Usefulness of Screen Mockups in Use Case Descriptions - A Formal Experiment

Filippo Ricca(1), Giuseppe Scanniello(2), Marco Torchiano(3), Gianna Reggio(1), Egidio Astesiano(1)

(1) DISI, Università di Genova, Italy
(2) Università della Basilicata, Potenza, Italy
(3) Politecnico di Torino, Torino, Italy

`filippo.ricca@disi.unige.it, giuseppe.scanniello@unibas.it, marco.torchiano@polito.it, gianna.reggio@disi.unige.it, astes@disi.unige.it`

Abstract

Use Cases represent a popular modeling technique amongst practitioners to capture and define requirements in the software development process. Different techniques, templates, and styles have been proposed in the literature to describe Use Cases. Some of these introduce scree mockups to increase the comprehension of functional requirements and then promote the communication among stakeholders, analysts, and developers.

This paper reports on a controlled experiment to assess the effectiveness of including screen mockups in the Use Case descriptions to comprehend software requirements. The context of the experiment is constituted of students of the Bachelor program in Computer Science. Results indicate a clear improvement in terms of understandability of software requirements when screen mockups are present with no significant impact on effort.

Keywords: Use Cases, Screen Mockups, Requirements comprehension, Controlled experiment.

1 Introduction

Well defined software requirements and communication of analysis information are recognized as vital factors for the success of software systems development. It is well know that a substantial proportion of code defects (as high as 85%) originates at the requirements phase [1] and that misunderstandings are costly in software development. The root causes for such defects are due to ambiguous, incomplete, inconsistent, silent (unexpressed), unusable, over-specific or verbose requirements [2] and to communication problems among stakeholders, analysts, and developers.

Use Cases are a simple way to capture and define requirements of a software system. They very often represent the starting point of a software development process and are recognized as a useful and powerful tool for describing software requirements and interacting with stakeholders. While much has been said and written on the benefits of Use Cases, surprisingly little empirical research has been carried out on this relevant topic.

According to the popularity of Use Cases, the basic principles of cognitive theory of multimedia learning [3] and to our previous positive experiences in industrial and academic contexts [4] it seems reasonable to us investigating the following research question. *Do screen mockups provide a more effective way to increase the comprehension of functional requirements with respect to the only adoption of Use Cases?.*

In this paper, we present a controlled experiment conducted with students of the Bachelor program in Computer Science at the University of Basilicata to compare the effectiveness of Use Cases with and without the support of screen mockups in the comprehension of software requirements. The goal of this experimentation is twofold. First, we evaluate the screen mockups effects in the software requirements comprehension. To this end, we have used a questionnaire based approach (as in [5, 6]) that requires the subjects to perform the following operations: reading and understanding the questions, surfing the Use Case descriptions, reasoning about the domain, focusing the attention on the actions executed by the actors on the system, and finally, answering to the questionnaire. The second concerns the evaluation of the screen mockups impact on the needed effort to understand software requirements. Results indicate that benefits are obtained adopting screen mockups with no substantial increase/decrease of effort.

The paper is organized as follows. Section 2 provides the study definition and details of the adopted experiment design, while Section 3 reports the obtained results. Section 4 discusses the results and threats to validity of the experiment. Section 5 presents related works. Finally, section 6 concludes the paper and outlines future work.

2 Study definition, design and procedure

This section presents the definition, the design, and the planning of the experiment, structured according to the guidelines by Wohlin *et al.* [7].

2.1 Context description

The experiment was conducted within a research laboratory at the University of Basilicata with second year students of the Bachelor program in Computer Science. This experiment represents an optional didactic activity of a Software Engineering course. On the other hand, as main and mandatory laboratory activity of the course, the students were grouped in teams and allocated on projects to design and develop software systems with a distributed architecture (typically a multi tiered) using Java, Web technologies and a relational DBMS. Note that the students had also attended courses on basic and advanced object oriented programming before carrying out the experiment. A course on database systems was attended as well.

The requirements analysis document of two different software systems were considered for the experiment. The systems are similar in complexity and refer to application domains in which the subjects are not completely familiar with. The first is a system (EasyCoin) for cataloging collections of coins, while the second is a system (AMICO) that helps Property Managers to manage a condominium (an activity very relevant in Italy). The requirements documents of EasyCoin and AMICO were realized (respectively during the academic years 2006/2007 and 2008/2009) by one of the authors (G. Reggio) for the projects of the Software Engineering course at the University of Genova [4]. The two documents are comparable also in size[1] as number of pages (both 25), number of Use Cases (19 for AMICO and 20 for EasyCoin) and number of screen mockups (31 for AMICO and 32 for EasyCoin). They are small enough to fit the time constraints of the experiment.

2.2 Experiment definition and hypotheses formulation

The *goal* of the experiment was to assess the improvement (if any) provided by the presence of screen mockups in Use Case descriptions, both in terms of comprehension level and comprehension effort. Use Cases are widely used in the requirements engineering field to describe system functionality from the end user's point of view. They are textually specified according to a more ore less rigorous template, which generally enables the definition of a sequence of simple steps to describe the interaction between one or more actors and the system to develop. Requirements analysis documents that exploit Use Case to specify software functionality [8] generally also include one or more UML Use Case diagrams [9] to show Use Cases, actors, and their mutual relationships. In this investigation we used documents containing:

- the mission of the software system;
- a UML Use Case diagram;
- the descriptions of the Use Cases according to the SWEED template[2] (see Figure 1). According to the experiment design, steps within a Use Case may be accompanied by screen mockups presenting what an actor will see in that moment. Figure 2 shows an example of screen mockup regarding the EasyCoin software system;
- a glossary including the needed definitions to comprehend the requirements specification.

[1]the following numbers are referred to the versions with screen mockups

[2]further details can be found at http://lglwww.epfl.ch/research/use_cases/

USE CASE: Insert Coin
Level: User-Goal
Intention in context: the collector wants to insert a coin in the collection
Primary actor: coin collector
Precondition: a non-void list of issues is selected
Main success scenario:
1 the collector chooses an issue of the list and asks for inserting a coin
2 the system asks for coin info **4 (**see Insert Coin screen mockup**)
3 the collector inserts the info and presses insert button
4 the system shows the new inserted coin to the collector and the Use Case ends with success

Figure 1: Insert Coin Use Case. (**4) refer to the item 4 of the glossary (not shown here)

In this study we are particularly interested in Use Cases as a communication tool among Analysts and Designers. In this context, a good comprehension of the requirements is vital for realizing the right system, moreover reducing the comprehension effort allows reducing the development time. The *perspective* is both of *Researchers*, investigating on the effectiveness of screen mockups in Use Case descriptions, and of *Project managers*, evaluating the possibility of adopting Use Case descriptions augmented with screen mockups in their organization. Accordingly, we have defined the following null hypotheses:

H_{l0} When performing a comprehension task, the presence of screen mockups in Use Case descriptions **does not significantly improve** the comprehension level.

H_{e0} When performing a comprehension task, the presence of screen mockups in Use Case descriptions **does not significantly reduce** the comprehension effort.

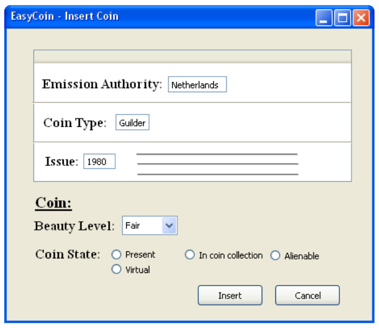
The objective of the statistical analysis will be rejecting the above null hypotheses and possibly accepting the following alternative hypotheses:

H_{la} When performing a comprehension task, the presence of screen mockups in Use Case descriptions **significantly improves** the comprehension level.

H_{ea} When performing a comprehension task, the presence of screen mockups in Use Case descriptions **significantly reduces** the comprehension effort.

2.3 Experiment design, material, procedure and pilot

We used the counterbalanced experiment design [7] shown in Table 2.3, which includes two objects, two tasks, four groups, and two treatments. This design ensures that each subject works on different *Objects* (AMICO or EasyCoin) in the two *Tasks*, receiving each time a different *Treatment* (S = Use Cases+Screen



Emission Authority: Netherlands

Coin Type: Guilder

Issue: 1980

Beauty Level: Fair

Coin State: ○ Present ○ Virtual ○ In coin collection ○ Alienable

Figure 2: Insert Coin screen mockup

mockups or T[3] = Use Cases only). The chosen design mitigates possible learning effects between tasks and permits the use of statistical tests (e.g., two-way ANOVA) for studying the effect of other factors (e.g., Object) [7]. It is important to say that, the screen mockups provide no additional information that could not be derived from the Use Cases.

We used a pre-questionnaire to get some information on the subjects of the experiment; in particular industrial experience and average grade of previous exams. Then we split the subjects into four groups (Group 1, Group 2, Group 3 and Group 4) equally distributing high and low experience/ability subjects among groups.

	Group 1	**Group 2**	**Group 3**	**Group 4**
Task 1	EasyCoin-S	EasyCoin-T	AMICO-T	AMICO-S
Task 2	AMICO-T	AMICO-S	EasyCoin-S	EasyCoin-T

Table 1: Experiment design

Two weeks before the experiment, a lesson of 2 hours reminded notions of software requirements and introduced Use Case modeling with the notation used in the experiment. An exercise similar to the tasks of the experiment was conducted one week before using a simple system (LaTazza) used also in other experiments (e.g., [10]).

The experiment was conducted by the students in a Laboratory equipped with computers in one Lab session, composed of two tasks (lasting approximately 1 hour each) with half an hour break between the two tasks.

The experimental material handed to the students included: (i) a MS Word file[4] of the requirements document, (ii) a paper copy of comprehension ques-

[3]T is for Text only

[4]we chose an electronic format to permit the ”Find” facilities that is very convenient with large documents

tionnaires, (ii) a paper copy of the final post-experiment questionnaire[5], and (iv) the slides of the lesson.

Similarly to [5], the defined *comprehension questionnaire* is composed of 10 open questions on the assigned systems. Questions are divided into three categories. The first three questions concern the domain of the system (*domain*), the following four concern the interaction with the system that will be built (i.e., input, output), and the execution flow (*IO*). The last three concern implementation/development of the system (*development*). For example, the question Q3 related to the domain of EasyCoin is: "*A coin in EasyCoin contains the following information: beauty level and coin state. Report an example of coin state.*". The answer can be easily deduced (e.g., present-alienable) from the linked screen mockups (see Figure 2) or from the textual description provided in the glossary (item 4). As suggested in [11], we also collected data on the source of information used to answer the questions of the *comprehension questionnaire.* To this end, we added an additional question for each question: "*Did the answer come from the Glossary (G), previous Knowledge (K), Screen mockups (S), Use Case descriptions (UC) or Use Case Diagrams (UCD)?*".

After receiving the experimental package, the students went through, for each task in the lab session, the following procedure:

1. specifying surname, name, start-time in the comprehension questionnaire;

2. answering independently at the questions surfing the Use Case descriptions;

3. (at the end) marking the end-time of the task in the comprehension questionnaire.

We did not suggest any approach (e.g., read a question and use the "Find" facilities of MS Word to circumscribe the portion of interest) on how facing the comprehension tasks. We only discouraged reading completely the Use Case descriptions (they are too long) before starting with the comprehension tasks. As a consequence each student chose a personal approach. After the two tasks, the students compiled the unique final post-questionnaire.

Pilot experiment: a pilot experiment was performed before the execution of the experiment: (i) to evaluate the comprehension questionnaires, and (ii) to compute the effort necessary to execute the tasks. Two students (i.e., one of the Bachelor program and one of the Master program in Computer Science at the University of Genova) and two of the authors not involved in the preparation of the material executed the tasks using Use Cases and screen mockups. For students and authors it took, respectively, 59, 65, 31 and 29 minutes. At the end, they filled the comprehension questionnaires. The gathered data were used only for tuning the experimental material. After this pilot experiment, it was concluded that the experiment was well suited for students of the Bachelor

[5]The post-experiment questionnaire aimed at gaining insights about the subjects' behavior during the experiment and at better explaining the obtained quantitative results. See below for further information about it

program and the total time of 3 hours sufficient. According to the comments of the volunteers involved in the pilot, minor changes at the comprehension questionnaires were made.

2.4 Variable selection

In this experiment we have only one independent variable that is a nominal with two possible values: $\{S, T\}$ (S = Use Cases+Screen mockups, T = Use Cases only). Additional variables that are measured in this study are the Object ($\in \{AMICO, EasyCoin\}$), the Task ($\in \{Task1, Task2\}$) and the Question category ($\in \{domain, IO, development\}$). These latter variables will be used in the analysis phase to get a more precise view of the results.

The dependent variables of our study are the *comprehension level* of the use Cases achieved and the *comprehension effort* required to complete the tasks (as suggested in [11]). Similar to [12], the comprehension level of the subjects has been assessed, using the following approach based on information retrieval theory. Since the plausible answers to each question consist of a list of string items (e.g., id, name, surname, address) we can define as:

$A_{s,i}$ the set of items mentioned in the answer to question i by subject s;

C_i the known correct set of items expected for question i.

Based on the above definition, we can compute *precision* and *recall* [13] for each answer. Precision measures the fraction of items in the answer that are correct:

$$precision_{s,i} = \frac{|A_{s,i} \cap C_i|}{|A_{s,i}|}$$

Recall measures the fraction of correct items that are in the answer:

$$recall_{s,i} = \frac{|A_{s,i} \cap C_i|}{|C_i|}$$

Since the two above metrics measure two different concepts, it may be difficult to balance between them. Accordingly, we used an aggregate measure, $F{-}Measure$ ($\in [0..1]$) [13], which is a standard combination of precision and recall, defined as:

$$F{-}Measure_{s,i} = \frac{2 \cdot precision_{s,i} \cdot recall_{s,i}}{precision_{s,i} + recall_{s,i}}$$

To obtain a single measure representing the comprehension level achieved by a subject, we computed the overall average F-Measure over all the questions of the questionnaire.

Instead, the comprehension effort is measured as time to answer the questionnaire. It was recorded directly by subjects noting down their start and stop time.

Pre and Post-questionnaires: The pre-questionnaire was mainly used to obtain some information about participants (e.g, the industrial working experience and GPA[6]) to assess the ability level and experience of each involved subject. It was useful to divide the subjects into groups trying to balance them.

The post-experiment questionnaire, used to better understanding quantitative results, was composed of seven questions. A first group of questions (**Q1** through **Q5**) concerned the availability of sufficient time to complete the tasks, the clarity of the Use Case, and the ability of subjects to understand them. **Q6** was devoted to measure the perceived usefulness of the screen mockups, while **Q7** aimed at measuring how much time, in percentage intervals, was spent to analyze Use Cases and screen mockups. All the questions, except **Q7** that is expressed in intervals of percentages, expected closed answers according to a five point Likert scale [14]: (1) strongly agree, (2) agree, (3) neither agree nor disagree, (4) disagree, (5) strongly disagree.

2.5 Analysis procedure

In all our statistical tests we decided (as usual) to accept a probability of 5% of committing Type-I-error [7], i.e., of rejecting the null hypothesis when it is actually true.

Because of the sample sizes (number of subjects = 33) and the distributions (they are not normal) we adopted non-parametric tests to reject the null hypotheses. In particular we selected Mann-Whitney and Wilcoxon tests because they very robust and sensitive. Moreover, we used one-tailed statistical tests due to the directionality of the hypotheses. We also performed paired analysis whenever possible.

While the statistical tests allows for checking the presence of significant differences, they do not provide any information about the magnitude of such a difference. The Cohen standardized difference between two groups [15], is defined as the difference between the means (M_1 and M_2), divided by the pooled standard deviation (σ_p) of both groups $d = (M_1 - M_2)/\sigma_p$. It measures the strength of the relationship between the main factor treatment and a dependent variable and gives important information about the magnitude of the effect. Typically, the effect size is considered small for $d \in theinterval(0.2; 0.5($, medium for $d \in (0.5; 0.8($ and large for $d \geq 0.8$.

We expected that the screen mockups is an effective source of information for better comprehending software requirements. The relevance of a source of information can be measured as the proportion of subjects using it to answer a question. An average importance source can be expected to be used by $1/n_s$ subjects, where n_s is the number of alternative sources. We assess the primary relevance of screen mockups as information sources by testing the proportion of users using them (when present) to be greater then the average proportion; for this purpose we used a proportion test [16].

Finally, we measured the effect of other factors on the dependent variables,

[6]Grade Point Average

namely of *Task* (i.e., whether a result was obtained in the first or second working session, to evaluate a possible learning effect), *System* and of the different *Question category* using a two-way Analysis of Variance (ANOVA).

3 Analysis

In the following subsection we present the data analysis of the controlled experiment presented here. As mentioned above, 33 were the involved subjects, who participated all in both the labs.

3.1 Comprehension Level

A composed boxplot[7] of the F-Measure vs. Treatment (S and T), including also cofactors Object and Question category is presented in Figure 3. The two diagrams on the left present results divided by Object and Question category. In the middle only the dimension Object is considered. Finally, on the right is represented the overall boxplot.

Table 2 presents a summary of the comprehension level according to Treatment and grouped by Question category and Object, in addition p-values of unpaired Mann-Whiteny tests are reported. Mann-Whitney tests resulted in a $p \ll 0.001$ (in particular $6.3 \cdot 10^{-6}$ and $1.0 \cdot 10^{-8}$ for un-paired and paired tests respectively). We can therefore reject the null hypothesis with a high degree of confidence as far as the average of questions is concerned. The practical difference is large as we can infer from the Cohen's $d = 1.2$.

	S			T			
Object	mean	med	σ	mean	med	σ	p
Overall	0.63	0.67	0.17	0.43	0.45	0.14	<**0.01**
AMICO	0.64	1.00	0.43	0.40	0.31	0.43	<**0.01**
develop	0.48	0.50	0.41	0.47	0.50	0.40	0.47
domain	0.79	1.00	0.38	0.37	0.00	0.44	<**0.01**
IO	0.64	1.00	0.44	0.37	0.00	0.45	<**0.01**
EasyCoin	0.62	0.75	0.42	0.46	0.50	0.44	<**0.01**
develop	0.39	0.50	0.40	0.19	0.00	0.33	<**0.01**
domain	0.81	1.00	0.34	0.68	0.86	0.40	**0.02**
IO	0.64	0.71	0.42	0.51	0.67	0.43	**0.04**

Table 2: Summary of comprehension level by Object and Question category, and p-values

The effect of the Object is not relevant as we can evince from the Figure 3, a significant difference can be found both systems ($p < 0.001$ for AMICO and $p = 0.001$ for EasyCoin). Such differences are also practically significant, $d = 1.27$ for AMICO and $d = 1.19$ for EasyCoin. The lack of interaction between

[7] A boxplot is a compact representation of the distribution of a variable: the thick horizontal line is the median, the box stands for the 2^{nd} and 3^{rd} quartiles, the whiskers represent the 1^{st} and 4^{th} quartiles.

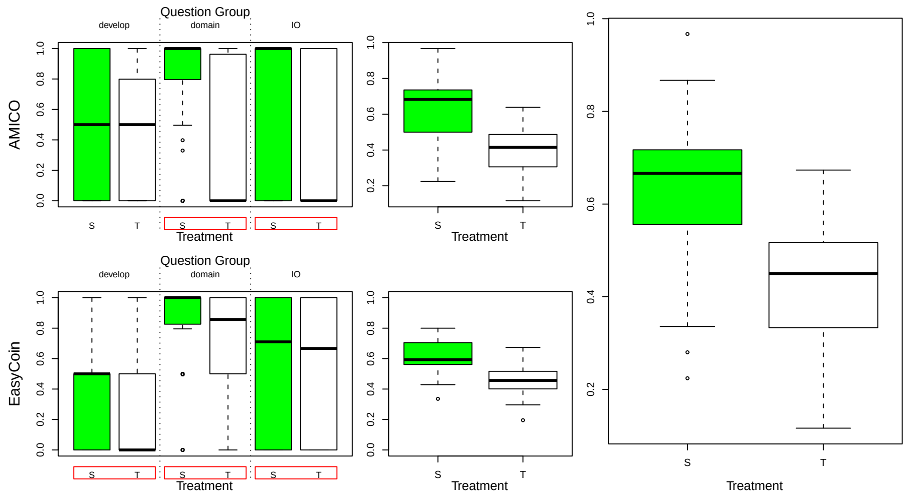


Figure 3: Boxplots of F-Measures averaged over all the questions of the questionanire. Left by (Treatment, Object, Question category), middle by (Treatment, Object) and right by Treatment only.

Treatment and Object can be confirmed by means of a two-way ANOVA as shown in Table 3.1.

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
Treatment	1	0.62	0.62	24.20	0.0000
Object	1	0.01	0.01	0.32	0.5749
Treatment:Object	1	0.03	0.03	1.08	0.3017
Residuals	62	1.58	0.03		

Table 3: Two-way ANOVA of Treatment and Object

Another important co-factor to be considered is represented by questions. In particular we consider the question under three main categories: $\{domain, IO, development\}$. For the object AMICO we observed a significant difference in two Question categories (domain and IO), while for EasyCoin we observed a significant difference for all the three categories. The significant differences have been highlighted with a rectangle in the two leftmost sub-diagrams in Figure 3.

When looking at the specific questions for each Object the situation is less clear cut: for AMICO we observe relevant difference in five questions, with the remaining question providing non definitive answer; for EasyCoin the relevant difference can be found in three questions.

Summarizing, we can reject the null hypothesis H_{l0} both overall and considering the two Objects separately. In addition the hypothesis can be rejected for most Question category, with the exception of the development Question category for AMICO.

3.2 Comprehension Effort

The statistical tests did not revealed any significance difference in the overall effort when comparing the two treatments: paired Mann-Whitney test resulted in a $p = 0.22$.

When considering also cofactor Object (see Figure 4) we observe no significant difference for AMICO ($p = 0.6$), and a significant difference for EasyCoin ($p = 0.02$). This latter difference can be considered medium sized (Cohen's $d = 0.78$) from a practical perspective.

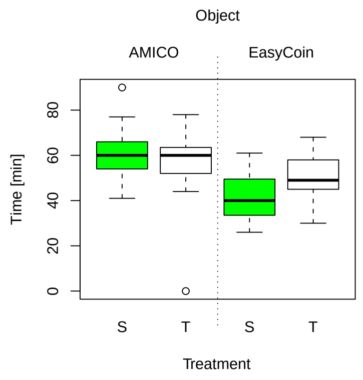


Figure 4: Boxplot of Effort by Treatment and Object.

Summarizing, even if we cannot reject the null hypothesis H_{e0} overall, we can observe that the direction is in favor of screen mockups. Overall medians and means are respectively (54.00, 51.12) for Use Cases+screen mockups and (55.00, 52.61) for Use Cases only. Also, we can reject the null hypothesis for one of the two objects: EasyCoin.

3.3 Source of information

The sources of information used to answer the questions, as declared by the subjects, are described in table 3.3. The data are reported aggregated by Question category, i.e. development, domain, and IO.

The table reports in boldface the numbers, only for the screen mockup (S) source, when the proportion of subject is significantly greater than the average expected (see Section 2.5). We can observe that, based on usage frequency, screen mockups represent a relevant and effective information source in all cases except for the domain related questions relative to Object AMICO.

3.4 Post-questionnaire

An analysis of the post-questionnaires revealed that: subjects judged time to complete the tasks sufficient (Q1), they found questions clear (Q2), and they

	Treatment	**S**			**T**		
	Question category	develop	domain	IO	develop	domain	IO
AMICO	Glossary	3	7	4	7	28	24
	Previous knowledge	13	3	1	18	5	3
	Screen mockups	9	**37**	**43**	0	1	0
	Use cases	15	1	12	17	5	20
	Use case diagrams	0	1	1	3	0	6
EasyCoin	Glossary	6	13	5	13	17	32
	Previous knowledge	17	0	0	22	13	2
	Screen mockups	**20**	**21**	**35**	0	0	0
	Use cases	0	4	17	5	4	30
	Use case diagrams	0	9	5	0	12	2

Table 4: Source used to answer questions (by Question category).

were able to understand both Use Cases and Use Case diagrams easily (Q3 and Q4). All the subjects found the exercise useful (Q5).

The perceived usefulness of screen mockups (Q6) is very high: on scale from 1 (very useful) to 5 (very useless) the median was 1 and the mean 1.3. There is strong evidence that most subjects considered the screen mockups very useful. The subjects estimate (Q7) they spent on average 50% of the time looking at screen mockups.

4 Discussion

The results of the experiment described in this paper provide strong evidence that the presence of screen mockups in Use Case descriptions yields a large advantage (Cohen's $d \geq 1.2$) in terms of **comprehension level**.

More interesting observation can be done by discriminating among different applications (i.e., the Objects of the experiment) and topics of questions (*Question category*). While the domain of EasyCoin (i.e., coin collection) is quite popular, the subjects were less familiar with the domain of AMICO (condominium management). We can observe that the benefit attainable from the presence of screen mockups is inversely proportional to the familiarity (further investigation is required in this respect).

When focusing on specific topics, we expected the IO and domain *Question category* to benefit more from screen mockups while the development *Question category* was expected to be affected less because screen mockups are mostly related with the user interface and scarcely, if not at all, with internal implementation details. In practice we observed that the only non-significant difference was found for the development *Question category* in AMICO, while a significant difference was detected in the other application, namely EasyCoin. In this case we speculate that the familiarity played as a counterpoint to the general behavior: a partial knowledge of the domain (as for EasyCoin) helped extract a little bit of extra information from screen mockups that made the difference, while in absence of such knowledge the information lying in the screen mockups could not be leveraged.

The above results in terms of comprehension level find a plausible explanation in the more intensive use of screen mockups to gather useful information. This interpretation is fully confirmed by the data about the sources of information that subjects declared to use (see table 3.3). We can observe that screen mockups, when present caught most of attention. This is always true except in one case: the development *Question category* for AMICO where a limited use of screen mockups resulted in a lack of improvement in comprehension level.

As far as the **comprehension effort** is concerned, in general there is no improvement when screen mockups are available. A consistent difference was found for EasyCoin only. In this case familiarity with the application affected positively the benefits achieved. The lack of effort reduction for AMICO could be explained by a less extensive use of screen mockups, in particular for the development *Question category*, which caused difficulties in comprehending and thus more time spent (or wasted).

4.1 Threats to validity

In this section we discuss the threats to validity [7] that could affect our results: *construct*, *internal*, *conclusion*, and *external* validity threats.

Construct validity threats concern the relationship between theory and observation. This threat is related to how comprehension comprehension level and effort were measured. Comprehension was measured using questionnaires and answers were evaluated using an information retrieval approach. The same approach was used also in other studies (e.g., [5, 6, 12]). Effort was measured by means of proper time sheets and validated qualitatively by researchers who were present during the experiment.

Internal validity threats concerns factors that may affect an independent variable. They can be due to learning and fatigue effects experienced by subjects between *Tasks*. The learning effect is mitigated by the chosen experiment design: subjects worked, over the two *Tasks*, on different systems with different treatments. Moreover, ANOVA analyses, used to study the influence of the *Task* factor, confirm that the learning effect is not significant. Instead, the fatigue effect is mitigated by the mandatory break that we imposed between the two tasks. Moreover, to avoid apprehension, students were not evaluated on their performance. And, they were not aware of the experimental hypotheses.

Conclusion validity concerns the relationship between the treatment and the outcome. In our study proper statistical tests were used. In particular, non-parametric tests (Mann-Whitney test for unpaired analyses, the Wilcoxon test for paired analyses) were performed to statistically reject the null hypotheses and two-way ANOVA was used to detect possible interactions between each co-factor and the main factor. Even if all the assumptions/conditions for using ANOVA were not valid, this test is quite robust and has been used extensively in the literature to conduct analysis similar to ours. Regarding the post-questionnaires (mainly intended to get qualitative insights) were designed using standard ways and scales [14].

External validity concerns the generalization of the findings. Threats belonging to this category are mainly related to the simple tasks used in the experiment (mainly for time reasons) and to the use of students as experimental subjects. Regarding the second point, we can say that the selected subjects represent a population of students specifically trained on software engineering tasks and in particular on requirements. This makes these students not so much inferior to professional young junior developers. Clearly, further studies with larger/real tasks and more experienced developers are needed to confirm or contradict the results.

5 Related work

Several methods and techniques for modeling requirements have been proposed in the past. However, only a few empirical studies have been proposed to assess them. For example, Anda *et al.* in [17] perform an exploratory study with undergraduate students to investigate three different sets of guidelines to construct and document Use Case models. The data analysis reveals that the guidelines based on templates are easier to understand. Furthermore, guidelines based on templates better supports students in constructing Use Case models and the quality of the produced documentation is higher. Zimmerman *et al.* in [18] present the results of an empirical study conducted to determine how some factors regarding a formal method (i.e., state-based requirements specification requirements language) affect the requirements readability of aerospace applications. The data analysis reveals that familiarity with state machines is a much more influential factor than is familiarity with the problem domain of the system under study.

Use Cases are widely employed to specify functional requirements in object oriented software systems. Accordingly, the need of writing guidelines to specify clear and accurate Use Case descriptions is evident. Phalp *et al.* in [19] describe an empirical work to assess the utility of two requirements writing guidelines, namely CREWS and CP. In particular, the Use Cases are assessed using a checklist previously presented by the authors. The study reveals that these techniques are comparable in terms of their ability to produce clear and accurate (comprehensible) descriptions. In [20] the design of an empirical experiment is presented. The authors define and sett up an experiment to detect differences in the understanding of Use Cases and to investigate the effect of using different notations in the comprehension of software requirements.

As mentioned above, Use Cases may be used together with UML Use Case diagrams. This aspect is empirically investigated in [21]. This study shows that subjects that employed both a Use Case diagram and Use Cases achieved a significantly higher level of understanding.

Controlled experiments have been also conducted to assess the effectiveness of UML to model requirements of object oriented software systems. For example, Gravino *et al.* in [6] presents a controlled experiment to assess whether the comprehension of software requirements is influenced by the use of dynamic modeling abstracted adopting UML sequence diagrams. The study is conducted using as subjects students of the Bachelor program in Computer Science of the University of Basilicata in Italy. That data analysis reveals that there is not a significant difference in the comprehension of system requirements when the subjects are provided or not with dynamic models. However, the results of an external replication [22] conducted with students of the Master program in Computer Science of the Universidad Politcnica de Valencia in Spain contradict the findings in [6].

In the requirement engineering, the business processes that should be supported by the system under development can be specified using different visual formalisms, such as UML Activity Diagrams, BPMN, or Event-driven Process Chains. In [23] two controlled experiments to compare UML Activity Diagrams and Event-driven Process Chains are reported. The main objective of that comparison concerns the model understandability from the requirements engineers' and customers' perspective. In the case of requirements engineers, better performances have been obtained when they use UML Activity Diagrams. On the other hand, the customers do not get significant differences in terms of business process understandability, when they use both the business process visual notations.

6 Conclusions and future work

In this paper, we have presented the results of a controlled experiment aimed at comparing the effectiveness of Use Cases with and without the support of screen mockups in the comprehension of software requirements. The data analysis indicates that benefits are obtained adopting screen mockups with no substantial increase/decrease of effort.

Future empirical studies will aim at investigating whether benefits of screen mockups will keep also for other categories of subjects (e.g., Master students, PHD students and professional developers). In addition, it would be worth analyzing whether the additional effort and cost due to the development of screen mockups, will be paid back by a improved comprehension of the Use case descriptions. In fact, from a manager point of view, the adoption of screen mockups, as prototyping tool in the software development life-cycle, must be considered keeping into account the costs it will introduce.

Our decision to investigate the effect of exploiting Use Cases and screen mockups through a controlled experiment is due to the fact that a number of confounding and uncontrollable factors could be present in an industrial context, thus biasing the results. In fact, in a real project setting, it may be quite impossible to control factors such as learning/fatigue effects and to select specific comprehension tasks [24].

7 Acknowledgments

We would like to thank the students that participated to the experiment.

References

[1] R. Young, *Effective Requirements Practice.* Boston, MA: Addison-Wesley, 2001.

[2] B. Meyer, "On formalism in specification," *IEEE Software*, January 1985.

[3] R. Mayer, *Multimedia learning.* New York: Cambridge University Press, 2001.

[4] E. Astesiano, M. Cerioli, G. Reggio, and F. Ricca, "Phased highly-interactive approach to teaching uml-based software development," in *Educators Symposium at ACM/IEEE 10th International Conference on Model Driven Engineering Languages and Systems*, September 30 2007.

[5] L. Kuzniarz, M. Staron, and C. Wohlin, "An empirical study on using stereotypes to improve understanding of UML models," in *Proceedings of the International Workshop on Program Comprehension.* IEEE CS, 2004, pp. 14–23.

[6] C. Gravino, G. Scanniello, and G. Tortora, "An empirical investigation on dynamic modeling in requirements engineering," in *MoDELS '08: Proceedings of the 11th international conference on Model Driven Engineering Languages and Systems.* Berlin, Heidelberg: Springer-Verlag, 2008, pp. 615–629.

[7] C. Wohlin, P. Runeson, M. Höst, M. Ohlsson, B. Regnell, and A. Wesslén, *Experimentation in Software Engineering - An Introduction.* Kluwer Academic Publishers, 2000.

[8] B. Bruegge and A. H. Dutoit, *Object-Oriented Software Engineering: Conquering complex and changing systems.* Prentice-Hall, 2000.

[9] OMG, "Unified modeling language (UML) specification, version 1.4," Object Management Group, Tech. Rep., September 2001.

[10] F. Ricca, M. Torchiano, M. Di Penta, M. Ceccato, P. Tonella, and C. A. Visaggio, "Are fit tables really talking? a series of experiments to understand whether fit tables are useful during evolution tasks," in *IEEE International Conference on Software Engineering (ICSE 2008)*, October 2008, pp. 361–370.

[11] J. Aranda, N. Ernst, J. Horkoff, and S. Easterbrook, "A framework for empirical evaluation of model comprehensibility," in *Modeling in Software Engineering (MISE), 2007. ICSE Workshop 2007.* Minneapolis, MN, USA: IEEE CS, 2007, pp. 7–13.

[12] F. Ricca, M. Di Penta, M. Torchiano, P. Tonella, and M. Ceccato, "The role of experience and ability in comprehension tasks supported by UML stereotypes," in *Proceedings of the International Conference on Software Engineering (ICSE).* IEEE Computer Society, May 2007, pp. 375–384.

[13] W. B. Frakes and R. Baeza-Yates, *Information Retrieval: Data Structures and Algorithms.* Englewood Cliffs, NJ: Prentice-Hall, 1992.

[14] A. N. Oppenheim, *Questionnaire Design, Interviewing and Attitude Measurement.* London: Pinter, 1992.

[15] J. Cohen, *Statistical power analysis for the behavioral sciences (2nd ed.).* Hillsdale, NJ: Lawrence Earlbaum Associates, 1988.

[16] A. Agresti, *An Introduction to Categorical Data Analysis.* Wiley-Interscience, 2007.

[17] B. Anda, D. I. K. Sjøberg, and M. Jørgensen, "Quality and understandability of use case models," in *ECOOP '01: Proceedings of the 15th European Conference on Object-Oriented Programming.* London, UK: Springer-Verlag, 2001, pp. 402–428.

[18] M. K. Zimmerman, K. Lundqvist, and N. Leveson, "Investigating the readability of state-based formal requirements specification languages," in *Proceedings of the 24th International Conference on Software Engineering.* New York, NY, USA: ACM, 2002, pp. 33–43.

[19] K. T. Phalp, J. Vincent, and K. Cox, "Improving the quality of use case descriptions: empirical assessment of writing guidelines," *Software Quality Control*, vol. 15, no. 4, pp. 383–399, 2007.

[20] B. Anda and M. Jrgensen, "Understanding use case models," in *13 th European Conference on Object-Oriented Programming (ECOOP2001.* Springer Verlag, 2000, pp. 402–428.

[21] G. Andrew and P. Drew, "Use case diagrams in support of use case modeling: Deriving understanding from the picture," *Journal of Database Management*, vol. 20, no. 1, 2009.

[22] S. Abrahao, E. Insfran, C. Gravino, and G. Scanniello, "On the effectiveness of dynamic modeling in uml: Results from an external replication," in *Empirical Software Engineering and Measurement, 2009. ESEM 2009. 3rd International Symposium on*, Oct. 2009, pp. 468–472.

[23] A. Gross and J. Doerr, "Epc vs. uml activity diagram - two experiments examining their usefulness for requirements engineering," *IEEE International Conference on Requirements Engineering,*, vol. 0, pp. 47–56, 2009.

[24] E. Arisholm, L. C. Briand, S. E. Hove, and Y. Labiche, "The impact of uml documentation on software maintenance: An experimental evaluation," *IEEE Transactions on Software Engineering*, vol. 32, pp. 365–381, 2006.