS) at Leiden University;
- 12 developers from the field of custom software development at various different organizations in New Zealand and the Netherlands, including Capgemini the Netherlands, Infoprofs and ASR Insurances.

Because of non-normal distributions, we use the non-parametric Mann-Whitney U test for comparison between groups and Kendall's τ for bi-variate correlation analysis. As expected, professionals are significantly older and reported significantly more academic ($U = 105$, $z = -2.9$, $p < 0.01$) and industrial experience ($U = 107$, $z = -2.6$, $p < 0.01$) than the students.

In the following sections we will evaluate the data obtained with respect to our hypotheses.

6.5.1 Media Effectiveness

We evaluated media effectiveness in terms of how well participants were able to extract information from the documents i.e. in terms of the amount of correct answers given. The distribution of the correctness of the answers to the 13 questions we posed is leptokurtic and left-skewed. A Shapiro-Wilk (S-W) normality test confirms that the distribution of correct answers (slightly) deviates from normality ($p = 0.04$). We therefore resort to non-parametric tests for statistical analysis.

24 Participants worked with version A of the experiment materials and 23 participants worked with version B. Media dominance was distributed equally over both versions of the experiment (see Table 6.2) to be able to examine the effect of media dominance. We used a Mann-Whitney U test to check if there were no significant differences between the measurements obtained for both versions of the experiment for all variables under study.

We tested for any potential advantage of diagram-dominant versus text-dominant document pairs. Descriptive statistics for the amount of given answers that were correct for text-dominant versus diagram-dominant architecture descriptions are depicted in Table 6.5. Histograms for the number of correct answers given for each treatment are depicted in Figures 6.3(a) and 6.3(b).

A visualization of the differences of the distribution of answers is depicted in Figure 6.4. Surprisingly, we found that neither diagram- nor text-dominance had a significant effect on correct answers given nor on the time spent on answering questions. We therefore reject $H1_1$. Diagrams were not more effective, despite a substantial number of participants whose first language was not English (77 percent of participants). We will further discuss the role of language in Subsection 6.5.7.

We could not observe significant amounts of initial media preference or average media preference either. These findings do not change if we look at a subset of the questions: For the four questions to which the answer could be found only in either medium, no medium type proved to be more effective in terms of causing more correct answers. The findings did not change either when we looked at professionals and students separately. This contradicted the first hypothesis: diagrams were not preferred over text.

The treatment showed no clear pattern. We therefore looked for patterns in the data. Analysis of the respondent behavior led to the identification of two groups: One group

Table 6.4: *Experiment Questions*

pair		question	type	nature	answer location (for experiment version)	
					A	**B**
ex		*Where is information x stored?*[1]	open	non-topological	T ∩ D	T ∩ D
α	1.	*Which external system is a source of information regarding x?*	open	topological	T ∩ D	T ∩ D
	2.	*What type of service is service x?*	open	non-topological	T ∖ D	D ∖ T
	3.	*How does system x search in system y?*	open	topological	T ∩ D	T ∩ D
β	1.	*Is input x accepted by system y?*	closed	non-topological	T ∖ D	D ∖ T
	2.	*Can system x directly provide functionality y?*	closed	topological	T ∩ D	T ∩ D
	3.	*Can message type x be ignored by system y?*	closed	non-topological	~ (T ∪ D)	~ (T ∪ D)
γ	1.	*Name one of the responsibilities of system x*	open	non-topological	D ∖ T	T ∖ D
	2.	*Is system x the only component that may modify attribute y?*	closed	topological	T ∩ D	T ∩ D
	3.	*Is there a limitation on functionality x when requested from system y?*	closed	non-topological	~ (T ∪ D)	~ (T ∪ D)
δ	1.	*What type of messages are sent from system x to system y?*	open	non-topological	D ∖ T	T ∖ D
	2.	*Through what node does system x connect to system y?*	open	topological	T ∩ D	T ∩ D
	3.	*Is the communication between system x and system y secure?*	closed	non-topological	~ (T ∪ D)	~ (T ∪ D)

[1] system and component names are anonymized to enable reuse of the experiment material

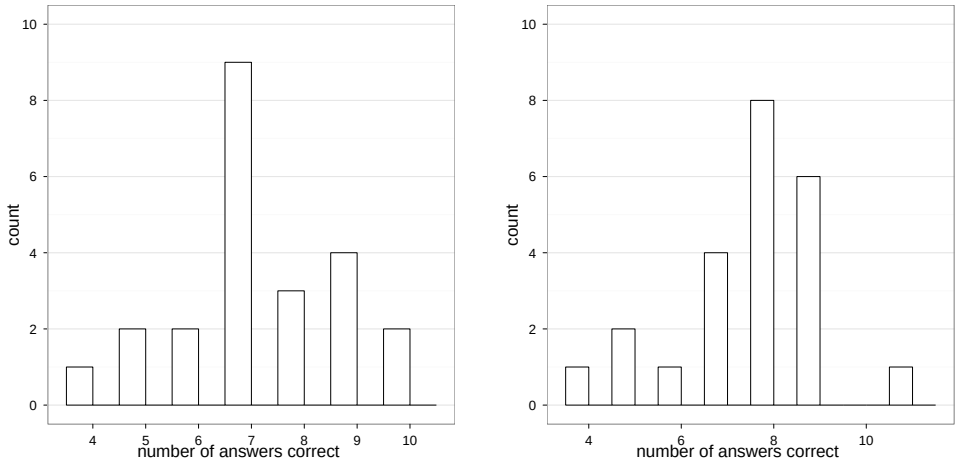
Table 6.5: *Descriptive statistics for the amount of given answers that were correct for text-dominant versus diagram-dominant architecture descriptions*

Treatment	**Answers Correct** (out of 12)		
	median $\tilde{x}$	mean $\bar{x}$	std. dev. σ
text–dominant[1]	8	7.70	1.54
diagram–dominant[2]	7	7.35	1.52

[1] version A of (α + β) + version B of (γ + δ)

[2] version B of (α + β) + version A of (γ + δ)

(for dominance distribution, also see Table 6.2)



(a) Diagram-dominant treatment (b) Text-dominant treatment

Figure 6.3: *Histograms for the amount of correct answers per treatment*

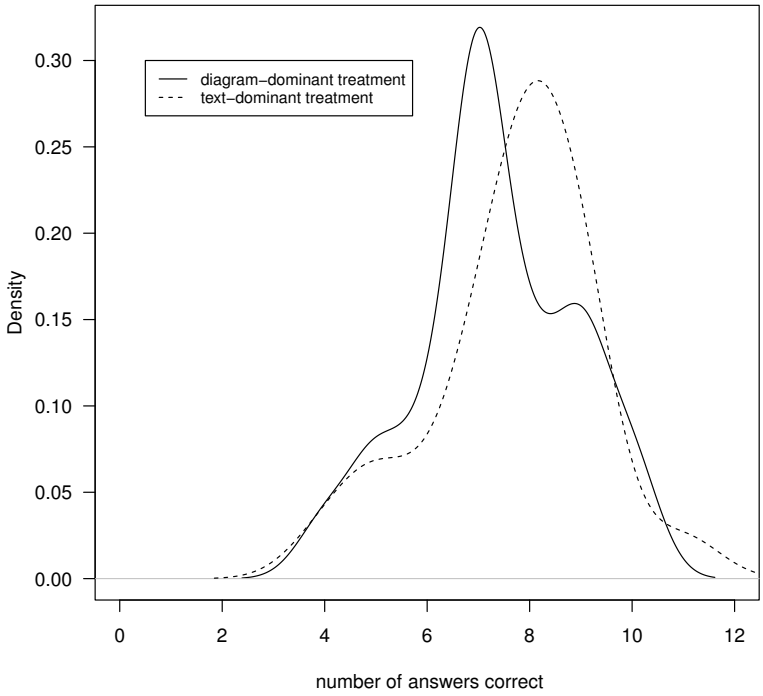


Figure 6.4: *Density plot for amount of correct answers per treatment*

(62 percent of the participants) predominantly used diagrams to answer questions, the other (the remaining 38 percent of the participants) used text more intensively. The group that uses diagrams more often, thinks that diagrams are more effective, answers faster and switches media more often and more frequently only resorted to diagrams to answer a question. The other group used text more often, was more experienced and scored better. Note that while the latter group suggests that participants who preferred text scored better, this does not imply a general advantage in effectiveness for text: participants who used the text-dominant architecture descriptions did not score better, as we mentioned earlier.

Using the post-experimental questionnaire, participants were asked to rate the effectiveness of both media on a 7-point Likert scale. Participants who prefer diagrams are significantly more likely to perceive the effectiveness of diagrams to be higher than participants who attribute a comparable score to their perception of the effectiveness of text. As mentioned, the diagram-preferring group of participants, who was faster ($\tau = -0.265, p \leq 0.05$) and rated their media type of preference as the most effective ($\tau = -0.265, p \leq 0.05$), did not score better in terms of correct answers, i.e. where not, in fact, more effective. In fact, we found that those participants who predominantly use text, score significantly better ($\tau = 0.281, p \leq 0.05$). So, participants who prefer text make a realistic judgment about the effectiveness of text. Participants who prefer

diagrams, on the other hand, significantly overrate the effectiveness of diagrams. As a result, the group that preferred diagrams not only scored lower, but thought they had used the more effective media type. Diagrams seem to offer a specific group of developers a false sense of confidence. This group was more willing to provide an answer based on inferencing from the diagram than to carefully examine the text. In addition, we found that those with more experience more often resort to text to answer a question ($\tau = 0.273, p \leq 0.05$).

In the following section we investigate whether questions relating to topological architecture properties are better catered for by either media type ($H2_1$).

6.5.2 Media Effectiveness for Topological Properties

In this section we explore whether diagrams are more effective at conveying topological design information to software developers. We define topological properties as the design information that is related to the static and structural dependencies between subsystems. Given that these topological properties lend themselves well to visualization, we expect that diagrams are the favored source for answering such questions. However, we could not observe that participants used diagrams more often to answer questions addressing properties of a topological nature (see Table 6.4). We therefore reject $H2_1$. Contrastingly, we found that in comparison to non-topological questions, participants significantly more often used the diagram when they provided their answer ($U = 6.5, z = -1.797, p \leq 0.05$).

Media dominance had no influence on participant behavior for most of the five topology-related questions, with the exception of two questions: We found significant differences in the amount of correct answers given for questions $\alpha 3$ ($U = 191.5, z = -2.64, p < 0.01$) and $\delta 2$ ($U = 230, z = -2.03, p < 0.05$) (of the type *"How does X search in Y"* and *"Through what node does X connect to the Y?"* respectively). The answer to either question could be found in the diagram and the text in both document versions.

6.5.3 Media Preference

Media preference denotes the characteristic of an inherent preference of a person to prefer to work with either textual or graphical representations. Analyzing the data from this perspective, we found that participants prefer the diagram as their first source of information: in 10 out of 13 questions, participants first started examining the diagram. This count includes participants who only briefly gaze over the diagram before examining the text. Media dominance does not significantly influence this behavior. A possible explanation is that diagrams are used to get an initial global overview of a system. This is in line with the "Visual Information-Seeking Mantra" that Shneiderman (1996) describes. In his work, Shneiderman outlines the steps followed in visual information retrieval. He summarizes his principle in a mantra: *"Overview first, zoom and filter, then details-on-demand."*

When aggregating the usage pattern of media for all questions, 98 percent of all participants used both media types to arrive at answers. Only one participant relied exclusively on diagrams for answering all questions. When analyzing answers to individual questions, 27 percent of the 611 answers given (47 participants × 13 questions), were based on the use of only one medium. In contrast, five participants (10 percent) always used both media to answer a question. The latter group might be classified as a "thorough" group, not only because of their comprehensive media usage but because this group scored higher ($\tau = -0.312, p \leq 0.01$). The latter fact is not a result of the group consisting mainly of experts. Participants of this group were not more experienced nor was it composed mainly of professional developers. Also, these participants had slower response times ($\tau = -0.464, p \leq 0.001$) further corroborating the notion of "thoroughness." Note that "non-thorough" participants had no particular reason to work quickly as no time limit was imposed.

Interestingly, we found that "thorough" participants were more likely to start and end with looking at a diagram. In contrast, another group that scored better than average, predominantly used text. This latter group was composed of experienced participants. So, "thoroughness" does not correlate with media preference but "experience" does. A potential explanation for that latter observation is that industrial practice could have made experienced developers diagram-averse in the sense that they prefer understanding all text accompanying a diagram. Perhaps the low quality of diagrams they had do deal with in the past created low expectations as to the utility of diagrams in general. By using an eye tracker, Yusuf et al. (2007) also found that more experienced developers read diagrams differently.

6.5.4 Guesses and Suppositions

For four questions, the answer could be found in either the text or in the diagram, depending on the version of the experiment a participant obtained. For each of these four questions ($\alpha 2, \beta 1, \gamma 1$ and $\delta 1$), we looked at the amount of participants that used *only* the medium that did not provide enough information to answer the question. These participants did not switch to the medium from which the answer could be derived. Note that nothing inhibited these participants from spending as much time as they needed to answer the questions. These participants therefore seemed to prefer guessing over continued consideration of the presented material. For all participants, 10 out of a 188 (4 questions × 47 participants) or 5.3 percent of all questions, were given based on a single medium from which the answer could not be derived. Only in one of these 10 cases did a participant guess the correct answer. This was for question β 1 which was the only closed question of this type. This participant therefore had a "fifty-fifty" chance of guessing the right answer. These 10 answers were given by 9 different participants. So, 19 percent of participants demonstrably and needlessly guessed the answer to at least one question. Two of these participants were experienced software developers who are active in industry. Note that "switching between media"

in the context of this experiment refers to the act of slightly moving one's head from left to right or vice versa. As architecture representations are often vast and dispersed over various sources, in practice (much) more effort is likely to be needed to understand a given aspect of a software architecture.

To be able to find a measure of the extent to which developers are satisfied with incorrect information, we inserted two "false friends" into the experiment design. These "false friends" constitute information that resembles a correct answer but would be easy to distinguish as incorrect information, if given sufficient attention would be given to detail. In the experiment design, the answers to three questions could not be derived from either medium (for both versions of the experiment). In two of these cases, we inserted these "false friends." For question β 3 (*"Can message type x be ignored by system y?"*), one of the classes contained the method "`+ignoreMessage()`." That particular class had no relation to external systems. For question γ 3 (*"Is there a limitation on functionality x when requested from system y?"*), the text contained the phrase "requests may have a maximum size of 300 kilobytes." Request volume and constraints on individual requests are unrelated, most participants correctly observed. For these questions, we took into consideration why a participant gave a specific answer (by means of the think aloud protocol) to be able to tell whether a "false friend" was the cause for the specific answer given. In univariate analysis, we found that those with more industrial experience, are less likely to fall for a false friend ($\tau = -0.333, p \leq 0.01$). These participants are less quickly satisfied with information that only resembles a correct answer. We found that the participants who demonstrably and needlessly guessed the answer to at least one question, were not more likely to fall for a false friend (Mann-Whitney $p = 0.88$).

6.5.5 A Case Against Overlap

While it is commonly accepted that images are far better remembered than text (e.g. McDaniel and Pressley, 1987), there are various restrictions to their use. For example, the third of the "ten commandments of picture facilitation", of Levin et al. (1987) (which were more recently validated by Carney and Levin (2002)) reads:

> *"Pictures shalt not be used in the presence of "heavenly" bodies of prose. If the text is highly memorable to begin with, there is no need to add pictures."*

For software design representations, this would imply that a diagram should only be used when a textual description is insufficient. Moreover, if a textual description is very clear (*"heavenly"*), diagrams should be avoided to prevent confusion. We reported that we found that neither text nor diagram-dominant descriptions are more efficient in communicating software architecture design. However, for five questions, we represented similar information in both media ($T \cap D$ - also see Table 6.4). Indeed, we found in industrial reality often an overlap of the information presented in diagrammatic and textual models (Heijstek and Chaudron, 2011). Given Levin's commandment, due

to confusion, participants could have scored lower for questions to which the answer could be derived from both media (the example question and questions $\alpha 1$, $\alpha 3$, $\beta 2$, $\gamma 2$ and $\delta 2$), compared to the other questions. The score per question varies too much to be able to compare whether this was the case. When we consider whether participants who predominantly used the text or the diagram, we found that this is not related to whether they answered these questions correctly.

6.5.6 Conflicting Information

In industrial reality, developers are confronted with design documentation that is mostly incomplete and often inconsistent. Both incompleteness and inconsistency in UML diagrams have been addressed by empirical research. For example, Lange and Chaudron (2006) studied the effect of defects in UML models on developer comprehension. In their controlled experiments with a group of 159 students and industrial practitioners, they found that defects in UML models often remain undetected and cause misinterpretations. In addition, they found no implicit consensus about the interpretation of undetected defects and conclude that defects in UML models are potential risks that can cause misinterpretation and miscommunication. Another interesting finding of this study is that the presence of domain knowledge strongly decreased the detection rate for a defect type. Domain knowledge might therefore lead people to more quickly fill in omissions based on assumed domain knowledge. Nugroho (2009) found that a higher level of detail in a UML model significantly improves correctness and efficiency of subjects in comprehending UML models. He also found that models with a lower level of detail were more often misunderstood or misinterpreted. Balzer (1991) reported that harsh consistency constraints on design in practice are often removed in favor of flexibility. The architecture used to support the first question, the example question, contained conflicting information between the diagrammatic and textual representations. By analyzing the answer and participant behavior, we could determine which medium type was more dominant for participants. Out of all participants, 85 percent examined both media. The other 15 percent used only the diagram (all of these participants on average preferred diagrams for all subsequent questions). Out of those who used both media for the first question, we found that only 35 percent preferred the answer that could be deduced from the text. We should note that the example architecture was text-dominant for all participants. Many participants noted that they found the inconsistency and deliberately choose the diagram. When confronted with conflicting information, developers seem to decide that the information presented in a diagram is more authoritative than the textual information.

6.5.7 Participant Characteristics as Performance Predictors

Gemino and Wand (2003) recommend examining three antecedents of knowledge construction in empirical evaluation of model representations, based on Oei et al.

(1992): (a) content, (b) presentation and (c) model viewer characteristics. Now that we have established that no media type was more effective in communicating architecture information, we set out to investigate: To what extent can developer (or: model viewer) characteristics explain the variance in the amount of correct answers given? In this section we investigate which participant characteristics can be used as predictors of performance. We employed a multiple regression analysis using the backward elimination method (Hocking, 1976). We considered the following variables:

1. *Experience* (Explained in Section 6.5.7)
2. *Media Preference* (Coded as diagram = 0, text = 1; Explained in Table 6.1)
3. *Media Exclusion* (i.e. how often a participant only used one medium)
4. *Diagram Working History* (i.e. how long it has been since the participant last worked with software design models)
5. *Media Inclusion* (i.e. average switches between media) (Also explained in Table 6.1)
6. *Self-Rated Modeling Skill* (Explained in Table 6.1)
7. *Average Time per Question* (Explained in Table 6.1)
8. *Linguistic Distance* (Explained in Section 6.5.7)

The first and last participant characteristics are explained next in more detail.

Linguistic Distance

For linguistic distance we adopted a measure reported by Chiswick and Miller (2004) based on a study by Hart-Gonzalez and Lindemann (1993). Chiswick and Miller pose the question: *How difficult is it for individuals who know language A to learn languages* B_1 *through* B_i, *where there are* i *other languages?* They go on to state that *"if it is more difficult to learn language* B_1, *than it is to learn language* B_2, *it can be said that language* B_1 *is more "distant" from A than language* B_2." A list of the languages encountered in the experiment and their associated linguistic distances to English can be found in Table 6.6.

We found that univariately, this measure significantly correlates with the amount of correct answers given ($\tau = -0.289$, $P \leq 0.05$). This implies that the further a participant's native language is removed from English, the fewer correct answers are given. Language distance, therefore, is important and should be minimized. This implies that diagrams were unable to bridge language barriers. In addition we found that participants whose language has a certain minimum distance away from English were significantly more likely to switch between media.

Table 6.6: *Language Grouping & Distance*

language[1]	*n*	score[2]	family	distance[3]
English	13	-	Indo-European	0
Romanian	1	3.00	Indo-European	0.33
Dutch	20	2.75	Indo-European	0.36
German	1	2.25	Indo-European	0.44
Spanish	1	2.25	Indo-European	0.44
Farsi	1	2.00	Indo-European	0.50
Bulgarian	1	2.00	Indo-European	0.50
Tagalog	1	2.00	Austroeasian	0.50
Bengali	1	1.75	Indo-European	0.57
Mandarin	4	1.50	Sino-Tibetan	0.67
Arabic	1	1.50	Afroasiatic	0.67
Chaouia	1	-	Afroasiatic	-
Nyanja	1	-	Niger-Congo	-

[1] self-reported by participant ("Native Language")

[2] reported in (Hart-Gonzalez and Lindemann, 1993)

[3] inverse of score ($\frac{1}{\text{score}}$) (Chiswick and Miller, 2004)

Participant Experience

In this paragraph, we explain the metrics we used to quantify participant experience. We used the proposed ordinal classes of participant experience by Höst et al. (2005):

- **E1:** undergraduate student with less than 3 months recent industrial experience
- **E2:** graduate student with less than 3 months recent industrial experience
- **E3:** academic with less than 3 months recent industrial experience
- **E4:** any person with recent industrial experience, between 3 months and 2 years

This metric is a compound, composed of the values we collected for academic and industrial experience (see Table 6.1). A comparison of the distribution of these three variables is depicted in Figure 6.5. In this figure, one can observe that the metric proposed by Höst et al. (the middle line) is a proper compound of academic and industrial experience for our participant.

The values for experience we obtained are not normally distributed (leptokurtic, right skewed, S-W $p \leq 0.001$). We use Kendall's τ (Noether, 1981) for bivariate statistical analysis. We found that the Höst et al. index of experience individually correlates with the amount of correct answers a participant gave ($\tau = 0.31, p = 0.1$). Experience,

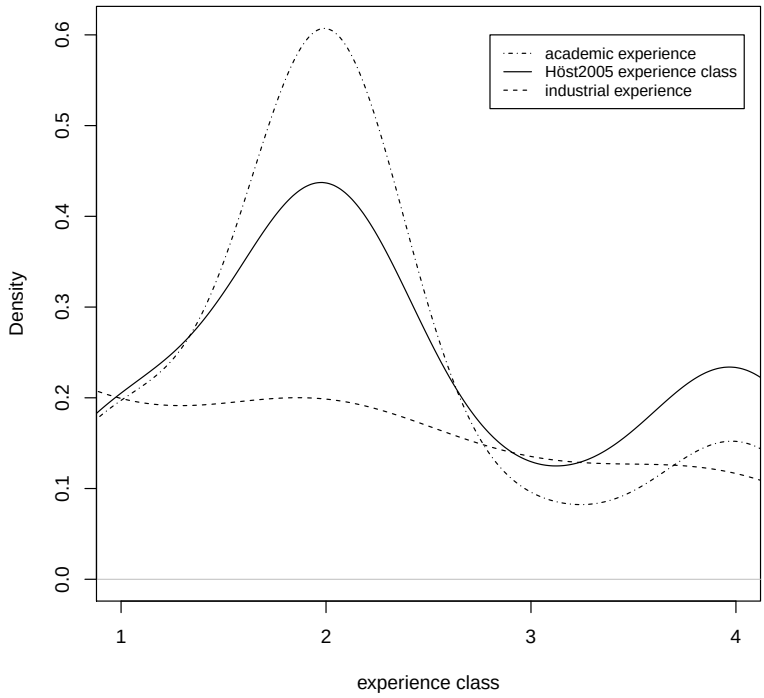


Figure 6.5: *Density plot for academic experience, industrial experience and Höst experience class of participants*

however, does not relate to the speed with which a participant was able to answer or the amount of times he switched between media.

Multivariate Analysis Results

The results of our multivariate analysis are summarized in Table 6.7. The multivariate analysis results in a model that features four significant variables, namely media preference, linguistic distance, experience and self-rated modeling skill. The model is highly significant ($p \leq 0.001$) and explains 50 percent of the variability of correct answers given. We were able to rule out multicolliniarity among the resulting predictors; the variance inflation factors (VIF) are not substantially higher than 1 (Bowerman and Richard, 1990). No individual subject seemed to influence the model as e.g. no case has a standardized residual larger than 2. We found that the coefficient for language distance is negative, implying that the further a participant's language is from English, the fewer correct answers the participant will give. Remember that media preference is coded as diagram = 0, text = 1. Therefore, this confirms the earlier result that a textual media preference was beneficial for correctly answering the questions. As expected, experience was also beneficial for providing correct answers.

Table 6.7: *Multivariate regression for "Amount of Correct Answers"*

Model*	**Unstdized. Coeff.**		**Std. Coeff.**	**t**	**Sig.**	**Correlations**			**Collinearity**	
	B	Std. Error	β			Zero-order	Partial	Part	Tolerance	VIF
(Constant)	5.959	0.760		7.842	0					
language distance	−3.288	0.927	−0.411	−3.548	0.001	−0.370	−0.499	−0.406	0.972	1.028
media preference	1.236	0.418	0.343	2.960	0.005	−0.362	0.443	0.338	0.971	1.029
modeling skill	0.442	0.150	0.342	2.949	0.005	0.294	0.432	0.337	0.974	1.027
experience	0.499	0.188	0.309	2.653	0.012	0.409	0.395	0.303	0.965	1.036

* Model Summary: $R^2 = 0.503; p \leq 0.001$

The fact that self-reported assessment of modeling skill turned out to be a predictor is consistent with similar findings that show that self-reported ability correlates with e.g. actual mechanical and spatial ability (Hegarty and Just, 1993). In order to be able to see which predictor was the strongest, we standardized coefficients (which are measured in standard deviation units). We found that language distance had the greatest contribution to the model that predicts the performance of participants. Those participants whose native language was further away from English had greater difficulty understanding the software architecture designs.

6.6 Threats to validity

In the following, we discuss a number of factors that may have influenced the results obtained from the experiment in a way that prevents them from being indicative for other contexts as well. We categorized our potential threats to validity based on Wohlin et al. (2000).

6.6.1 Internal Validity

We addressed problems relating to maturation i.e. increasing familiarity with a problem, in several ways: We interacted with participants by means of a strict interaction protocol so to be able to react to questions or remarks in a uniform way. We thus prevented participants to use the experimenter as an additional source of information. We also changed the order in which participants received the architectural descriptions (which also prevented effects due to fatigue).

We minimized the threat of "diffusion of treatment" by asking participants not to discuss the experiment with their colleagues or fellow students. No materials could be taken by participants from the experiment environment.

Threats related to instrumentation were addressed by using a protocol for the use of the camera. On the different locations where we conducted the experiment, we "camouflaged" the camera by placing it inconspicuously among other equipment in its surroundings (as in Figure 6.1(b)). Whenever the camera needed to be handled, it was done while no participant was in the room.

A related potential threat is that participants behave differently because they are aware of the experiment situation. Either stress or the "Hawthorne effect" could introduce a negative or positive bias respectively. In order to address such factors, we introduced an example question to allow participants to get used to the environment. We believe that participants found the questions challenging enough to fully concentrate on the design documentation, i.e. practically became oblivious to the experiment situation. This is supported by our analysis of the video material. The experiment protocol prescribed interaction while documentation was switched in order to keep

participants occupied. We equivocated the use of a camera and by only mentioning the session would be "recorded" without providing further detail how this would be done.

Another possible threat is that various studies of the process of solving problems reported that verbalization of this process improves problem-solving performance (e.g. Johnson and Shih-Ping, 1999, Flaherty, 1975). Therefore, this might have influenced the extent to which participants were able to provide a correct answer. In addition, the intensity and contents of the verbalization varied strongly among participants. Not all participants will have benefited from the performance increases that verbalization yields. Measures taken to diminish the described threats to validity are (1) not facing the participant, (2) not answering participant questions, (3) making use of an interaction protocol to limit the amount of possible responses that can be given to a participant. The interaction protocol contains a process for stimulating thinking out loud.

The phrasing of the questions might have been suggestive in terms of which medium is likely to provide the answer. This might have introduced a bias for media preference and effectiveness, which may or may not become directly apparent. Our findings do not suggest any such bias at all but due to the complexity of the subject we cannot rule it out for every question.

Finally, one might argue that the separation of diagrams and text into two pieces of paper might have invited participants to always use both. We had to create a sufficient amount of physical separation between the sheets in order to be able to distinguish their use in the video material. However, it is not clear as to whether less physical separation would have made a difference and certainly some participants almost exclusively used diagrams, demonstrating the potential to ignore one of the documents if that was felt to be appropriate.

6.6.2 External Validity

The extent to which graduate, undergraduate and doctoral students are representative of professional software developers and the threat of interaction of selection and treatment respectively, have been addressed in various other studies (e.g. Arisholm and Sjøberg, 2004, Briand et al., 2005, Höst et al., 2000). The students who participated in this experiment were all exposed to software modeling in general and the notation used for the diagrams (UML) in particular during various courses at the bachelor or master level. In addition, more than a quarter of participants were actual professional software developers (more than one for every three students). We maintain that for understanding the software architecture designs we used, expert knowledge of the UML is not a prerequisite for extracting information from the diagrams needed for answering our questions.

In other words, we do not know whether these participants would not have done equally well, if they had used the diagrams more often. If experience with low-quality diagrams really played a role in our experiment then this particular finding would imply "media adversity of good participants" rather than on "media effectiveness for

text-centric developers." Note, however, that overall we detected no better effectiveness for text, so the potential bias did not affect our overall results.

The findings were based on fragments of architectural descriptions that were closely based on samples of such descriptions taken from industrial SADs. Hence, these are considered to be representative for the class of project that use a mix of text and diagrams.

The threat of apprehension was mitigated by assuring anonymity and by explaining the recordings could only be accessed by the involved researchers (whose names were mentioned) and would be destroyed after data collection. Also, no time limit was imposed upon participants. We avoided hypothesis guessing by keeping the participants uninformed about the study objective.

6.6.3 Conclusion Validity

Reliability of measures was ensured by means of testing the used cameras and determining proper positioning to be able to clearly obtain head and eye gaze movement. This information was documented in the experiment protocol. A coding protocol was established for coding the videos and a second researcher re-coded videos to randomly assess accurateness. Database entry was semi-automated and subject to rigorous consistency checks. Threats regarding random irrelevancies were addressed by choosing quiet locations to conduct experiments. The experiment materials and environment were tested before real sessions commenced. The validity of the applied statistical tests (e.g. the assumptions and correct application), subject selection and the data collection process were discussed in previous sections.

6.7 Recommendations

This section contains an overview of the recommendation that follow from the findings of this study.

- The use of both text and diagrams is needed. The use of only one medium is not recommended, not even when the nature of the design decision to be communicated is topological. Also for topological information, for which diagrams intuitively seems to be the preferred communication medium, a textual description of the topology should be added.
- Another important reason for not defaulting to the use of only diagrams are that diagrams do not seem to bridge linguistic distance. In addition, they also potentially induce a false sense of confidence in developers in the sense that they make assumptions.
- When using diagrams, make clear how these should be read. In an earlier study, Holsanova et al. (2009) found that an "integrated format with spatial contiguity

between text and illustrations facilitates integration." Employing this method would imply annotating a diagram with a description of how to read it, either in natural text or using a more formal notation such as OCL.

- Consider your audience when engaged in GSD: Make sure that documentation is unambiguous.
- Do not use ambiguous constructs. Use similar terminology in both diagrams and text.
- Developers who are better at modeling, read diagrams better. UML training or training regarding conceptual modeling might be beneficial for developer understanding of architecture representation.

6.8 Conclusions and Future Work

Architectural design documentation is essential for communicating an architect's intentions. In current practice such documentation consists of a mix of diagrams and textual descriptions but their creation is not informed by solid knowledge about how the documentation is perceived by developers. We therefore conducted a controlled experiment in order to evaluate the merits of different mixes of diagrammatic and textual descriptions in which we tried to approximate industrial reality. One of the results is that neither diagrams nor textual descriptions proved to be significantly more efficient in terms of communicating software architecture design decisions. Another unexpected result is that diagrams are not more suited to convey design decisions of a topological nature. Remarkably, participants who predominantly used text, scored significantly better; not just overall but with respect to topology-related questions as well. Also, diagrams were not able to alleviate the difficulties participants with a native language other than English had in extracting information from the documentation. In combination, these findings question the role of diagrams in software architecture documentation.

However, while diagrams were not superior regarding media effectiveness they still seem to perform a special role. Participants were more likely to use diagrams as their first source. They were more likely to look at the diagram at the very moment when they provided answers to questions of a topological nature. Interestingly, thorough developers tend to start and end with diagrams. More research is required to fully understand how text and diagrams could complement each other, in particular with respect to topological system properties.

Our analysis discovered two emerging group characterizations. One group predominantly utilized diagrams, was faster and overrated the effectiveness of diagrams; the other group was more experienced and preferred text. Further analysis needs to be

performed in order to understand these groups, so to be able to specifically cater for them in the creation of software architecture documentation.

Finally, by conducting a multivariate regression analysis we identified developer characteristics that can be used as developer performance predictors: linguistic distance, media preference, experience and self-rated modeling skill. The participants who performed best had a native language close to English, used more text than diagrams, were more experienced and rated their modeling skill to be relatively high.

Summarizing, while our experiment and subsequent analyses produced some very interesting concrete findings, we feel that their ultimate value lies in the impetus they provide to perform further research to better understand the effectiveness and roles of media types in software architecture descriptions.