

Experimentally Investigating the Effectiveness and Effort of Modeling Conventions for the UML

Christian F.J. Lange¹, Bart DuBois², Michel R.V. Chaudron¹, and Serge Demeyer²

¹ Department of Mathematics and Computer Science, Technische Universiteit Eindhoven, P.O. Box 513, 5600 MB Eindhoven, The Netherlands,

`C.F.J.Lange@tue.nl`, `M.R.V.Chaudron@tue.nl`

² Lab On REngineering (LORE), University of Antwerp, Belgium,
`Bart.Dubois@ua.ac.be`, `Serge.Demeyer@ua.ac.be`

Abstract. Modelers tend to exploit the various degrees of freedom provided by the UML. The lack of uniformity and the large amount of defects contained in UML models result in miscommunication between different readers. To prevent for these problems we propose modeling conventions, analogue to coding conventions for programming. This work reports on a controlled experiment to explore the effect of modeling conventions on defect density and modeling effort. 106 masters' students participated over a six-weeks-period. Our results indicate that decreased defect density is attainable at the cost of increased effort when using modeling conventions, and moreover, that this trade-off is stressed if tool-support is provided. Additionally we report observations on the subjects' adherence to and attitude towards modeling conventions. Our observations indicate that efficient integration of convention support in the modeling process, e.g. through training and seamless tool integration, forms a promising direction towards preventing defects.

1 Introduction

The Unified Modeling Language (UML [35]) is the de facto software modeling language. Software development will become even more model-centric with the advent of Model Driven Architecture (MDA [34]). The UML is used in different phases during software development such as requirements analysis, architecture, detailed design and maintenance. In these phases it serves various purposes such as communication between project stakeholders, documentation of design decisions, prediction of quality properties and test case generation. The UML is designed as a visual multi-purpose language to serve all these needs. It allows to choose from 13 diagram types, it offers powerful extension mechanisms and it lacks a formal semantics. Due to these characteristics the user has the freedom to choose the language features that fit his purpose of modeling. However, the UML does not provide guidelines on how to use the language features for a specific purpose. For example, there is no guidance that describes when it is useful to use multiplicities or when a class should be described by a state diagram. As a result, the UML user is confronted with a large degree of freedom.

The UML possesses the risk for quality problems due to its multi-diagram nature, its lack of a formal semantics and the large degree of freedom in using it. The large degree of freedom and the lack of guidelines results in the fact that the UML is used in several different ways with respect to rigor, level of detail, style of modeling and amount of defects. Industrial case studies [25] and surveys give empirical evidence that individuals use the UML in many different ways (even within the same project team) and that the number of defects is large in practice. Moreover, experiments have shown that defects in UML models are often not detected and cause misinterpretations by the reader [24].

The effort for quality assurance is typically distinguished between *prevention* effort and *appraisal* effort [39]. Prevention effort aims at preventing for deviations from quality norms and appraisal effort is associated with evaluating an artifact to identify and correct deviations from quality norms. There are techniques in software development to detect and correct the aforementioned deviations from quality norms. Reviews, inspections and automated detection techniques are used in practice to detect weak spots. They are associated with appraisal effort. In programming preventive techniques to assure a uniform style and comprehensibility of the source code are established as coding conventions or coding standards [36]. As an analogy for UML modeling we propose *modeling conventions* to prevent modellers to deviate from quality norms. We define modeling conventions as **Conventions to ensure a uniform manner of modeling and to prevent for defects.**

The main purpose of this paper is to explore experimentally the effectiveness of modeling conventions for UML models with respect to prevention of defects.

An additional purpose of this study is to explore subjects' attitude towards modeling conventions and how modeling conventions are used. The observations can be used to improve the future use of modeling conventions.

This paper is structured as follows: Section 2 describes modeling conventions and related work. Section 3 describes the design of the experiment. Section 4 presents and discusses the results. Section 5 discusses the threats to the validity of the experiment and Section 6 discusses the conclusions and gives directions for future work.

2 Modeling Conventions

2.1 Related Work

One of the first guidelines for programming is Dijkstra's 'Go to considered harmful' [11]. The concept of coding conventions is nowadays well-established. There is a large variety of coding conventions (also known as guidelines, rules, standards, style) for almost all programming languages. The amount of research addressing coding conventions is rather limited though. Omam and Cook [36] present a taxonomy for coding conventions which is based on an extensive review of existing coding conventions. They identify four main categories of coding conventions: general programming practice, typographic style, control structure

style and information style. They found that there are several conflicting coding conventions and that there is only little work on theoretical or empirical validation of coding conventions such as [30]. Bieman [5] investigates the adherence to ‘style guidelines’, i.e. conventions, of source code and other software artefacts on real-world software systems. Initial results reveal a large amount of violations.

Our review of literature related to modeling conventions for the UML revealed the following categories: design conventions, syntax conventions, diagram conventions and application-domain specific conventions.

Design conventions address the design of the software system in general, i.e. they are not specific for UML. Riel [38] and Coad and Yourdon [8][9] provide conventions for the design of object-oriented system aiming at the maintainability of the system. The conventions that include for example high cohesion and low coupling are empirically validated by Briand et al. [6]. The results of their experiment show that the conventions have a beneficial effect on the maintainability of object-oriented systems.

Syntax conventions deal with the correct use of the language. Similar to Strunk and White’s book *The Elements of Style* for the English language Ambler [3] presents a collection of 308 conventions for the style of UML. His conventions aim at understandability and consistency and address syntactical issues, naming issues, layout issues and the simplicity of design. Ambler claims that the conventions are based on ‘real-world experience and proven software engineering experience’, but to the best of our knowledge there is no empirical validation of the conventions in literature.

Object-oriented reading techniques (OORT) are used in inspections [15] to detect defects in software artefacts. OORT’s for UML are related to modeling conventions in the sense that the rules they prescribe for UML models can be used in a forward-oriented way during the development of UML models to prevent for defects. The rules in OORT’s are used to detect consistency and completeness defects. Conradi et al. [10] conducted an industrial experiment where OORT’s were applied for defect detection (i.e. an appraisal effort). The results show defect detection rates between 68% and 98% in UML models.

Diagram conventions deal with issues related to the visual representation of UML models in diagrams. Diagram conventions were addressed in the Fourth Workshop on Graphical Documentation [33]. Koning et al. [20] provide a collection of diagram conventions that are not specific for UML models but apply to IT-architecture diagrams in general. Their conventions aim at improving the readability of diagrams and they provide a lightweight validation where industrial architects acknowledged the usefulness of 97% of their presented conventions. MacKinnon et al. [31] present a collaborative process to improve the readability of UML diagrams for technical documentation. A validation of the process is left as future work. Purchase et al. [37] present layout guidelines for UML class diagrams and collaboration diagrams based on experiments. Eichelberger [13] proposes 14 layout conventions for class diagrams aiming at algorithms for automatic layout of class diagrams. Sun et al. [40] present diagram

conventions based on perceptual theory and analyze the adherence to their conventions of layout-algorithms of two UML case tools.

Application-domain specific conventions. A collection of application-domain specific conventions is given by Loján-Mora et al. [29]. They describe conventions to use UML package diagrams to create multidimensional models for data warehouses. Additionally to their conventions they use an UML profile. UML profiles are sets of stereotypes, tagged values and constraints that are used to assign custom semantics to UML model elements. The purpose of profiles is to support modeling in a particular application domain. Hence, profiles are in fact application-domain specific conventions. Kuzniarz et al. [21] conducted an experiment on the effect of using stereotypes to improve the understandability of UML models. Their results show that stereotypes improve the correctness of understanding UML class diagrams by 25%.

2.2 Model Quality

Modeling conventions are proposed to improve the quality of UML models during development. Hence, in this experiment we investigate the effectiveness of modeling conventions on model quality. In this experiment we investigate

- Syntactic quality, which is the degree to which the model adheres to the modeling language.

Syntactic quality is one of the three notions of model quality according to Lindland's [28] framework for conceptual models. The two other notions according to Lindland are:

- Semantic quality: The degree to which the model correctly represents the problem domain.
- Pragmatic quality. The degree to which the model is correctly understood by its audience.

Evaluation of semantic and pragmatic quality involves participation of several people, and, hence, is an experiment itself. This would be out of the scope of this experiment, but we will investigate the effect of modeling conventions on semantic and pragmatic quality in a follow-up experiment.

2.3 Modeling Conventions in this Experiment

Based on the literature review and the experience from our case studies, we selected a set of modeling conventions. We assume that subjects with limited experience and time for training like the students in this study cannot handle a very large set of modeling conventions. To keep the set of modeling conventions manageable and comprehensible we decided that it should fit on one A4 page. This led to a set of 23 modeling conventions after applying the following selection criteria:

- Relevance. The modeling convention should be relevant to improve the quality of the UML model by preventing for frequent defects [26] [23]
- Comprehensibility. The modeling convention should be easy to comprehend (e.g. it relates to well known model elements).
- Measurability. The effect of the modeling convention should be measurable.
- Didactic value. Applying the modeling convention should improve the subjects’ UML modeling skills.

The collection of modeling conventions used in this experiment is presented in Appendix A. In this experiment we focus on assessing syntactic quality, but we deliberately don’t limit the collection of modeling conventions to syntactic conventions only. As described by Omam and Cook [36] there happens interaction between several conventions. Therefore it is necessary to use a representative set of modeling conventions, i.e. a set representing several categories, to obtain realistic results. Moreover we assess the semantic and pragmatic quality of the delivered models in a follow-up experiment such that we eventually assess quality in general.

3 Experiment Design

In this section we describe the experimental design according to Wohlin et al. [41].

3.1 Purpose and Hypotheses

In the previous section we have described the concept of modeling conventions. The purpose of the experiment reported in this paper is to investigate the effectiveness of modeling conventions in improving the quality of UML models. The goal of this experiment is summarized in Table 1 according to the GQM template by Basili et al. [4].

Modeling conventions require model developers to adhere to specific rules. Therefore we expect the quality of models to be better, i.e. there are fewer defects in a model that is created using modeling conventions. When additionally using a tool to check for adherence to the modeling conventions, we expect the model quality to be even better than without tool usage. In other words, we formulate in the null hypothesis that there is no difference between the treatments:

- $H1_0$: There is no difference between the syntactic quality of UML models that are created without modeling conventions, with modeling conventions and with tool-supported modeling conventions.
- $H1_{alt}$: There is a difference between the syntactic quality of UML models that are created without modeling conventions, with modeling conventions and with tool-supported modeling conventions.

Adherence to modeling conventions requires special diligence. We expect that this leads to higher effort for modeling. When additionally using the tool, the expected effort is even higher. Therefore we formulate the second hypothesis of this experiment as follows:

- H_{2_0} : There is no difference between the effort for modeling UML models that are created without modeling conventions, with modeling conventions and with tool-supported modeling conventions.
- $H_{2_{alt}}$: There is a difference between the effort for modeling UML models that are created without modeling conventions, with modeling conventions and with tool-supported modeling conventions.

Analyze modeling conventions for UML
for the purpose of investigating their effectiveness
with respect to model quality and effort
from the perspective of the researcher
in the context of masters students at the TU Eindhoven

Table 1. Goal according to GQM template

3.2 Design

We are interested in the effect of modeling conventions. Therefore the treatment in this experiment is the use of modeling conventions during model development. Additionally we are interested in the use of modeling conventions in combination with a tool to control adherence to the conventions. Hence, we define three treatment levels:

NoMC: no modeling conventions. The subjects use no modeling conventions. This is the *control group*.

MC: modeling conventions. The subjects use the modeling conventions that are described in Section 3.3.

MC+T: tool-supported modeling conventions. The subjects use the modeling conventions and the analysis tool as described in Section 3.3.

The experimental task was carried out as an assignment in a university course. The didactic constraints of the course were that the assignment was carried out in teams of three students which reflects practical software development better than individual work. Most teams consisted of three students. Due to individual circumstances a few exceptions of this team size had to be made (see Section 4). Experience showed that students of the same skill-level tend to gather in self-selected teams, which would lead to large differences between teams in skill and motivation. To avoid this effect we have randomly assigned student to teams.

We have also assigned the teams to treatment levels by randomization. According to [16] this allows us to assume independence between the treatment groups. This assumption is strengthened by the background questions (see Section 4.6). Each team performed the task for one treatment level. Hence we have an unrelated between-subjects design with twelve teams for each treatment level as depicted in Table 2.

Treatment	NoMC	MC	MC+T
Number of Teams	12	12	12

Table 2. Design of the Experiment

3.3 Objects and Task

The experimental objects are a textual description of a system, a set of modeling conventions and a UML analysis tool.

The task of the subjects was to develop a UML model of the architecture of an information system for an insurance company. The required functionality of the system is described in a document of four pages. The system involves multiple user roles, administration and processing of several data types. The complexity of the required system was chosen such that on the one hand the subjects were challenged but on the other hand there was enough spare time for possible overhead effort due to the experimental treatment. The subjects used the Poseidon [2] UML tool to create the UML models.

The task of the teams who received treatment MC and MC+T was to apply modeling conventions during development of the UML model. We described the selection of modeling conventions for this experiment in Section 2.3. The modeling conventions document used in this experiment contains for each modeling convention a unique identifier, a brief descriptive name, a textual description of the convention, and the name of the metric or rule in the analysis tool, that it relates to.

The subjects of treatment MC+T had tool-support to assure their adherence to the modeling conventions. In this study we used the SDMetrics [42] UML analysis tool. SDMetrics calculates metrics and performs rule-checking on UML models. We have customized the set of metrics and rules to allow checking adherence to the modeling conventions used in this experiment. Our customized metrics-definition file is available at [22].

3.4 Subjects

In total 106 students participated in the experiment. The experiment was conducted within the course “Software Architecting” in the fall term of 2005 at the Eindhoven University of Technology (TU/e). This course is taught in the first year of the Masters program in computer science, hence all subjects hold a bachelor degree or equivalent. Most students have some experience in using the UML and object oriented programming through university courses and industrial internships. The students’ background is described in Section 4.6.

The students were motivated to perform well in the task, because it was part of an assignment which was mandatory to pass the course. The student’s motivation is confirmed by the self-assessment.

The students were not familiar with the goal and the underlying research question of the experiment to avoid biased behavior.

3.5 Operation

Prior to the experiment we conducted a pilot run to evaluate and improve the comprehensibility of the experiment materials. The subjects of the pilot experiment did not participate in the actual experiment.

The experiment was conducted as a mandatory assignment of the course “Software Architecting”. Despite prior UML knowledge of the students we presented and explained UML during the course before the experiment. The assignment started with an instruction session where the task was explained to all students and the elementary functions of the Poseidon and SDMetrics tools were demonstrated. Additionally the subjects were provided with the assignment material [22] including a detailed task description, the description of the insurance company system, and instructions of the tools. The instructions included information about the required diagram types (class, sequence, and use case) and about the expected granularity of modeling (to obtain comparable models). Only the teams of treatment MC and MC+T had access to the modeling conventions and the teams of treatment MC+T had exclusive access to the SDMetrics tool. The teams of treatment MC and MC+T were explicitly instructed to apply the treatment regularly and to contact the instructors in case of questions about the treatment. The task description did not include instruction on roles within the teams, but explicitly encouraged cooperation.

The experiment was executed over a period of six weeks. During this period most groups contacted the instructors at least once. This enabled the instructors to observe whether the treatment was applied correctly within the team.

3.6 Data Collection

For the analysis of the experiment collected data on the quality of the UML models, effort data, background data of the subjects, and data about how the subjects experienced the task.

In this study we measured model quality in terms of adherence to rules and metrics. 16 of the 23 modeling conventions are directly related to rules and metrics that we included in the SDMetrics tool (see Table 5). Therefore we used SDMetrics to obtain the relevant data.

We used logbooks to measure the effort needed for the task. The subjects were provided with an Excel Logbook template to record the time spent during the assignment in a uniform manner. The time was recorded for the three activities related to the development of the UML model: modeling itself, reviewing the model and meetings related to the model. We collected the logbooks after the assignment was completed, such that we had the effort data for individual subjects and for the entire teams.

We used a post-test questionnaire to collect data about the subjects’ educational background, experience, how the task was executed and how the task was experienced. The questionnaire was distributed through the university’s internal survey system PollWeb [14] and contained 17 questions.

3.7 Analysis Techniques

For quality and effort we have to analyze number of defects and time in minutes, respectively. These metrics are measured on a ratio scale. We use descriptive statistics to summarize the data. For hypothesis testing we compare the means using a one-way ANOVA test [32]. We have analyzed the data with respect to the assumptions of the ANOVA test and have found no severe violations. The analysis is conducted using the SPSS [1] tool, version 12.0. As this is an exploratory study we apply the significance level of $p=0.10$ to reject the null hypothesis, i.e. we reject the null hypothesis if $p<0.01$.

The data for developers' attitude and background is obtained from the post-test questionnaire, which was designed as a multiple-choice questionnaire. The answers are on a five-point Likert-scale [27] and, hence, measured on an ordinal scale. We summarize the data by presenting the frequencies as percentages for each answer option and providing additional descriptive statistics where appropriate. We compare the equality of answer distributions between different treatment groups using the χ^2 -test [32]. For this test we used Microsoft Excel. We apply the threshold of $p<0.10$ for statistical significance. When we compare three distributions (NoMC, MC and MC+T) a χ^2 value less than 13.36 implies that $p<0.10$. In cases where we compare only two distributions the threshold is $\chi^2 = 7.78$.

4 Results

This section presents the results of the experiment. During the duration of the experiment eight subjects dropped out, which is a mortality rate of 7.5%. As a result three teams consisted only of a single subject. As these teams were distributed evenly over all three treatments we assume that in case results are affected by the reduced team size, all treatments are affected equally. Hence, we do not remove the results of the single-person teams from our analysis. In the treatment group MC+T one team completed the task with only two members (we include its data) and one team did not finish the task (we exclude its data). All other teams consisted of three members.

4.1 Outlier Analysis

To check whether the data is reasonable and to identify invalid data sets we analyze the outliers. Figure 1 shows the boxplots for the size of the obtained models (number of classes), the total amount of time needed by the teams to complete the task and the defect density (number of defects per class) in the obtained models.

We identify two outliers and one extreme outlier for size, one outlier for time and no outliers for defect density. According to Wohlin [41] the reasons for an outlier should be analyzed in order to decide whether to include or to exclude the data point in the analysis. Outliers that are due to a rare event such as an error

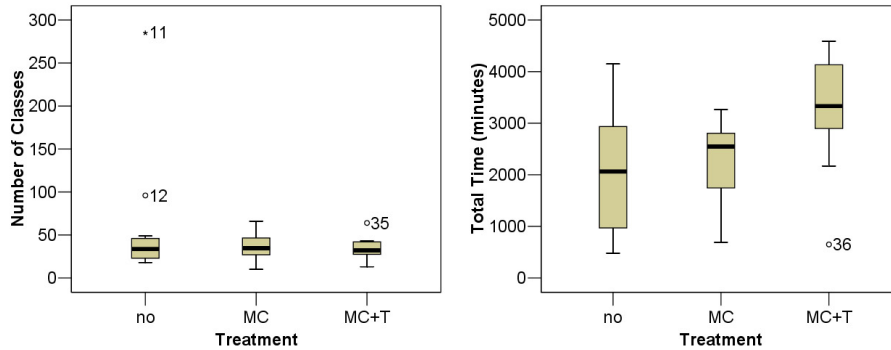


Fig. 1. Boxplots for Number of Classes and Total Time

in the execution of the experiment or lack of subjects' seriosity in the experiment should be excluded. The numbers in the boxplots indicate the internal identifiers of the data set. We have analyzed all collected data (logbooks, UML model, post-test questionnaire) of teams that are identified as outliers to decide what to do with them:

- Outliers 11, 12 and 35: The UML models contain a large number of classes (285 classes for Outlier 11, 96 classes for Outlier 12, 64 classes for Outlier 35). The reason is that the level of detail of the model is very high. For Outlier 11 it is known that subjects in the team were very enthusiastic about UML modeling. The other data of the teams shows no anomalies. As these outliers are not due to a rare situation and can happen in other situations again we decide to not exclude them from the analysis.
- Outlier 36: This team needed only 650 minutes to complete the task. This low number is explained by the drop out of two team members. The other data of the remaining subject shows no anomalies. As drop outs can also happen in other situations and even occur in professional software development we decide to not exclude this data from the analysis.

4.2 H1: Presence of Defects

Total Number of Defects We assess the quality of the UML model in terms of number of defects as described in Section 3.2. Figure 2 shows the boxplot for the total number of defects (on the left) and the number of defects normalized by the size of the model (on the right). Table 3 shows the descriptive statistics. The percentages in Table 3 are relative to the treatment level 'no treatment'. The descriptive statistics for the normalized number of defects show that modeling conventions (MC) improve the mean and the median by 9,5% and 8,9%, respectively. Modeling conventions with tool support (MC+T) improve mean and the median by 18,0% and 17,8%, respectively. As described in Section 3.7 we performed an ANOVA analysis for hypothesis testing. The results shown in

Table 4 reveal that the quality improvements in terms of defect reduction are not statistically significant.

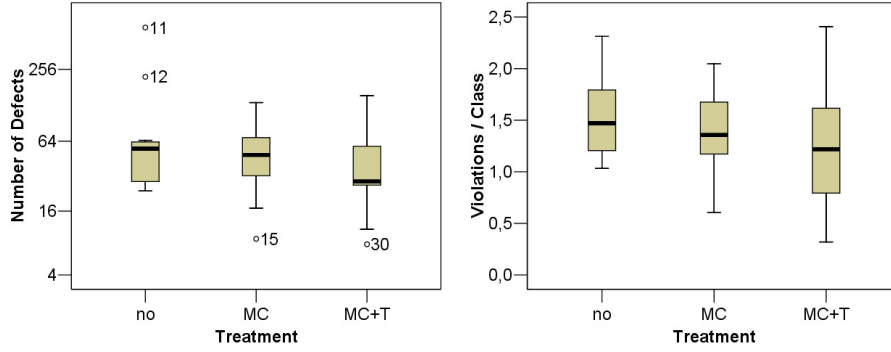


Fig. 2. Boxplots for absolute Number of Defects and Defect Density

	Treatment	Mean	Perc.	Median	Perc.	StDev	Max	Min
Defects (total)	NoMC	102,42	100,0%	55,5	100,0%	157,280	572	42
	MC	53,67	52,4%	49,0	88,3%	34,102	135	9
	MC+T	46,91	45,8%	29,0	52,3%	40,990	154	8
Defects (normalized)	NoMC	1,5181	100,0%	1,4720	100,0%	0,3964	2,312	1,032
	MC	1,3740	90,5%	1,3564	92,1%	0,4121	2,045	0,607
	MC+T	1,2443	82,0%	1,2195	82,8%	0,6671	2,406	0,320

Table 3. Descriptive Statistics for Model Quality

Detailed Results We have collected several metrics of the delivered UML models to investigate the effect of individual modeling conventions. Table 5 summarizes the detailed results of selected metrics. For most modeling conventions we used more than one metric to measure the effect, but for sake of brevity we omit some metrics with similar results in the table. Note that not all of the metrics discussed here contribute to the *Total Number of Defects* discussed before, because some metrics do not measure defects, but other model properties. The Table shows for each metric its name and in column *MC* the identifier of the modeling conventions it relates to (see Table 11 for the description of the modeling conventions). The results are categorized as indicated in column *Cat.* according to the following scheme:

- $A - mean(NoMC) > mean(MC)$ and $mean(NoMC) > mean(MC + T)$,

		Sum of Squares	df	Mean Square	F	Sig.
Defects (total)	Between Groups	21570.1	2	10785.09	1.144	.331
	Within Groups	301708.5	32	9428.39		
	Total	323278.7	34			
Defects (normalized)	Between Groups	.432	2	.216	.858	.433
	Within Groups	8.048	32	.251		
	Total	8.479	34			

Table 4. Results of the ANOVA test for Quality

- $B - \text{mean}(\text{NoMC}) > \text{mean}(\text{MC})$ and $\text{mean}(\text{NoMC}) < \text{mean}(\text{MC} + T)$,
- $C - \text{mean}(\text{NoMC}) > \text{mean}(\text{MC})$ and $\text{mean}(\text{NoMC}) < \text{mean}(\text{MC} + T)$,
- D - result is inconclusive (will be discussed below).

The *Mean*-columns show the mean of the metric for each treatment group. The Δ -columns give the relative difference between NoMC and MC and MC+T, respectively. Note that the reliability of the differences between the reported results varies. We report the results of the ANOVA-test for equality of means in the columns *F* (test statistic) and *Sig.* (p-value) to allow the reader to judge the significance of the difference between the results.

Uniformity of Classes. The modeling conventions 4, 6 and 7 aim at uniformity in the modeling of classes, i.e. modellers should apply the same standards to decide which attributes and (accessor-)methods to model. The results suggest that the majority of the subjects has interpreted modeling conventions 4 and 7 such that they would have the choice to decide whether to define accessors and attributes everywhere or not at all. As a result many subjects decided to not model accessors, which shows that the modeling convention led to more uniformity in modeling accessors. In practice it should be decided whether this kind of uniformity is desirable, otherwise the modeling convention should be rephrased carefully. Treatment MC led to the same effect for attributes (convention 7), but MC+T resulted in an increased use of attributes. The number of empty classes is lower for both treatments than for NoMC. The results significance for Methods per Class is very high, therefore we do not interpret the effect of modeling convention 6 which aims at uniformity of method usage.

Sequence Diagram Coverage is the degree to which the interaction of a model's classes is described by sequence diagrams. A high sequence diagram coverage indicates that a large number of classes is instantiated as objects and the methods occur as messages in sequence diagrams. This is addressed by modeling conventions 9, 10 and 12. The results show that these conventions contribute to a better sequence diagram coverage. The number of classes that are not instantiated and the number of methods that are not called are lower for MC and MC+T than for NoMC. The ratio of sequence diagrams to use cases indicates to what degree the functionality defined in the use cases is indeed described by sequence diagrams, hence, implemented in the system. This ratio is higher for both MC and MC+T than for NoMC.

Design. The modeling conventions 16 and 19 address interconnectivity of and specialization of classes, respectively. The metrics CBO (coupling between objects) and DIT (depth of inheritance tree) [7] measure the effect of the two modeling conventions. Surprisingly, The average CBO for treatment MC+T is higher than for the other treatments. Possible causes could be misinterpretation of the convention or more completeness in modeling associations between classes. However the latter was not addressed by any convention. After analyzing the results we could not interact with the subjects anymore to clarify whether there was a misunderstanding of this convention. The results for DIT are inconclusive, because the largest DIT measured in this experiment was three, hence, no subject violated the modeling convention “DIT at most 7”.

Various Defects. The seven bottom rows of the table present metrics related to various defects. The results show that modeling conventions 11, 13 and 22 are effective to prevent for the associated defects. The metrics related to the conventions 2, 14, 15 and 17 are inconclusive, because the number of occurrences of the defects is so low, that no significant differences can be observed.

Name	MC	Cat.	F	Sig.	NoMC	Mean		Δ	
						MC	MC+T	MC	MC+T
Empty Classes	6,7	A	0,948	0,398	33,11	16,56	12,63	-50,0%	-61,9%
Data Classes	6,7	B	1,390	0,264	7,00	2,78	7,13	-60,3%	1,8%
Methods per Class	6	D	0,067	0,935	2,15	2,02	1,91	-5,7%	-11,1%
Attributes per Class	7	B	3,613	0,038	1,32	0,99	1,80	-25,0%	36,7%
Setters per Class	4	A	2,971	0,066	1,24	0,00	0,96	-100,0%	-21,9%
Getters per Class	4	A	0,912	0,412	0,31	0,23	0,22	-22,4%	-26,9%
Objects	12	B	2,760	0,078	61,89	42,11	72,25	-32,0%	16,7%
Classes not Instantiated	9	A	1,127	0,336	2,81	2,17	1,58	-22,5%	-43,5%
Uncalled Methods	12	A	0,282	0,756	7,95	5,54	5,71	-30,3%	-28,2%
Seq. Diag. per Use Case	10	A	2,695	0,088	0,69	0,90	1,24	30,8%	80,5%
DIT (max)	16	D	1,525	0,233	0,78	1,00	1,50	28,6%	92,9%
CBO (average)	19	C	1,123	0,338	1,21	1,22	1,71	0,5%	41,1%
Objects without Type	11	A	2,007	0,151	0,67	0,00	0,00	-100,0%	-100,0%
Objects without Name	13	A	2,103	0,139	0,22	0,00	0,00	-100,0%	-100,0%
Duplicate Class Names	2	D	0,126	0,882	0,10	0,04	0,11	-55,6%	12,5%
Messages without Name	22	A	3,589	0,039	0,50	0,08	0,00	-83,3%	-100,0%
Abstract Leaf Classes	15	D	0,000	0,000	0,00	0,00	0,00	0,0%	0,0%
Abstract Concrete	17	D	0,000	0,000	0,00	0,00	0,00	0,0%	0,0%
No Message Type	14	D	0,000	0,000	0,00	0,00	0,00	0,0%	0,0%

Table 5. Detailed Results for Model Quality.

4.3 H2: Effort

We measure the effort to develop the UML model in development time. As described in Section 3.6 we collect the effort data using logbooks. Table 6 shows

the descriptive statistics for modeling, reviewing and team meetings. The unit of measurement is person minutes. The columns showing percentages are relative to the treatment level NoMC. The descriptive statistics show that both the mean and the median increase for modeling conventions (MC) are higher for tool-supported modeling conventions (MC+T). Additionally we performed an ANOVA-test for hypothesis testing. The results of the ANOVA-test are shown in Table 7. The results for the total effort are statistically significant. However, when we analyze at the level of activities, we see that only the results of modeling are statistically significant.

	Treatment	Mean	Perc.	Median	Perc.	StDev	Max	Min
Effort (Modeling)	NoMC	1069,17	100,0%	910	100,0%	670,22	2125	120
	MC	1157,92	108,3%	982,5	108,0%	718,225	2280	105
	MC+T	1885	176,3%	2010	220,9%	834,554	3130	540
Effort (Reviewing)	NoMC	367,5	100,0%	300	100,0%	329,224	1155	0
	MC	385,83	105,0%	272,5	90,8%	299,4	900	75
	MC+T	524,55	142,7%	600	200,0%	379,727	1250	0
Effort (Meeting)	NoMC	555,42	100,0%	375	100,0%	499,297	1710	0
	MC	720	129,6%	640	170,7%	632,488	1770	0
	MC+T	862,73	155,3%	690	184,0%	839,069	3060	0
Effort (Total)	NoMC	1992,08	100,0%	2062,5	100,0%	1187,498	4150	480
	MC	2245,42	112,7%	2545	123,4%	852,471	3265	690
	MC+T	3272,27	164,3%	3330	161,5%	1151,838	4590	650

Table 6. Descriptive Statistics for Modeling Effort (in Minutes)

		Sum of Squares	df	Mean Square	F	Sig.
Effort (Modeling)	Between Groups	453675.4	2	2268187.708	4.129	.025
	Within Groups	17580265	32	549383.268		
	Total	22116640	34			
Effort (Reviewing)	Between Groups	166964.89	2	83482.446	.738	.486
	Within Groups	3620239.4	32	113132.481		
	Total	3787204.3	34			
Effort (Meeting)	Between Groups	544447.47	2	272223.736	.614	.547
	Within Groups	14183091	32	443221.597		
	Total	14727839	34			
Effort (Total)	Between Groups	10421703	2	5210851.564	4.535	.018
	Within Groups	36772764	32	1149148.875		
	Total	47194467	34			

Table 7. Results of the ANOVA test for Effort

4.4 Attitude

Objective measurement results for quality and effort are given above. However, to fully investigate the usefulness of modeling conventions it is also necessary to assess the subject's attitude towards modeling conventions. We investigated the subjects's attitude towards the task and the treatment using the post-test questionnaire. The questions are multiple-choice questions with answers on a five-point Likert scale [27] ranging from 1 (very low agreement) to 5 (very high agreement). The results are summarized in Table 8.

The results show that the subjects perceived the difficulty of the task as medium. The mean for the difficulty of performing the task with tool-supported modeling conventions is about 10% higher, but this result is only significant at a p-value of 0.076.

There is a statistically significant difference in the degree to which the subjects enjoyed the task. The mean for the subjects without special treatment (NoMC) is almost one point higher than for the other two treatment groups. The lower enjoyment might be caused by the extra effort (see Section 4.3) for tool-based modeling conventions.

The results show that the subjects of all treatment groups slightly indicate that they have confidence in the quality of their models. There is no significant difference between the treatment groups. Additionally the subjects in the treatment groups MC and MC+T were asked whether they believe that the applied technique improves the quality of the UML model. For modeling conventions with and without tool-support there is slight agreement.

To be able to draw valid conclusions from the experiment it is necessary that the task and the treatments (MC and MC+T) are well understood. Additionally the subjects must be motivated to perform well in the experiment. The answers to the relevant questions show that the task and the treatment were well understood and that the subjects were well motivated. The χ^2 -test did not show significant differences between the treatments for task understanding, motivation, understanding of modeling conventions. If differences would be present this could bias the experiment results.

4.5 How was the task conducted?

We analyzed how the work was organized within the teams and how well the treatment was applied. The results are described in this section.

Organization within the team (Question 9) Figure 3 shows how the subjects organized the work within their teams. The figure shows the results per treatment level and the average results over all participants. The majority of the teams created and reviewed their models together. In about 15% of the teams one person created the model and other persons reviewed the models (either with fixed roles or with changing roles). In only a small fraction of the teams the entire work was done by one single person.

	Treatment	N	Chi-2	Mean	1	2	3	4	5
Difficulty	NoMC	34	11,860	2,94	0,00%	23,53%	61,76%	11,76%	2,94%
	MC	36		3,00	2,78%	19,44%	52,78%	25,00%	0,00%
	MC+T	33		2,61	6,06%	42,42%	36,36%	15,15%	0,00%
Enjoy	NoMC	34	*18,886	3,47	0,00%	14,71%	32,35%	44,12%	8,82%
	MC	36		2,58	16,67%	27,78%	36,11%	19,44%	0,00%
	MC+T	33		2,58	21,21%	21,21%	36,36%	21,21%	0,00%
Confidence in Quality	NoMC	34	5,526	3,18	2,94%	17,65%	41,18%	35,29%	2,94%
	MC	36		3,31	0,00%	11,11%	47,22%	41,67%	0,00%
	MC+T	33		3,24	3,03%	21,21%	27,27%	45,45%	3,03%
MC improves model	MC	35	1,671	3,37	2,86%	17,14%	28,57%	42,86%	8,57%
	MC+T	32		3,59	3,13%	9,38%	21,88%	56,25%	9,38%
Tool improves model	MC+T	33		3,21	6,06%	9,09%	45,45%	36,36%	3,03%
Understanding Task	NoMC	34	4,089	3,18	8,82%	14,71%	35,29%	32,35%	8,82%
	MC	36		3,08	2,78%	27,78%	33,33%	30,56%	5,56%
	MC+T	33		2,91	9,09%	27,27%	30,30%	30,30%	3,03%
MC understanding	MC	36	0,482	3,56	0,00%	11,11%	30,56%	50,00%	8,33%
	MC+T	33		3,45	0,00%	15,15%	33,33%	42,42%	9,09%
Tool understanding	MC+T	33		2,79	9,09%	30,30%	33,33%	27,27%	0,00%
Motivation	NoMC	34	3,862	3,56	5,88%	8,82%	23,53%	47,06%	14,71%
	MC	36		3,44	5,56%	5,56%	36,11%	44,44%	8,33%
	MC+T	33		3,67	3,03%	3,03%	30,30%	51,52%	12,12%

Table 8. Subjects' Attitudes towards the Task

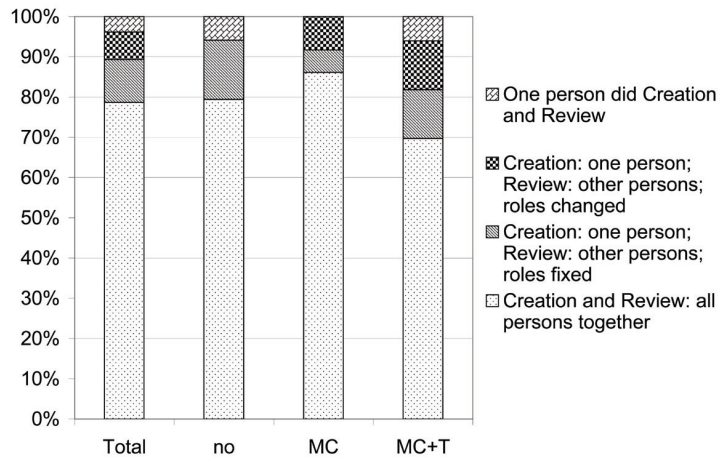


Fig. 3. Organization within the teams.

How were the treatments applied? Besides the organization of the task within the teams we are interested in the way the teams applied the treatment. In particular we are interested in the teams who are exposed to treatment MC and MC+T. We use the answers of the post-test questionnaire for this investigation. First, we are interested to which extent the subjects adhere to the modeling conventions and to the output of the analysis tool. The answers to the adherence-questions are summarized in Table 9. The table shows the percentages for the points ‘1’ (very low adherence) to ‘5’ (very high adherence) for modeling conventions (treatment groups MC and MC+T) and analysis tool (only group MC+T). On average both treatment groups adhere better than neutral to the modeling conventions (the mean is greater than 3). The mean of the teams that have to apply only the modeling conventions is greater than the mean of the teams that apply tool-supported modeling conventions. However, the χ^2 -test shows that the difference is not statistically significant.

The reported average adherence to the analysis tool is below the neutral point (3). We conducted a χ^2 -test to find out whether the adherence differs significantly from the adherence to the modeling conventions (of the same treatment group). The difference is not statistically significant at the commonly accepted 5% significance level. However, the test statistic χ^2 is 9.326 and only slightly below the threshold of 9.49.

Adherence to	Group	N	χ^2	Mean	1	2	3	4	5
Modeling Conventions	MC	36	5,027	3,638	0,00%	5,56%	33,33%	52,78%	8,33%
	MC+T	33		3,303	3,03%	6,06%	54,55%	30,30%	6,06%
Analysis Tool	MC+T	33	9,326	2,727	12,12%	27,27%	42,42%	12,12%	6,06%

Table 9. Adherence to the treatment.

Secondly we asked the subjects how they applied the treatment. Figure 4 shows how the modeling conventions were applied. For both treatment groups that applied modeling conventions, more than 80% of the subjects indicate that they read the modeling conventions several times during the project. The histogram in Figure 5 shows how many times the analysis tool was used to check adherence to the modeling conventions during the project. The maximum was ten times during the project and the minimum was one time. On average the tool was used 3.32 times and the median is 3. We have no data about when the modeling conventions and the tool were applied during the project. But the two authors who were instructors of the course report that they received questions about both the modeling conventions and the analysis tool starting from the second week of the experiment.

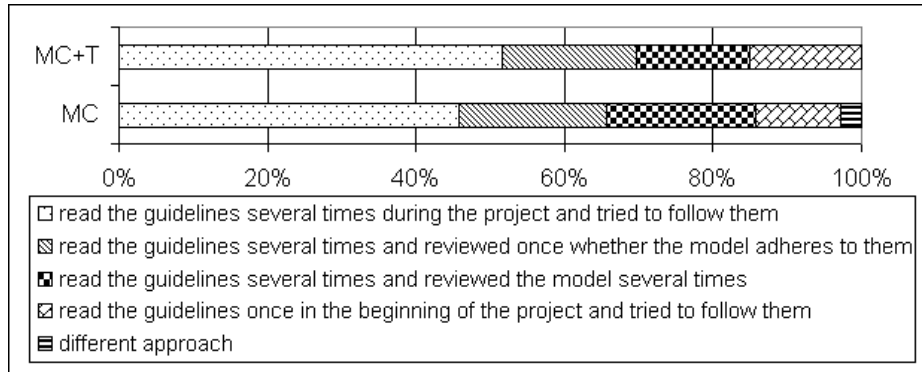


Fig. 4. How were the modeling conventions applied?

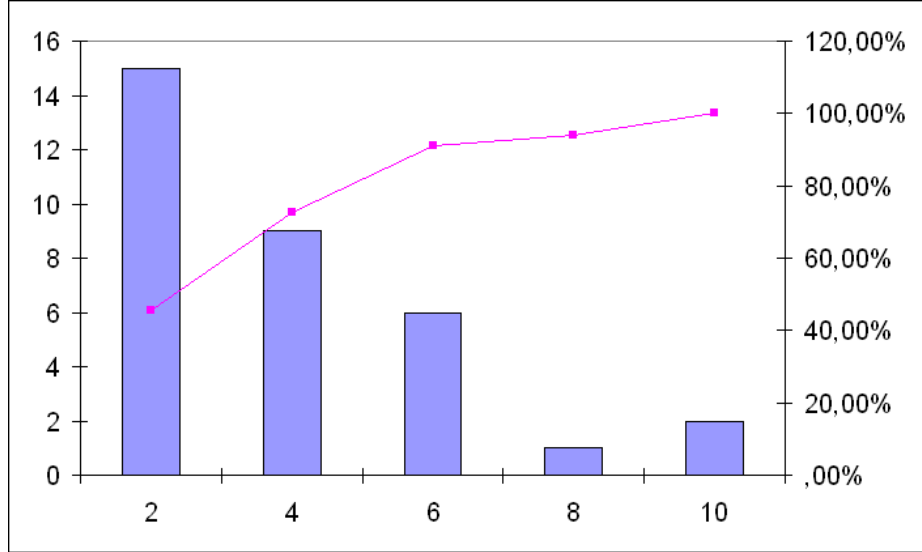


Fig. 5. Histogram for the Frequency of using the SDMetrics Tool within group MC+T?

4.6 Background

Table 10 shows the subjects' answers to the background questions of the post-test questionnaire. The percentages in the five rightmost columns show the proportion of the subjects who have the corresponding experience level for the skill stated in the first column. The skill levels range from '1' (no experience) through '5' (several years of professional experience). The skills are ordered according to relevance for the experiment with the most relevant skill in the first row. The results show that the majority of the subjects has at least some experience with the relevant techniques.

The results are given for each treatment group to analyze whether there are differences in experience level which could be a confounding factor. The test statistic χ^2 is less than the threshold of 15.51 for all skills, i.e. there is no statistically significant difference.

Skill	Treatment	N	χ^2	Mean	1	2	3	4	5
UML	NoMC	34	6,090	2,441	8,82%	52,94%	26,47%	8,82%	2,94%
	MC	36		2,500	11,11%	44,44%	33,33%	5,56%	5,56%
	MC+T	33		2,727	15,15%	36,36%	24,24%	9,09%	15,15%
Software Design	NoMC	34	4,538	2,912	5,88%	32,35%	35,29%	17,65%	8,82%
	MC	36		3,250	2,78%	25,00%	33,33%	22,22%	16,67%
	MC+T	33		2,848	12,12%	27,27%	36,36%	12,12%	12,12%
Design Reviews	NoMC	34	13,044	2,235	17,65%	50,00%	23,53%	8,82%	0,00%
	MC	36		2,528	22,22%	36,11%	19,44%	11,11%	11,11%
	MC+T	33		2,333	30,30%	21,21%	39,39%	3,03%	6,06%
Implementing	NoMC	34	7,928	3,000	8,82%	29,41%	29,41%	17,65%	14,71%
	MC	36		3,361	13,89%	8,33%	33,33%	16,67%	27,78%
	MC+T	33		3,182	15,15%	15,15%	33,33%	9,09%	27,27%
Code Reviews	NoMC	34	13,265	2,294	23,53%	44,12%	17,65%	8,82%	5,88%
	MC	36		3,083	16,67%	22,22%	19,44%	19,44%	22,22%
	MC+T	33		2,727	12,12%	33,33%	36,36%	6,06%	12,12%

Table 10. Educational Background of the Subjects

5 Threats to Validity

Conducting experiments involves threats to the validity of the results. Different threats to validity are conflicting. Hence, prioritizing amongst them is an optimization problem. According to Wohlin et al. [41] the priorities of the threats to validity depend on the purpose of the experiment. The purpose of this study is to explore the effectiveness of modeling conventions for UML. The threats to the validity are discussed in decreasing order.

Internal Validity Threats to internal validity can affect the independent variables of an experiment. A possible threat to internal validity is that the treatment groups behave differently because of a confounding factor such as difference in skills, experience or motivation. We used randomization to assign subjects to treatment groups to avoid differences between treatment groups. In Section 4.6 we analyzed subjects' skill and experience level and in Section 4.4 we analyzed their motivation and understanding. The results show no significant differences between the treatment groups.

To be closer to industrial projects in terms of model size, project duration and project organization, the reported experiment is not conducted as a few-hours in-lab experiment but over a period of six weeks. This involves less control over the subjects and their execution of the tasks. A risk is that subjects are eager to learn about new technology, i.e. apply a treatment from a different group. We took the following precautions to minimize this risk: (i) we did not tell the subjects the goal of the experiment, but they were instructed that applying a different treatment affects the result (ii) we informed the subjects that their grade is not influenced by the treatment group that they were in (iii) we made the modeling conventions and the analysis tool available only to the appropriate teams (iv) we informed the subjects that all technology would be made available to all subjects after completion of the task. Despite these precautions there is no guarantee that subjects did not receive the modeling conventions and the analysis tool from their peers in other groups. However, this would only decrease the effect between the treatment groups. Hence, in case this happened, the effect would be larger in reality.

Mortality, i.e. subjects dropping out of the experiment is discussed in Section 4 and is not regarded as a threat to the validity.

External Validity Threats to external validity reduce the generalizability of the results to industrial practice. As described in Section 3 the experiment is designed to render a realistic situation. The task is to model a software architecture of non-trivial size in a project team using up-to-date tools which are used in industrial practice. Hence, the experimental environment is designed to maximize generalizability (at the cost of statistical significance, which is discussed later in this section). We use students as subjects, which might be a threat to external validity. However, Kitchenham et al. [19] state that students can be used as subjects. All students in this experiment hold a BSc degree and have relevant experience (refer to Section 4.6).

Due to curricular constraints the amount of training and, hence, experience with modeling conventions and the analysis tool is limited, which is a possible threat to external validity. Limited amount of experience renders the situation in the introduction phase of the technology in an industrial environment. We assume that more experience results in a reduction of extra effort and possibly a larger effect on model quality. Hence, the limited amount of experience reduces the observed effectiveness and efficiency of modeling conventions compared with a more matured situation.

Construct Validity Construct validity is the degree to which the variables measure the concepts they are to measure. The concept of quality is difficult to measure and it consists of several dimensions [17][18]. It is not feasible to cover all dimensions in a single experiment. We limit the scope of this experiment to defect containment. Using well-established tooling to measure the defect containment we are confident to measure this dimension of model quality correctly.

Conclusion Validity Conclusion validity is concerned with the relation between the treatment and the outcome. The statistical analysis of the results is reliable, as we used robust statistical methods. As this is an exploratory study to gain insights on the effectiveness of modeling conventions rather than aiming at theory testing we accept that the significance for some results is weak. The significance could have been better if we would have increased the number of data points by defining the task for individual subjects instead of teams of subjects. When we were designing the study we decided that a generalizable task is more important than high statistical power for our explorative study. Therefore we decided that the models should be developed by teams, which is closer to real world software engineering than individual development. Additionally it fitted better with the didactic purpose of the course.

We minimized possible understanding problems by testing the experiment material in a pilot experiment and improving it according to the observed issues. The course instructors were available to the students for clarification questions. The results of the post-test questionnaire show that the task was well understood. Hence, we conclude that there were no understanding problems threatening the validity of the reported experiment.

The metrics of the UML models (defects, size...) were collected using an analysis tool and are therefore repeatable and reliable. A possible threat to the conclusion validity is the reliability of the measured time and the data from the post-test questionnaire. For time collection a logbook template was used to assure uniformity. The reported time data might be imprecise. The data was checked by the authors and seems valid. A discussion of outliers is given in Section 4.1. The data from the post-test questionnaire is of course subjective. However we minimized the risk for different interpretation of the questions by conducting a pilot experiment as described above.

6 Conclusions

The UML consists of different diagram types, has no formal semantics and does not provide guidelines on how to use the language features. Inherent to these characteristics is the risk for quality problems such as defects and non-uniform use of the language. In this study we propose modeling conventions as a forward-oriented means to reduce these quality problems. Our literature review shows that existing work focusses on particular categories of conventions for UML modeling and that there is lack of empirical validation of conventions for UML modeling.

Our main contribution is an experiment that provides empirical data about the application of modeling conventions in a realistic environment. Our results show that the defect density in UML models is reduced through the use of modeling conventions. However, the improvement is not statistically significant. Additionally, we provide data about the additional effort needed to apply modeling conventions with and without tool-support. The presented data quantifies the trade-off between improved model quality by using modeling conventions and the cost of extra effort. Additional observations describe the developers' attitude towards modeling conventions and how the modeling conventions were applied within the development teams. We observed that the adherence to modeling conventions, especially for tool-supported modeling conventions, bears potential for improvement. Furthermore the subjects using modeling conventions enjoyed their task less than the subjects who did not use modeling conventions, indicating that the commitment in using modeling conventions can be improved.

Due to the time constraints of the experiment, we provided the subjects with a set of modeling conventions. However, the subjects had no experience whether the modeling conventions were useful for their task, and the subjects received no reward for delivering a better quality model (the typical reward would be less effort during use of the UML models in a later phase). In practice it would be desirable if the developers who must eventually use the conventions participate in establishing the set of modeling conventions. This would increase their knowledge about and trust in the conventions and we expect they would have more commitment in using modeling conventions. We expect that the commitment will also be improved in a practical situation because the models will be used after they have been developed. The subjects in this experiment were not experienced using modeling conventions or the analysis tool. Therefore the experiment resembles the introduction of modeling conventions to a project. We expect that for more experienced developers the quality improvement is larger and the amount of extra effort will be reduced.

The tool-support for adherence to the modeling conventions was given by a stand-alone tool. We expect that integrating adherence checks into UML development tools will decrease the extra effort and result in higher adherence, because of a shorter feedback loop. Egyed's instant consistency checking [12] is a promising technique for short feedback loops.

The observations made in this experiment potentially lead to the following guidelines for applying UML modeling conventions:

- Attention must be paid to control the adherence to the modeling conventions.
- Commitment of the developers increases the adherence to the modeling conventions.
- Modeling conventions should be tailored for a specific purpose of modeling.
- Tool support to enforce adherence to the modeling conventions increases the quality improvement. A short feedback loop is required to minimize the amount of necessary rework.

In future work the effect of adherence and experience on the effectiveness and efficiency of modeling conventions should be investigated in more detail.

External replications of the reported experiment should be conducted to further confirm our findings. We focussed at syntactical quality of UML models in this experiment. We are conducting a follow-up experiment where we investigate semantic and pragmatic quality.

Acknowledgements. We thank the students from the 2005/2006 course ‘Software Architecting’ at the Technische Universiteit Eindhoven for their participation. The valuable comments of our colleagues Reinder Bril and Richard Verhoeven improved the quality of this paper.

References

1. SPSS, version 12.0. <http://www.spss.com>.
2. Gentleware AG. Poseidon for UML, community edition, version 3.1. <http://www.gentleware.com>.
3. Scott W. Ambler. *The Elements of UML 2.0 Style*. Cambridge University Press, 2005.
4. Victor R. Basili, G. Caldiera, and H. Dieter Rombach. The goal question metric paradigm. In *Encyclopedia of Software Engineering*, volume 2, pages 528–532. John Wiley and Sons, Inc., 1994.
5. James M. Bieman, Roger Alexander, P. Willard Munger III, and Erin Meunier. Software design quality: Style and substance. In *Proceedings of the Workshop on Software Quality (WoSQ)*. ACM, 2001.
6. Lionel C. Briand, Christian Bunse, and John William Daly. A controlled experiment for evaluating quality guidelines on the maintainability of object-oriented designs. *IEEE Transactions on Software Engineering*, (6):513–530, June.
7. S. R. Chidamber and C. F. Kemerer. A metrics suite for object-oriented design. *IEEE Transactions on Software Engineering*, 20(6):476–493, 1994.
8. Peter Coad and Edward Yourdon. *Object Oriented Analysis*. Prentice-Hall, second edition, 1991.
9. Peter Coad and Edward Yourdon. *Object Oriented Design*. Prentice-Hall, first edition, 1991.
10. Reidar Conradi, Parastoo Mohagheghi, Tayyaba Arif, Lars Christian Hedge, Geir Arne Bunde, and Anders Pedersen. Object-oriented reading techniques for inspection of UML models – an industrial experiment. In *Proceedings of the European Conference on Object-Oriented Programming ECOOP’03*, volume 2749 of *LNCS*, pages 483–501. Springer, July 2003.
11. Edsger Wybe Dijkstra. GO TO statement considered harmful. *Communications of the ACM*, 11(3):147–148, March 1968.
12. Alexander Egyed. Instant consistency checking for the UML. In *Proceedings of the 28th International Conference on Software Engineering (ICSE’06)*. ACM, May 2006.
13. Holger Eichelberger. Aesthetics of class diagrams. In *Proceedings of the First IEEE International Workshop on Visualizing Software for Understanding and Analysis (VISSOFT 2002)*, pages 23–31. IEEE CS Press, 2002.
14. Technische Universiteit Eindhoven. Pollweb system. <http://ai5.wtb.tue.nl/enquetes/pollweb.info.php>.
15. M. E. Fagan. Advances in software inspections. *IEEE Tr. on Software Engineering*, 12(7):744–751, 1986.

16. Norman E. Fenton and Shari Lawrence Pfleeger. *Software Metrics, A Rigorous and Practical Approach*. Thomson Computer Press, second edition, 1996.
17. David Garvin. What does ‘product quality’ really mean? *Sloan Management Review*, 26(1):25–45, 1984.
18. Barbara Kitchenham and Shari Lawrence Pfleeger. Software quality: The elusive target. *IEEE Software*, 13(1):12–21, Januari 1996.
19. Barbara A. Kitchenham, Shari Lawrence Pfleeger, Lesley M. Pickard, Peter W. Jones, Dafic C. Hoaglin, Khaled El Emam, and Janett Rosenberg. Preliminary guidelines for empirical research in software engineering. *IEEE Transactions of Software Engineering*, 28(8):721–734, August 2002.
20. Henk Koning, Claire Cormann, and Hans van Vliet. Practical guidelines for the readability of IT-architecture diagrams. In *Proceedings of SIGDOC’02*, pages 90–99, Toronto, Canada, 2002. ACM.
21. Ludwik Kuzniarz, Mirosław Staron, and Claes Wohlin. An empirical study on using stereotypes to improve understanding of UML models. In *Proceedings of the 12th IEEE International Workshop on Program Comprehension (IWPC’04)*, pages 14–23. IEEE CS Press, 2004.
22. Christian F. J. Lange. Material of the modeling conventions experiment. <http://www.win.tue.nl/~clange>.
23. Christian F. J. Lange and Michel R. V. Chaudron. An empirical assessment of completeness in UML designs. In *Proceedings of the 8th International Conference on Empirical Assessment in Software Engineering (EASE’04)*, pages 111–121, 2004.
24. Christian F. J. Lange and Michel R. V. Chaudron. Effects of defects in UML models - an experimental investigation. In *Proceedings of the 28th International Conference on Software Engineering (ICSE’06)*. ACM, May 2006.
25. Christian F. J. Lange, Michel R. V. Chaudron, and Johan Muskens. In practice: UML software architecture and design description. *IEEE Software*, 23(2):40–46, March 2006.
26. Felix Leung and Narasimha Bolloju. Analyzing the quality of domain models developed by novice systems analysts. In *Proceedings of the 38th Hawaii International Conference on Systems Sciences*. IEEE, 2005.
27. Rensis A. Likert. A technique for the measurement of attitudes. *Archives of Psychology*, (No. 140), 1932.
28. Odd Ivar Lindland, Guttorm Sindre, and Arne Sølvberg. Understanding quality in conceptual modeling. *IEEE Software*, 11(2):42–49, March 1994.
29. Sergio Loján-Mora, Juan Trujillo, and Il-Yeol Song. Multidimensional modeling with UML package diagrams. In *Proceedings of the International Conference on Conceptual Modeling, LNCS*, pages 199–213. Springer Verlag, 2002.
30. Tom Love. An experimental investigation of the effect of program structure on program understanding. *ACM Sigplan Notices*, 12(3):105–113, March 1977.
31. Neil MacKinnon and Steve Murphy. Designing UML diagrams for technical documentation. In *Proceedings of SIGDOC’03*, pages 105–112, San Francisco, CA, 2003. ACM.
32. Meerling. *Methoden en technieken van psychologisch onderzoek*, volume 2. Boom, Meppel, The Netherlands, 4th edition, 1989.
33. Steve Murphy, Scott Tilley, and Shihong Huang. Fourth workshop on graphical documentation: UML style guidelines. In *Proceedings of SIGDOC’04*, pages 118–119, Memphis, 2004. ACM.
34. Object Management Group. *MDA Guide, Version 1.0.1*, omg/03-06-01 edition, June 2003.

35. Object Management Group. *Unified Modeling Language, Adopted Final Specification, Version 2.0*, ptc/03-09-15 edition, December 2003.
36. Paul W. Omam and Curtis R. Cook. A taxonomy for programming style. In *Proceedings of the 18th ACM Computer Science Conference*, pages 244–250, 1990.
37. Helen C. Purchase, Jo-Anne Alder, and David Carrington. Graph layout aesthetics in UML diagrams: User preferences. *Journal of Graph Algorithms and Applications*, 6(3):255–279, 2002.
38. Arthur J. Riel. *Object-Oriented Design Heuristics*. Addison Wesley, 1996.
39. Sandra A. Slaughter, Donald E. Harter, and Mayuram S. Krishnan. Evaluating the cost of software quality. *Communications of the ACM*, 41(8):67–73, August 1998.
40. Dabo Sun and Kenny Wong. On evaluating the layout of UML class diagrams for program comprehension. In *Proceedings of the 13th International Workshop on Program Comprehension (IWPC'05)*. IEEE CS Press, 2005.
41. Claes Wohlin, Per Runeson, Martin Höst, Magnus C. Ohlsson, Björn Regnell, and Anders Wesslen. *Experimentation in Software Engineering - An Introduction*. Kluwer Academic Publishers, 2000.
42. Jürgen Wüst. The software design metrics tool for the UML, version 1.3. <http://www.sdmetrics.com>.

A Modeling Conventions

The set of modeling conventions used in this experiment is shown in Table 11.

ID	Name	Description	Category
1	Abstraction Level	Classes in the same package must be of the same abstraction level	Abstraction
2	Unique Names	Classes, packages and use cases must have unique names	Abstraction
3	Size of Use Cases	All use cases should cover a similar amount of functionality	Abstraction
4	Homogeneity of Accessor Usage	When you specify getters/setters/constructors for a class, specify them for all classes	Balance
5	Homogeneity of Visibility Usage	When you specify visibility somewhere, specify it everywhere	Balance
6	Homogeneity of Method Specification	Specify methods for the classes that have methods! Don't make a difference in whether you specify or don't specify methods as long as there is not a strong difference between the classes.	Balance
7	Homogeneity of Attribute Specification	Specify attributes for the classes that have attributes! Don't make a difference in whether you specify or don't specify attributes as long as there is not a strong difference between the classes.	Balance

8	Dynamic Classes	For classes with a complex internal behaviour, specify the internal behaviour using a state diagram	Completeness
9	Model Class Interaction	All classes that interact with other classes should be described in a sequence diagram	Completeness
10	Use Case Instantiation	Each Use Case must be described by at least one Sequence Diagram	Completeness
11	Specify Object Types	The type of ClassifierRoles (Objects) must be specified. (Which class is represented by the object?)	Completeness
12	Call Methods	A method that is relevant for interaction between classes should be called in a Sequence Diagram to describe how it is used for interaction.	Completeness
13	Role Names	ClassifierRoles (Objects) should have a role name	Completeness
14	Specify Message Types	Each message must correspond to a method (operation)	Consistency
15	No Abstract Leafs	Abstract classes should not be leafs (i.e. child classes should inherit from abstract classes)	Design
16	DIT at most 7	Inheritance trees should have no more than 7 levels	Design
17	Abstract-Concrete	Abstract classes should not have concrete superclasses	Design
18	High Cohesion	Classes should have high cohesion. Don't overload classes with unrelated functionality.	Design
19	Low Coupling	Your classes should have low coupling. (The number of relations between each class and other classes should be small)	Design
20	No Diagram Overload	Don't overload diagrams. Each diagram should focus on a specific concept/problem/functionality/	Layout
21	No X-ing Lines	Diagrams should not contain crossed lines (relations)	Layout
22	Use Names	Classes, use cases, operations, attributes, packages, etc must have a name	Naming
23	Meaningful Names	Naming should use commonly accepted terminology, be non-ambiguous and precisely express the function / role / characteristic of an element.	Naming

Table 11: List of Modeling Conventions used in this Experiment.

B Post-test Questionnaire

We present the questions of the post-test questionnaire in this section. The questionnaire conducted as an online-questionnaire using the PollWeb [14] system of the Technische Universiteit Eindhoven. The student were identified by the system while logging on. This enables us to relate the data from the questionnaire to the other data obtained during the experiment.

B.1 Introduction

During the execution of the course assignment you took part in a research experiment on the usefulness of modeling standards and metrics tooling. By completing this questionnaire you provide us with useful information for the analysis of the results. The information you give here does not influence your mark and will be treated confidentially! Please answer each question carefully. Thank you!

B.2 Background Questions

1. What is your practical work experience in software engineering (in years)?
2. What is your knowledge in the following fields?

Please answer according to this scale: 1 = no knowledge; 2 = gained knowledge through academic classes or literature study; 3 = applied it in academic context; 4 = applied it in one industrial project; 5 = applied it in more than one industrial project

	1	2	3	4	5
Unified Modeling Language (UML)	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Designing Software Systems	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Implementing Software Systems	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Reviewing Source Code	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Reviewing Software Designs	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Software Inspections	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

B.3 About the Assignment

3. Indicate how difficult the UML modeling task was to you.

1 – very difficult	2 – difficult	3 – intermediate	4 – easy	5 – very easy
<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

4. Indicate how well you understood what was required of you in the UML modeling task.

1 – very poor	2 – poor	3 – intermediate	4 – good	5 – very good
<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

5. What is the confidence of the quality of the UML model you delivered?

1 – very poor	2 – poor	3 – intermediate	4 – good	5 – very good
<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

6. How much did you enjoy developing the UML model using traditional modeling / modeling standards / modeling standards + analysis tool? (chose what is applicable for you)

1 – very poor	2 – poor	3 – intermediate	4 – good	5 – very good
<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

7. Indicate how motivated you were to perform well in the UML modeling task.

1 – very poor	2 – poor	3 – intermediate	4 – good	5 – very good
<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

8. How was the work organized within your group ?

- one person developed the model on his own
- one person created the model, other persons reviewed the model (fixed roles)
- one person created the model, other persons reviewed the model (roles changed over project)
- several persons created the model and reviewed the model together
- different approach:

B.4 Questions about Modeling Conventions

Only the subjects who used modeling conventions were asked to answer the questions of this section.

9. How well did you adhere to the modeling conventions?

1 - not at all	2 - poor	3 - intermediate	4 - good	5 - very good
<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

10. How well did you understand the meaning of the modeling conventions?

1 - very poor	2 - poor	3 - intermediate	4 - good	5 - very good
<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

11. Indicate your approach, how you applied the modeling conventions.

- read the guidelines once in the beginning of the project and tried to follow them
- read the guidelines several times during the project and tried to follow them
- read the guidelines several times during the project and reviewed whether the model adheres to the guidelines once
- read the guidelines several times during the project and reviewed whether the model adheres to the guidelines several times
- different approach:

12. Do you thin development using modeling standards lead to a better UML model?

1 - not at all	2 - rather not	3 - neutral	4 - probably yes	5 - yes, a lot
<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

B.5 Questions about using the Conformance Tool

Only the subjects who used the conformance tool were asked to answer the questions of this section.

13. How well did you adhere to the critics and analysis results of the SDMetrics tool?

1 - not at all	2 - poor	3 - intermediate	4 - good	5 - very good
<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

14. How well did you understand the meaning of the tool output?

1 - not at all	2 - poor	3 - intermediate	4 - good	5 - very good
<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

15. How many times did you analyze the model during the project using the tool?

16. Do you think tool-based model-analysis leads to a better UML-model?

1 - not at all	2 - rather not	3 - neutral	4 - probably yes	5 - yes, a lot
<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>