Abrahao Gonzales, SM.; Gravino, .C.; Insfrán Pelozo, CE.; Scaniello, .G.; Tortora, .G. (2013). Assessing the effectiveness of sequence diagrams in the comprehension of functional requirements: results from a family of five experiments. IEEE Transactions on Software Engineering. 39(3):327-342. doi:10.1109/TSE.2012.27.

Assessing the Effectiveness of Dynamic Modeling in the Comprehension of Software Requirements: Results from a Family of Five Experiments

Silvia Abrahão[1], Carmine Gravino[2], Emilio Insfran[1], Giuseppe Scanniello[3], *Member, IEEE*,

Genoveffa Tortora[2], *Senior Member, IEEE*,

[1]*Department of Computer Science, Universidad Politécnica de Valencia*

c/ Camino de Vera, s/n, 46022, SPAIN

{sabrahao, einsfran}@dsic.upv.es

[2]*Dipartimento di Matematica e Informatica, University of Salerno*

Via Ponte Don Melillo, 84084, Fisciano (SA), ITALY

{gravino, tortora}@unisa.it

[3]*Dipartimento di Matematica e Informatica, University of Basilicata*

Viale Dell'Ateneo, Macchia Romana, 85100, Potenza, ITALY

giuseppe.scanniello@unibas.it

Abstract

Modeling is a fundamental activity within the requirements engineering process and concerns the construction of abstract descriptions of software requirements that are amenable to interpretation and validation. The choice of a modeling technique is a critical issue whenever it is necessary to discuss the interpretation and validation of software requirements. This is particularly true in the case of stakeholders with divergent goals and different backgrounds and experience. This paper presents the results of a family of experiments conducted with students and professionals to investigate whether the comprehension of software requirements is influenced by the use of dynamic models. The family contains five experiments performed in different locations and with 112 subjects of different abilities and levels of experience with UML. The results show that dynamic models improve the comprehension of software requirements in the case of high ability and more experienced subjects.

Keywords – *Documentation, Software Engineering, Requirements Specifications.*

1. Introduction

Software requirements are validated to establish whether they provide an accurate account of stakeholders' needs [32]. Adapted from the problem of validating scientific knowledge, requirements validation is the task of making sufficient empirical observations to verify that a real world problem is properly captured [41]. In order to assess whether a requirement has been properly captured, the associated models should be interpreted and understood by all the stakeholders [50]. It may occur that stakeholders do not correctly interpret and comprehend the software requirements models [1], thus increasing the cost needed to fix these misinterpretations later in the development process [11], [49].

Several methods with which to model software requirements have been proposed in the past, and of these, behavioral modeling is one of the most broadly employed approaches [29], [41], [63]. Behavioral modeling involves the modeling of the dynamic and/or functional behavior of users and software systems [15], [50] and produces the functional, analysis object (or conceptual), and dynamic models.

In this paper, we present the results of a family of five controlled experiments[1] carried out to investigate whether the use of dynamic models improves the comprehension of software requirements. The original experiment was conducted with a group of second-year Computer Science undergraduate students from the University of Basilicata in Italy [36]. The data analysis shows that there was no significant effect of dynamic models on the comprehension of software requirements, although the subjects considered these models to be useful.

In order to further investigate the results of the original experiment, we carried out four replications with subjects of different backgrounds and experience in modeling with UML [52]. These replications were conducted with Computer Science Master's degree students from the University of Salerno in Italy, Software Engineering Master's degree students from the Universidad Politécnica de Valencia (UPV) in Spain, PhD students at UPV and Spanish software professionals. In order to increase our confidence in the results attained, the latter two replications were based on more complex software system requirements from different domains. The results of the replications show that dynamic models significantly improve the comprehension of software requirements, whatever the complexity of the requirements is. Furthermore, the data analysis on all the experiments suggests that subjects with a high ability and more experience benefit more from the use of dynamic models than low ability and less experienced subjects.

[1] A controlled experiment is normally conducted in a laboratory environment, thus providing a high level of control and identifying cause-effect relationships. Controlled experiments are generally used to show with statistical significance that a particular process, method, or tool is better than others.

The paper is structured as follows: Section 2 presents the family of experiments. Section 3 provides details on the design of the individual experiments, including the definition of the experiments, context selection, hypothesis formulation and instrumentation. Section 4 reports on the results of each experiment, together with their interpretations. Section 5 presents and discusses the results of the family of experiments. Section 6 discusses the threats to validity, while Section 7 reports on related works. The paper concludes with final remarks and future research directions.

2. The Family of Experiments

An increasing understanding exists within the Software Engineering community that empirical studies are needed to create, improve, or assess processes, methods, and tools for software development [7], [8], [31] and maintenance [21], [30]. An empirical study is generally an act or operation by which to discover something that is unknown, or to test hypotheses [6]. Research strategies include controlled experiments [12], [13], [24], qualitative studies [39], [59], surveys [28], [46], and archival analyses [25], [37], [44]. In order to achieve greater validity of the results, replications are necessary [48]. Unfortunately, in Software Engineering an excessive amount of empirical studies tend to be isolated and are not replicated. For example, in [60], Sjøberg *et al.* report that 20 out of 113 controlled experiments are replications, and of these, 15 are differentiated replications[2].

The concept of replication is extended to that of the "family of experiments" reported by Basili *et al.* [9]. A family is composed of multiple similar experiments that pursue the same goal in order to build the knowledge that is needed to extract significant conclusions that can be applied in practice.

In this section, we present the family of experiments that we have performed. The method adopted is an extension of the five-steps proposed by Ciolkowski *et al.* [20], in which the fifth step, "Family data analysis", is replaced with "Family data analysis and meta-analysis".

Step 1. Experiment preparation. The goal of our study is to analyze the use of dynamic models with the purpose of assessing whether they improve the comprehension of software requirements for different categories of users in terms of experience and ability. The study is conducted from two perspectives: that of the researcher point of view

[2] These replications introduce variations in essential aspects of the experimental conditions. Differentiated replications can be conducted to identify potentially important environmental factors that affect the experimental results. One prominent variation of the environmental factors concerns the executions of replications with different kinds of subjects, thus enabling us to understand whether the conclusions from the original experimental context can be generalized to the new context.

in order to investigate the support provided by dynamic models in the comprehension of software requirements, and that of the project manager point of view in order to evaluate the possibility of adopting these requirements to enhance software requirements abstractions. We have therefore defined and investigated the following research question:

- Do dynamic models improve the comprehension of software requirements for subjects with different levels of experience and ability?

Step 2. Context definition. The groups of subjects with different level of experience are as follows:

- Undergraduate students in Computer Science, who can be considered as the next generation of professional developers [45]. It has been shown that, under certain conditions, there is no great difference between such students and professionals [9], [40].
- Master's students in Computer Science, many of whom are, or have been, professionals in industry.
- PhD students in Computer Science, who have attended various Software Engineering courses.
- Software industry professionals, who design and develop software systems.

With regard to the subjects' ability, the students were classified according to the average grades attained in their academic degrees, i.e., for Italian subjects an average of below 24/30 is considered to be Low, otherwise High, while for the Spanish subjects the threshold was 9/10. Each subject's participation was voluntary. All the subjects were given one point in their final grades, regardless of their performances. A different classification was applied for the professionals (see Section 3.5).

Step 3. Material and Experimental Tasks. The material was composed of two experimental objects (containing a set of UML models with an attached comprehension questionnaire), training material, and a post-experiment survey. The experimental objects were the software requirements of an e-commerce system and other typical management information systems (e.g., course management). The comprehension questionnaire included multiple-choice questions in order to obtain a quantitative evaluation of requirements comprehension. The training materials included: *(i)* a set of instructional slides describing an introduction to dynamic modeling containing examples of UML models; *(ii)* comprehension tasks related to these models.

Similarly to [12], [13], [56], at the end of each controlled experiment, the subjects were asked to fill in the post-experiment survey questionnaire. The aim of the questionnaire was to gain sufficient insight through which to strengthen and explain the results; it included questions to assess the overall quality of the material provided, the

perceived usefulness of the dynamic models, and the clearness of the tasks and the goals of the experiment.

Step 4. Individual experiments. With regard to Figure 1, the second and third experiments (Italy 2 and Spain 1, respectively) were differentiated replications which aimed to replicate the original experiment (i.e., Italy 1) in different settings with different subjects. In particular, Italy 2 was an *internal* replication (the same experimenters as the original one), while Spain 1 was an *external* replication (different experimenters to verify whether the results are independent both of the experimenters and of the setting). In Italy 2 and Spain 1 the same experimental objects were employed as with Italy 1. The fourth and fifth experiments (Spain 2 and Spain 3, respectively) were differentiated replications whose goal was to repeat the first experiment with different subjects and materials (different experimental objects). These experiments were also conducted by an independent group of experimenters, and can thus be considered as *external* replications.

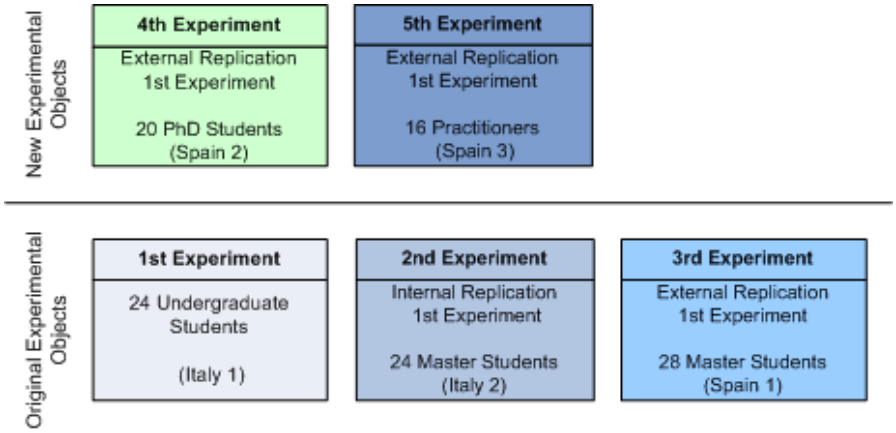


Figure 1. Experiments in the family

Step 5. Family data analysis and meta-analysis. The results of each experiment and of the family of experiments were collected and analyzed. We also performed meta-analysis, since the experimental conditions were the same or very similar within each experiment. This enabled us to extract more general conclusions with regard to the analysis of each individual experiment.

3. Design of Individual Experiments

In the following sub-sections we present the design of the experiments according to the guidelines for experimental software engineering proposed by Wohlin *et al.* [64] and Juristo and Moreno [42]. The experimental material and the raw data are available for download at www.scienzemfn.unisa.it/scanniello/RE_Exp1/.

3.1 The Original Experiment

The software requirements were abstracted by considering the method suggested in [15], in which a requirements analysis document includes a functional model, an analysis object (or conceptual) model, and a dynamic model. A functional model focuses on the software functionality, while an analysis object model focuses on the individual concepts of the problem domain that will be manipulated by the software system. Finally, a dynamic model concerns the software system behavior of the meaningful use cases presented in the functional model. In this study, use case diagrams were used to characterize a functional model, while class diagrams and sequence diagrams were employed to abstract analysis object models and system behavior, respectively.

3.1.1 Planning

Context. The experimental objects were selected within the requirements analysis specifications of the following two systems:

- **ECP** – An e-commerce platform from which CDs and books can be ordered via the Internet from an on-line catalogue.
- **E-Plat** – A software system for the management of courses, lecturers, and students of a given University.

We randomly selected the functional requirement "Processing an order" for the first system (from here on, ECP), while "Adding a course" was the requirement employed for the latter system (from here on, E-Plat). In order to assess the complexity of the functional requirements used and to identify possible mistakes in the associated models, a pilot experiment was carried out by a Master's student before the actual experiment. From the pilot, we deduced that the complexity of the two requirements could be considered comparable since the student involved spent almost the same time on accomplishing both the tasks using dynamic models (30 minutes on average).

The subjects that participated in this study were a group of 24 Computer Science undergraduate students from the University of Basilicata in Italy. The students were chosen for convenience (they were students enrolled on the "Software Engineering" course at the University of Basilicata in the first semester, from September 2007 to January 2008). One of the main topics of the course is the modeling of object-oriented systems using UML. The subjects also had experience in object-oriented programming and Web technology. During the course, the subjects were grouped in teams, and each team was involved in the modeling and development of a software system. Two young

researchers were responsible for coordinating the projects, which were based on requirements analysis, on the high-level design of the software system, and on an incremental development of the identified subsystems. The subjects' participation was voluntary. One point was added to each subject's final grade, regardless of their performances.

Hypotheses formulation. We have formulated the following one-tailed null hypothesis:

- H_{n0}: The use of dynamic models *does not significantly improve* the comprehension of software requirements.

In the case of it being possible to reject the null hypothesis with relatively high confidence, it is possible to formulate an alternative hypothesis that admits a positive effect:

- H_{a0}: The use of dynamic models *significantly improves* the comprehension of software requirements.

In the experiment, the *Control Group* is "requirements specifications *without* dynamic models", while the *Treatment Group* is "requirements specifications *with* dynamic models". The only independent variable (also named main factor) is, therefore, Method, which is a nominal variable that can assume two possible values: DM (Dynamic Model) and NO_DM (NO Dynamic Model).

In order to verify the null hypothesis, we considered a measure based on the subjects' comprehension of the software requirements. The comprehension questionnaire consisted of eight multiple-choice questions with one or more correct answers for each of a set of four answers (see Figure 2). The number of possible answers was the same for each question. The purpose of this questionnaire was to rapidly evaluate various comprehension aspects of the system requirements.

Q3. The end-user of the system can*:

☐ Add a delivery.

☐ Modify an order.

☐ Modify the data of an invoice.

☐ Create an order.

*Mark the right answer/s

Figure 2. A sample question from the comprehension questionnaire for ECP

The use of comprehension questionnaires enabled us to assess each subject's comprehension in terms of the measures: recall, precision, and F-measure [5]. Similarly to [3], [65], the recall and precision measures have been defined as follows:

$$recall_s = \frac{\sum_i \left| answers_{s,i} \cap correct_i \right|}{\sum_i \left| correct_i \right|} \quad (1)$$

$$precision_s = \frac{\sum_i \left| answers_{s,i} \cap correct_i \right|}{\sum_i \left| answer_{s,i} \right|} \quad (2)$$

where $answer_{s,i}$ is the set of responses that the subjects s provided for the question i. On the other hand, $correct_i$ is the set of correct responses expected for the question i.

Precision concerns the correctness of the responses provided by the subject s, while the completeness of the responses is measured by employing recall. In order to obtain a balance between correctness and completeness, we used the harmonic mean of precision and recall [5]:

$$F - measure_s = 2 * \frac{precision_s * recall_s}{precision_s + recall_s} \quad (3)$$

To test the previously presented null hypothesis, the following dependent variable was then considered:

– **Comprehension**: the F-measure of the precision and recall values of all the questions of the comprehension questionnaire.

Since we were also interested in assessing the effect of ability on the comprehension of software requirements (see Section 2), the Ability nominal variable was additionally considered and its effect was also analyzed. The values that the Ability co-factor can assume are Low and High.

Experiment Design. We adopted a counterbalanced design by dividing the subjects into four groups, namely A, B, C, D (see

Table 1). The subjects in each group were asked to perform the tasks without interacting with each other. The two tasks were performed in sequence. The chosen design mitigates possible learning effects and permits the study of the effect of other factors, which are discussed as follows.

Table 1. Design of the controlled experiments

	Groups			
	A	**B**	**C**	**D**
Task1	ECP, DM	ECP, NO_DM	E-Plat, DM	E-Plat, NO_DM
Task2	E-Plat, NO_DM	E-Plat, DM	ECP, NO_DM	ECP, DM

Other Factors to be Controlled. Although Method is the only factor of interest, other factors (also named co-factors) may influence the results. In particular, different extraneous factors may have an undesirable effect on the subjects' comprehension of the requirements, and this effect may be confused with the Method effect:

- ***Object.*** The complexity of the requirements and the subjects' familiarity with the application domain of the systems may affect the subjects' comprehension of the requirements.
- ***Order of Method.*** The order in which the subjects perform the tasks may produce learning effects, thus biasing the results.

3.1.2 Operation

Preparation and Execution. The experiment took place in a single room. We gave the subjects all the materials described in Section 2. All the subjects attended an introductory lesson in which detailed instructions on the tasks to be performed were presented, although only the goal of the experiment was highlighted, and details on the experimental hypothesis were not provided. The experiment's execution was controlled, and no interaction between subjects was allowed. Furthermore, no time limit was imposed in order to avoid a possible ceiling effect [61]. The subjects were told that their answers would be treated anonymously and were also informed that their grade on the course would not be affected by their performance in the experiment. The subjects were also aware of the pedagogical purpose of the exercises.

After the comprehension tasks, the subjects were asked to fill in the post-experiment survey questionnaire shown in Table 2.

Analysis procedure. We used statistic tests to analyze the collected data. In all the tests we decided, as usual, to accept a probability of 5% of committing a Type-I-Error [64], i.e., of rejecting the null hypothesis when it is actually true.

The defined null hypothesis was tested by using the non-parametric Wilcoxon test [23] since a statistical

difference between two dependent groups (paired analysis) was under investigation. In contrast, in order to analyze the effect of Ability and Object, we performed an unpaired analysis by applying the Mann–Whitney test [23]. These tests have been chosen since they are very robust and sensitive, and have been used in experiments similar to ours in the past, e.g., [12], [56].

Table 2. The post-experiment survey questionnaire

Id	Question	Possible Answers
Q1	I had enough time to perform the tasks.	(**1-5**)
Q2	The task objectives were perfectly clear to me.	(**1-5**)
Q3	The tasks I performed were perfectly clear to me.	(**1-5**)
Q4	Judge the difficulty of the task on the e-commerce system.	(**A-E**)
Q5	Judge the difficulty of the task on the system for the management of courses, lecturers, and students.	(**A-E**)
Q6	Assess your experience level in analysis object modeling and the UML class diagrams.	(**A-E**)
Q7	Assess your experience level in dynamic modeling and the UML sequence diagrams.	(**A-E**)
Q8	What is the most useful method to comprehend and interpret software requirements?	**DM**, **NO_DM**, **Neutral**
Q9	Which models did you perceive as the most useful to comprehend and interpret software requirements?	**Functional**, **Analysis Object**, **Dynamic**
1 = Strongly agree, **2** = Agree, **3** = Neutral, **4** = Disagree, **5** = Strongly disagree		
A = Very high, **B** = High, **C** = Medium, **D** = Low, **E** = Very low		

These tests allow the presence of a significant difference between dependent or independent groups to be verified, but they do not provide any information about this difference. We therefore used the Cohen's "d" [19] to obtain the standardized difference between two groups that can be considered *negligible* for $|d| < 0.2$, *small* for $0.2 \leq |d| < 0.5$, *medium* for $0.5 \leq |d| < 0.8$, and *large* for $|d| \geq 0.8$. Similarly to [56], in the context of paired analyses, we used the difference between the means of the distributions divided by the standard deviation of the (paired) differences between the samples. In the context of unpaired analyses, we used the difference between the means of the distributions divided by the pooled standard deviation.

The counterbalanced experimental design also allows the interaction of the co-factors with the main factor to be analyzed. We use interaction plots to study the interaction between Ability and Method and between Object and Method. Interaction plots [27] are simple line graphs in which the means of the dependent variables for each level of one factor are plotted over all the levels of the second factor. The resulting lines are parallel when there is no interaction and nonparallel when interaction is present.

In order to test the effect of Order of Method (the measurement differences between the subjects' observations when using and not using dynamic models), we used the method suggested by Briand *et al.* in [12]. In particular, let: *Diff(NO_DM)* be the differences in comprehension of those subjects who performed the tasks using NO_DM first and then DM; and *Diff(DM)* be the differences in comprehension of those subjects who performed the tasks using DM first and then NO_DM. We used the one-tailed t-test to verify whether Diff (NO_DM) is greater than Diff (DM). When applying this test, the normal distribution assumption should be verified using the Shapiro-Wilk test [58]. In the case of this assumption not being verified, we planned to use the non-parametric Mann–Whitney test. In our case we expected that Diff (NO_DM) would be larger than Diff (DM). The rationale behind this concern relies on the fact that the capability of using dynamic models may improve during the tasks. As a result, we should verify the null hypothesis H_{0d}: Diff (NO_DM) = Diff (DM). On the other hand, the alternative hypothesis is H_{ad}: Diff (NO_DM) > Diff (DM).

3.2 Italy 2

Italy 2 was conducted by the same experimenters (internal replication). The same design and conditions were employed, while the kind of subjects was changed. This replication was necessary to increase our confidence in the conclusion validity of the original experiment.

The subjects were 24 students from a Computer Science Master's degree program at the University of Salerno in Italy who were enrolled on the Software Advanced Software Engineering course. The experiment was conducted as part of a series of laboratory exercises carried out within this course. This course was held in the second semester (from March 2008 to June 2008). All the students had a comparable level of background since they were all graduate students of the same Bachelor's degree program.

The subjects were required to have a reasonable level of technical maturity and knowledge of UML-based, object-oriented software development, and software project management. In particular, the subjects were familiar with concepts of the requirements engineering process and had experience in supervising teams of developers.

3.3 Spain 1

The same experimental objects as those used in Italy 1 and Italy 2 were employed. The experiment was an external replication since it was performed by an independent group of experimenters at the UPV in Spain.

The subjects were 33 students enrolled on a Master's degree program in Software Engineering, Formal Methods

and Information Systems at the UPV. They were asked to perform the experiment as a part of a series of optional laboratory exercises conducted within the Software Engineering with Models course. This course was held in the first semester, from September 2008 to January 2009, and was selected because it is a specialized teaching unit in which students learn advanced techniques regarding conceptual modeling, UML, and model-driven engineering. Most of the students on this course had completed two Software Engineering courses in which they had acquired training in UML modeling.

3.4 Spain 2

This experiment varied the manner in which the other experiments were run with the intention of increasing confidence in the experimental results by testing the same hypothesis as before, but altering the details of the experiment to increase external validity. In particular, different experimental objects were employed and the study participants were PhD students. The systems from which we selected these objects were:

- **M-Shop** – A software system for managing the sales in a music shop.
- **Theater** – A software system for managing a theater's ticket reservations.

We randomly selected the functional requirement "Search Album by Singer" for the first system (from here on, M-Shop). For the second system (from here on, Theater) we selected "Buy Theater Ticket". As with Italy 1, we conducted a pilot experiment. This was carried out with a more experienced subject, namely a research fellow from the University of Salerno. The results indicated that 45 minutes were, on average, sufficient to accomplish each task using dynamic models. The time needed to accomplish the tasks, the size of the models within the experimental objects, and the experience of the pilot subjects enabled us to deduce that the M-Shop and Theater objects were more complex than those used in the previous experiments (i.e., Italy 1, Italy 2, and Spain 1).

The subjects were 20 PhD students enrolled in the Software Engineering, Formal Methods and Information Systems PhD program at the UPV in Spain. The experiment was conducted in the Software Engineering with Models course. The experiment was conducted as part of a series of laboratory exercises carried out within this course. This course was held in the first semester (from September 2008 to February 2009).

3.5 Spain 3

Spain 3 also varied the manner in which the original experiment was run. The same design and materials as those used in Spain 2 were employed. The subjects involved were 16 professionals from Spanish software development

companies. The experiment was performed in the context of the Modeling with UML course. The main objective of this course was to teach its participants the principles of object-oriented analysis and design. The subjects in this experiment had various backgrounds. They primarily worked as software analysts and programmers in industry and had various levels of experience in modeling with UML (from 3 months to 4 years). In order to group the professionals as Low and High ability subjects we considered their grades in the Modeling with UML exam. Those subjects that had attained a grade of less than 9/10 were qualified as Low, otherwise High. This enabled us to classify subjects as being of high and low ability in a repeatable and just manner. The professionals' participation was voluntary.

4. Results and Discussion

In this section the results of the single experiments is discussed by analyzing: *(i)* the influence of the method; *(ii)* the Influence of Object, Ability, and Order of Method; *(iii)* the results of the post-experiment survey questionnaire.

4.1 Influence of Method

Table 3 shows descriptive statistics for Comprehension. The descriptive statistics show that the mean comprehension scores obtained for the subjects when using dynamic models were superior to those obtained when not using them. Upon analyzing the comprehension mean scores grouped by Object, we can observe that the subjects from Spain 1 were less effective than the other subjects when answering the ECP questionnaire. One possible reason for this might be that the subjects perceived ECP to be more complex than E-Plat. This must be analyzed in greater detail in the subsequent steps of the data analysis.

The results of the Wilcoxon test performed to study the effect of Method on the software requirements comprehension are summarized in Table 4. The test revealed that the null hypothesis H_{n0} cannot be rejected (p-value = 0.260) for Italy 1 with a negligible effect size. This table also shows the number of subjects who benefitted from DM in the comprehension tasks, those who achieved the same results from both DM and NO_DM, and those who obtained better comprehension scores with NO_DM. For example, 11 out of 24 Italy 1 subjects benefitted from the use of dynamic models, while 10 subjects did not benefit from them. The remaining 3 subjects obtained almost the same comprehension level when performing the tasks regardless of whether or not they used dynamic models.

Table 3. Descriptive Statistics for Comprehension

Data set	Object	DM			NO_DM		
		Med.	Mean	Std. Dev.	Med.	Mean	Std. Dev.
Italy 1	All	0.625	0.633	0.198	0.630	0.612	0.159
	ECP	0.730	0.638	0.205	0.690	0.647	0.142
	E-Plat	0.595	0.629	0.199	0.620	0.578	0.173
Italy 2	All	0.685	0.673	0.115	0.565	0.526	0.089
	ECP	0.670	0.649	0.097	0.595	0.564	0.071
	E-Plat	0.720	0.697	0.130	0.500	0.488	0.092
Spain 1	All	0.458	0.423	0.172	0.353	0.356	0.111
	ECP	0.235	0.286	0.110	0.353	0.307	0.117
	E-Plat	0.556	0.561	0.092	0.444	0.405	0.083
Spain 2	All	0.727	0.727	0.088	0.591	0.618	0.166
	M-Shop	0.727	0.709	0.103	0.546	0.519	0.123
	Theater	0.727	0.746	0.072	0.636	0.655	0.200
Spain 3	All	0.636	0.631	0.122	0.546	0.534	0.119
	M-Shop	0.636	0.602	0.047	0.500	0.523	0.106
	Theater	0.682	0.659	0.167	0.546	0.546	0.137

The null hypothesis H_{n0} can be rejected for all the replications with an effect size ranging from small (i.e., almost medium) to large. This means that the use of dynamic models significantly improves the comprehension of software requirements. For example, for the Spain 2 data set, this analysis revealed that 12 out of 20 subjects benefitted from dynamic models, while 5 subjects did not benefit from them. The remaining subjects obtained almost the same comprehension level with or without a dynamic model. These results diverge from those which were found in the original experiment, thus providing support for the hypothesis. One possible reason for this difference might be that the subjects in these experiments were more experienced with UML than the subjects of the original experiment.

Table 4. Wilcoxon test results for the family of experiments

Data Set	#obs	# of DM > NO_DM	# of DM < NO_DM	# of DM = NO_DM	Influence of Method (p-value)	Effect Size (Cohen d)
Italy 1	48	11/24	10/24	3/24	No (0.260)	Negligible (0.11)
Italy 2	48	20/24	3/24	1/24	**Yes (1.2e-4)**	Large (0.99)
Spain 1	56	16/28	6/28	6/28	**Yes (0.027)**	Small (0.43)
Spain 2	40	12/20	5/20	3/20	**Yes (0.017)**	Medium (0.55)
Spain 3	32	10/16	2/16	4/16	**Yes (0.010)**	Medium (0.64)

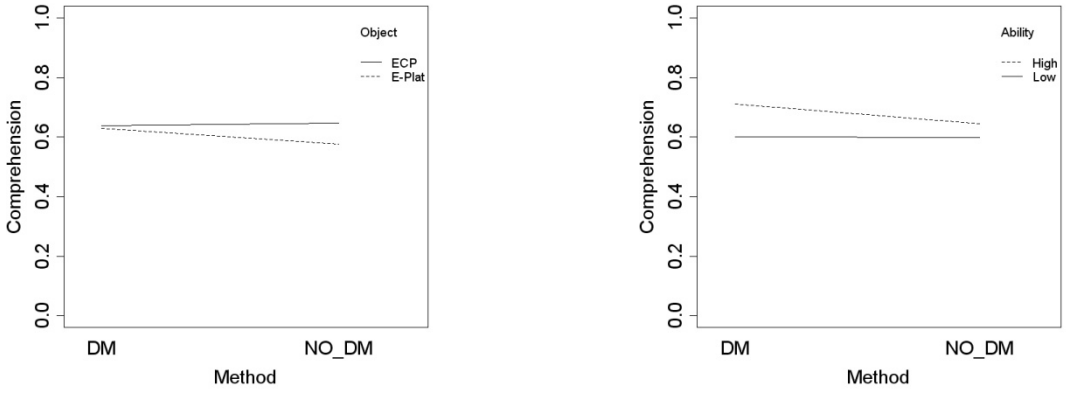
4.2 Influence of Object, Ability, and Order of Method

The results of the Mann-Whitney test when applied to the observations grouped by Method are shown in Table 5.

Table 5. Influence of Ability and Object on the comprehension of software requirements. The effect size is only shown when a statistical significant difference is present.

Experiment	Co-factor	Influence on Comprehension (p-value)	
		DM	**NO_DM**
Italy 1	Object	No (0.908)	No (0.213)
	Ability	No (0.203)	No (0.610)
Italy 2	Object	No (0.455)	**Yes (0.034)** Large (0.92)
	Ability	No (0.264)	No (0.726)
Spain 1	Object	**Yes (2.45e-5)** Large (2.71)	No (0.105)
	Ability	No (0.252)	**Yes (0.041)** Large (0.84)
Spain 2	Object	No (0.321)	No (0.263)
	Ability	No (0.640)	No (0.440)
Spain 3	Object	No (0.479)	No (0.626)
	Ability	No (1.00)	No (0.502)

Italy 1 - The results of the original experiment showed that there was no significant effect of Object on the comprehension of software requirements. Moreover, Ability did not influence the comprehension of software requirements. In addition, the plots in Figure 3 show that no interaction exists between Method and Ability or between Method and Object.



(a) **(b)**

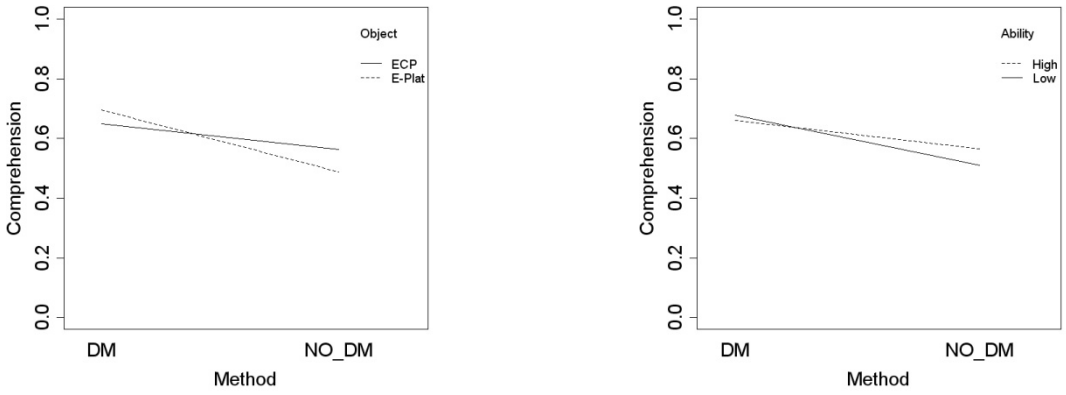
Figure 3. Interaction between Method and Object and between Method and Ability (Italy 1)

In order to verify the influence of Order of Method, we first verified the normality assumption by using the Shapiro-Wilk test. For each experiment, the test revealed that the distributions of Diff (NO_DM) and Diff (DM) were normal. Therefore, in order to test the null hypothesis H_{0d} we applied the t-test, which revealed that this hypothesis could not be rejected in the original experiment (p-value = 0.730). The same conclusion was proved to

hold in all the replications. This means that the subjects of all the experiments did not have a significantly greater Comprehension score in the second task.

Italy 2 - Table 5 shows a significant effect of Object on Comprehension when dynamic models were not used (p-value = 0.034). In particular, the subjects achieved a better comprehension level for ECP, with a large effect size (i.e., 0.92). The plot in Figure 4.a shows an interaction between Method and Object. This plot also indicates that ECP and E-Plat dynamic models better support the subjects in their accomplishment of the experimental tasks. Indeed, a better comprehension level was obtained for E-Plat.

The results of the Mann-Whitney test revealed that there was no significant effect of Ability on the comprehension of software requirements. However, the interaction plots in Figure 4.b suggest that there was some interaction between Method and Ability. Although the low ability subjects achieved better comprehension scores than the high ability subjects when using dynamic models, this difference can be considered as meaningless (the difference is less than 0.048). This result, and the fact that the high ability subjects achieved a better comprehension level when they did not use dynamic models, suggest that these models appear to fill the gap between the low and high ability subjects. It is also possible to note that both high and low ability subjects benefitted from dynamic models.



(a) **(b)**

Figure 4. Interaction between Method and Object & Method and Ability (Italy 2)

Spain 1 – The Mann-Whitney test showed a significant effect of Object on Comprehension when using dynamic models (p-value = 2.45e-5) with a very large effect size (i.e., 2.71) in favor of E-Plat. The effect of Object was not statistically significant when dynamic models were not used (p-value = 0.105). The plot in Figure 5.a indicates no interaction between Method and Object. We can also observe that the dynamic models do not improve the

comprehension of ECP. Indeed, the subjects achieved almost the same comprehension level whether using dynamic models or not (see Table 3 and Figure 5.a). This result, together with the fact that a better comprehension level was achieved for E-Plat by using dynamic models, could indicate that the subjects were more familiar with ECP, thus reducing the possible positive effect of models on the comprehension of software requirements.

The statistical test (p-value = 0.041) also showed that Ability affected the requirements comprehension when dynamic models were not used. In particular, low ability subjects achieved better comprehension levels with a large effect size (i.e., 0.84). In addition, the interaction plot in Figure 5.b indicates that an interaction between Method and Ability is present. However, no interaction was shown between Method and Object. Low ability subjects obtained slightly better results than high ability subjects when performing comprehension tasks without a dynamic model. A possible explanation for this might be that high ability subjects may make several interpretations of the requirements specifications that do not include a dynamic model. The comprehension level of low ability subjects is almost the same whether dynamic models are used or not. However, high ability subjects achieved better results than low ability subjects when using the dynamic model. It would thus appear that high ability subjects benefit from dynamic models more than low ability subjects. One possible reason for this might be that the use of dynamic models constrained possible interpretations of the software requirements.

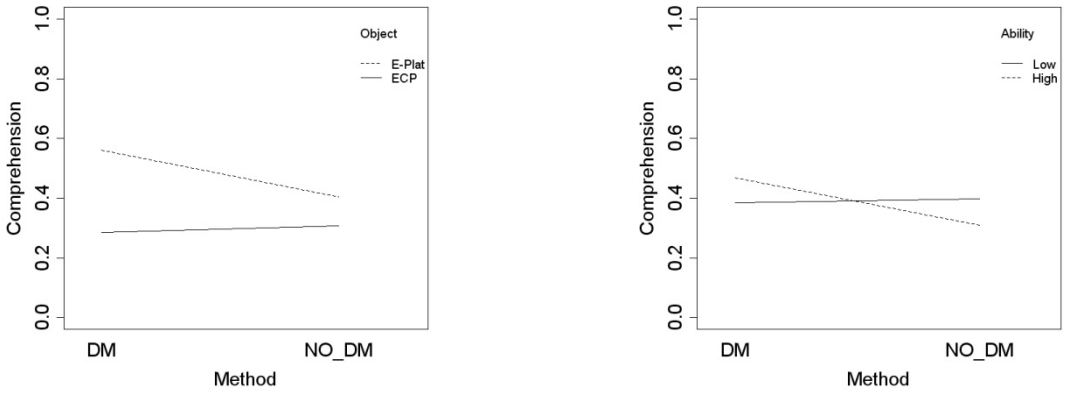


Figure 5. Interaction between Method and Object & Method and Ability (Spain 1)

Spain 2 - The results of data analysis revealed that there was no significant effect of Object and Ability on Comprehension. Furthermore, the interaction plots in Figure 6 indicate that there is no interaction between Method and Object, while there is an interaction between Method and Ability. In particular, low ability subjects obtained better results than high ability subjects when performing tasks without a dynamic model, while high ability subjects

achieved better results than low ability subjects when using dynamic models. These results are similar to those obtained with the final year undergraduate students from the same university (Spain 1), thus suggesting that high ability subjects benefit from dynamic models more than low ability subjects.

Spain 3 - The results of the Mann–Whitney test revealed that there was no significant effect of Object on comprehension. The results of this test showed that Ability did not influence the results attained. The interaction plots in Figure 7 indicate that there is no interaction between Method and Object, while there is an interaction between Method and Ability. In particular, low ability subjects obtained better results than high ability subjects when performing comprehension tasks without a dynamic model, while high ability subjects achieved better results than low ability subjects when using the dynamic model. These results are in line with those obtained in Spain 1 and Spain 2, since high ability subjects benefit from dynamic models more than low ability subjects.

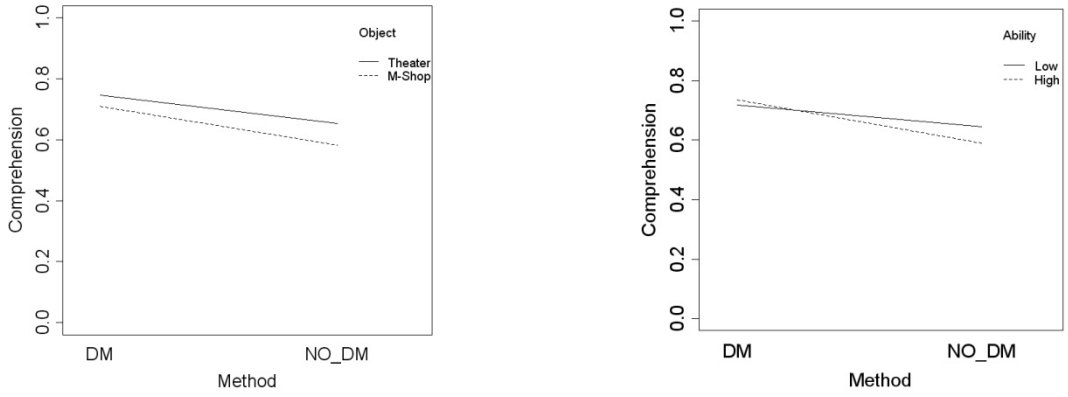


Figure 6. Interaction between Method and Object & Method and Ability (Spain 2)

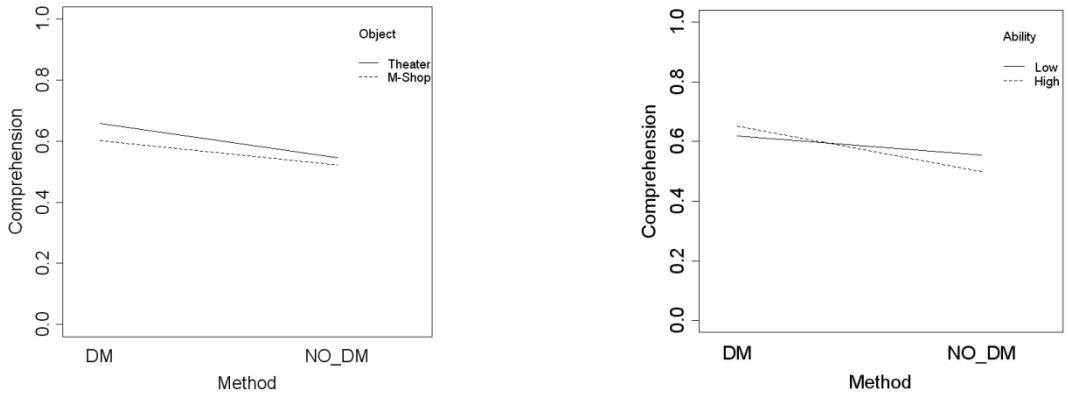


Figure 7. Interaction between Method and Object & Method and Ability (Spain 3)

4.3 Survey questionnaire results

The responses provided to the questions ranging from Q1 to Q7 are visually summarized in the boxplots in Figure 8. This figure shows the responses for the first three experiments (i.e., Italy 1, Italy 2, and Spain 1). We do not show the boxes for questions Q8 and Q9 owing to the different scales adopted. The responses to these questions are, however, analyzed and discussed. Our rationale for analyzing the responses to the Italy 1, Italy 2, and Spain 1 post experiment survey questionnaires together is that they are based on the same experimental objects. Accordingly, the results of the Spain 2 and Spain 3 post experiment survey questionnaires are also presented together (see Figure 9).

The analysis of the collected answers for question Q1 showed that the time needed to carry out the experiments was considered to be appropriate (see Figure 8). The boxes for the responses to questions Q2 and Q3 revealed that the objectives and the tasks were considered to be clear in all cases. With regard to questions Q4 and Q5, the Italy 1 and Italy 2 subjects manifested a neutral judgment in relation to the complexity of both tasks. However, for Spain 1, the box for Q4 shows that the subjects judged the difficulty of the ECP comprehension tasks to be medium/low, while a neutral judgment (i.e., medium) was indicated for the complexity of the E-Plat comprehension tasks (see the box for Q5). This indicates a case in which there is a difference between the perceived complexity of a task and its actual complexity (see, for example, Table 3 and Figure 5).

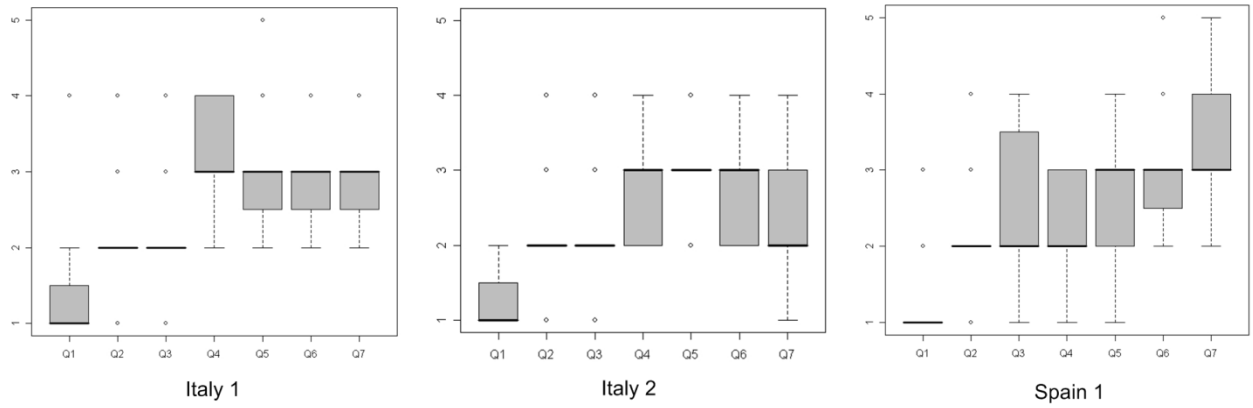


Figure 8. Subjects' responses for the experiments conducted using ECP and S2

With regard to Q6, the Italy 1 and Spain 1 subjects asserted that their knowledge of analysis object modeling and UML class diagrams was sufficient, while the Italy 2 subjects expressed a neutral judgment. With regard to Q7, the Italy 1 subjects indicated that their knowledge of the dynamic modeling and UML sequence diagrams was sufficient, while the Italy 2 and Spain 1 subjects expressed high and low knowledge, respectively.

Finally, the responses to questions Q8 and Q9 for the three experiments revealed that the subjects generally considered dynamic models and the UML sequence diagrams to be more useful in the comprehension of software requirements. With regard to Italy 1, this is a case in which a controlled experiment provides insight into the difference between the perceived usefulness of a given method and the effective advantage of using it.

Figure 9 shows that the time needed to carry out the experiments was considered to be appropriate for the Spain 2 and Spain 3 subjects (see boxes for Q1). The boxes for the responses to questions Q2 and Q3 revealed that the objectives and the tasks were considered to be clear in all the cases.

The box for Q4 in Spain 2 shows that the subjects judged the difficulty of the comprehension tasks for M-Shop to be low, while a neutral judgment (i.e., medium) was indicated for the complexity of comprehension tasks for Theater (see the box for Q5). The Q4 and Q5 boxes for Spain 3 show that the difficulty of the comprehension tasks was considered as medium. Note that the Spain 2 and Spain 3 subjects considered the tasks to be more complex than the subjects of the other three experiments. This indicates a case in which there is a difference between the perceived complexity and its actual complexity (see boxes for Q4 and Q5 of Figure 8).

The box for Q6 in Spain 2 shows that the subjects asserted that their knowledge regarding analysis object modeling and the UML class diagrams was sufficient (i.e., medium), while a worse judgment in their knowledge of dynamic modeling and sequence diagrams was indicated (see the box for Q7). The boxes for Q6 and Q7 of Spain 3 show that the subjects asserted that their knowledge of analysis object modeling and the UML class diagrams was sufficient, along with that of dynamic modeling and sequence diagrams.

Finally, the responses to questions Q8 and Q9 revealed that, for both experiments, the subjects generally considered object modeling and the UML class diagrams to be more useful in the comprehension of software requirements.

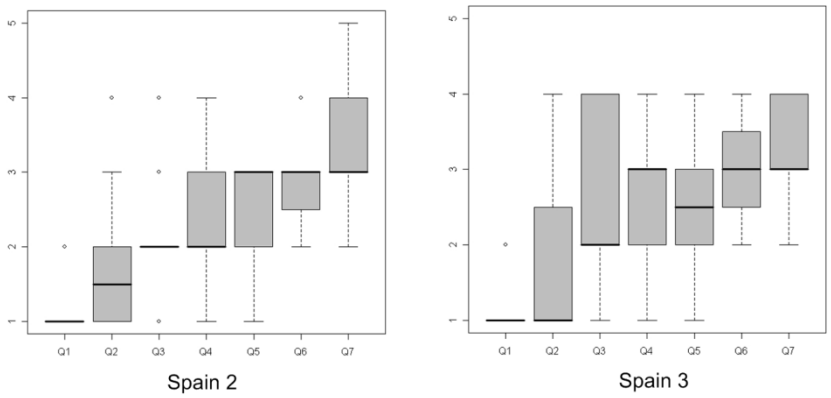


Figure 9. Subjects' responses for the experiments conducted using M-Shop and Theater

5. Family Data Analysis

This section provides a summary for the results obtained from all the experiments. We first present an analysis of the results in the context of the family of experiments, followed by the results of a meta-analysis that aggregates the empirical findings obtained in the individual experiments.

5.1 Summary of Results

We performed a global analysis of the results in the context of a family of experiments in order to determine whether the general goal of our study had been achieved. We also studied the results of all the experiments together to search for possible differences. A general summary of the experiments and the results is provided in Table 6.

Table 6. Summary of the experiments

Exp. ID	Context	Description	Type of replication	# of subjects	Type of subjects	Exp. objects	Hypothesis Rejected?	Influence of co-factors
Italy 1	University of Basilicata	Original experiment	–	24	Bachelor Students	ECP E-Plat	No	No
Italy 2	University of Salerno	Different environment and subjects with respect to Italy1	Internal	24	Master's Students	ECP E-Plat	Yes	Influence of Object on NO_DM
Spain 1	Universidad Politécnica de Valencia	Different environment and subjects with respect to Italy1.	External	28	Master's students	ECP E-Plat	Yes	Influence of Object on DM and Ability on NO_DM
Spain 2	Universidad Politécnica de Valencia	Replication that varies the manner in which Italy 1 is run	External	20	PhD students	M-Shop Theater	Yes	No
Spain 3	Spanish Professionals	Replication that varies the manner in which Italy 1 is run	External	16	Professionals	M-Shop Theater	Yes	No

The general result of the family of experiments indicates that support was found for hypothesis H_{a0} in all the experiments, with the exception of the original experiment. This indicates that dynamic models significantly improved the comprehension of software requirements. However, the experimentation opens up a number of issues which are summarized and discussed below:

- **Comprehension.** The comprehension mean scores of the Spain 1 subjects were low when compared to the subjects in the other data sets (see Table 3). This might be owing to the students' low familiarity with ECP. Although the complexity of ECP and E-Plat can be considered as comparable, it would appear that the subjects were more familiar with E-Plat. Moreover, the comprehension mean scores obtained by the Spanish PhD students and professionals are comparable to those of the Italian students.
- **Subjects' Experience.** The Spanish students and professionals, and the Italian students from Italy 2, had more experience than those in Italy 1, particularly with regard to the use of dynamic models and

sequence diagrams. This finding, together with the results of each experiment, therefore indicates that more experienced subjects obtain a greater benefit from dynamic models.

- **Influence of Object.** The data analysis showed that, with the exception of Italy 2, there was no interaction between Method and Object. Moreover, the results of the family of experiments indicated that dynamic models significantly improved the comprehension of software requirements, independently of task complexity. This means that, in spite of increasing the task complexity, the subjects obtained a better comprehension level when using a dynamic model.
- **Influence of Ability.** The interaction plots provided further information that might be useful in improving our understanding of what type of subjects (high or low ability) will benefit from the use of dynamic models. In particular, all the experiments indicated that there is an interaction between Method and Ability with the exception of Italy 1. Furthermore, the experimental results indicated that the high ability subjects obtained better results than the low ability subjects when using dynamic models, with the exception of Italy 2 in which a slight difference in favor of low ability subjects was observed. This may indicate the presence of a possible experience threshold that the subjects should have, signifying that high ability subjects benefit from the use of dynamic models.

The five experiments allowed us to gather knowledge concerning those conditions in which the use of dynamic models is more effective. According to the previously discussed results, we can conclude that subjects with a high ability and more experience benefit from dynamic models more than low ability and less experienced subjects. The results also showed that requirements complexity did not influence the subjects' comprehension when using dynamic models. However, our results are only applicable to requirements specifications that use sequence diagrams. The generalization of these results to other types of the UML diagrams will be the subject of future work.

In summary, the results support the hypothesis stated. Running a family of experiments (including replications) rather than a single experiment provided us with more evidence of the external validity, and thus the generalizability, of the study results. The same hypothesis was tested and confirmed (with very few exceptions) in five different environments using different experimental objects and four types of participants. Each replication provided further evidence of the confirmation of the hypothesis. We can thus conclude that the general goal of the empirical validation has been achieved.

5.2 Meta-Analysis

We also employed a meta-analysis with which to integrate the results of the individual experiments in order to obtain stronger results with regard to the contribution of dynamic models to requirements comprehension. Although several statistical methods exist with which to aggregate and interpret a set of results obtained through different experiments that are inter-related [34], [38], [57], [62], we used meta-analysis because it allowed us to extract more general conclusions, since the experimental conditions were the same.

Meta-analysis is a set of statistical techniques for combining the different effect sizes of the experiments to obtain a global effect of a factor. In particular, the estimation of effect sizes can be used after comparing studies to evaluate the average impact across studies of an independent variable on the dependent variable. As measures may come from different settings and may not be homogeneous, a standardized measure for each experiment needs to be obtained, and these measures must then be combined to estimate the global effect size of a factor. In our study, we considered that Method was the main factor in the family of the experiments.

The meta-analysis was conducted by using the Meta-Analysis v2 tool [10]. As suggested by a previous study [26], we employed the mean value obtained using dynamic models minus the mean value achieved when not using them to calculate the effect sizes for Comprehension for each of the individual experiments, and from these values, we obtained the Hedges' g metric [38], [43], which was used as a standardized measure. This value expresses the magnitude of the effect of Method which, in our case is the use of dynamic models that are relative to the within-group standard deviations.

The Hedges' g metric is a weighted mean whose weights depend on sample size:

$$\overline{Z} = \frac{\sum_i w_i z_i}{\sum_i w_i} \qquad (4)$$

where $w_i = 1/(n_i-3)$ and n_i is the sample size of the *i-th* experiment.

Table 7 summarizes the results obtained with the meta-analysis. For each experiment, we report both the values of the Hedges' g metric and the effect size. Specifically, the cells related to the effect size contain two pieces of information:

- An indication of the magnitude of effect size, classified as *small*, *medium*, or *large*. The magnitude of effect size is computed based on the standardized difference between two means. For studies conducted in Software

Engineering, we can classify the magnitude of effect sizes according to their numerical value: small (0 to 0.37), medium (0.38 to 1.00) and large (above 1.00) [43]. For example, an effect size of 0.5 indicates that the mean value obtained when using dynamic models is half a standard deviation larger than the mean obtained when not using them; in other words, a positive value signifies that dynamic models improve the comprehension of requirements measured by the dependent variable defined.

- An indication of whether the result is statically significant.

For the reader's convenience, we show the meta-analysis results in diagram form, as provided by the tool used. Figure 10 shows the Hedges' g metric with a confidence interval of 95% for Comprehension. It is important to note that not all the experiments contribute equally to the overall conclusion, which is represented by the diamond in the last row of the figure. Each of them receives a specific weight in the meta-analysis, i.e., the study's effect size, represented by the squares in the figure. The estimations for studies with a large sample size are more accurate, meaning that they make a greater contribution to the overall effect. However, sample size is not the only factor that contributes to the weight of a study. The weight of a study is proportional to the area of the corresponding square in the figure.

Table 7. The Hedges'g metric values for the comprehension of requirements specifications

Experiment	Hedges' g	Effect Size	Significance
Italy 1	0.092	Small	No (p=0.518)
Italy 2	0.563	Medium	**Yes (p=0.000)**
Spain 1	0,260	Small	No (p=0.053)
Spain 2	0.341	Small	**Yes (p=0.033)**
Spain 3	0.423	Medium	**Yes (p=0.019)**
Global Effect Size	0.319	Small	**Yes (p=0.000)**

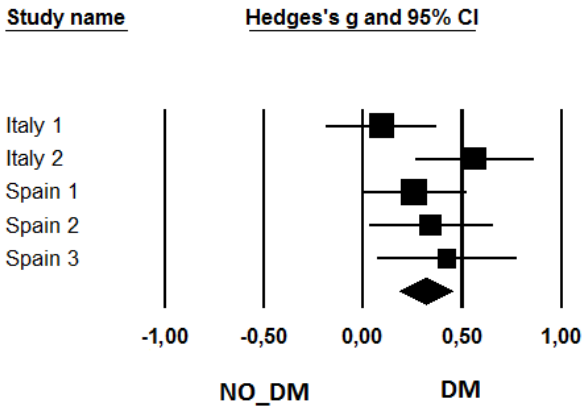


Figure 10. Comprehension of requirements specification meta-analysis

The effect size obtained varies between small and medium in all the cases, probably as a result of the number of experiments used in the data meta-analysis. Despite the fact that the first and third experiments (i.e., Italy 1 and Spain 1) did not produce significant results (see Table 7), the overall results of the meta-analysis present a significant positive effect, thus signifying that we can reject the null hypothesis, namely "the use of dynamic models does not significantly improve the comprehension of software requirements". The meta-analysis therefore strengthens the alternative hypothesis, which can be easily derived.

6. Threats to Validity

This section discusses the threats that could affect the validity of the results attained.

6.1 Internal validity

The threat to internal validity is relevant in those studies that attempt to establish a causal relationship. The experiments presented in this paper aimed to assess whether dynamic models improve the comprehension of software requirements. In our case, the subjects were students and professionals with varying knowledge and backgrounds. The Italy 1 subjects were less experienced in dynamic modeling and UML than the subjects of the other experiments. However, all the subjects found the experimental tasks to be clear, as the post-experiment survey revealed (see Section 4.3).

A key question in the threat to internal validity is whether the differences observed can be attributed to the learning effect and not to other causes. Indeed, as the subjects subsequently performed two different comprehension tasks, it may have occurred that they benefitted from the first task when performing the second one. This fact was mitigated in all the experiments of the family by the design, since each group worked, over two tasks, on different requirements in which dynamic models were and were not used. We also assessed the effect of the order of the investigated methods by using statistical tests. The threat to internal validity has been further mitigated since the subjects in each group within the same controlled experiment had similar experience with UML and software system modeling.

Another issue concerns the possible information exchange among the subjects. We prevented this in several ways. The subjects were monitored by supervisors while performing the experiments. Hence, the subjects were not allowed to communicate with each other while performing the tasks. The supervisors also controlled that no communication among subjects occurred while passing from the first task to the second. Furthermore, at the end of

each task the subjects were asked to return all the material provided.

With regard to Italy 2 and Spain 1, a different threat to internal validity was present. It was possible that the subjects of these experiments might have been able to obtain information on the tasks from the Italy 1 subjects. However, the subjects of the original experiment did not know the Italy 2 and Spain 1 subjects since they resided in different regions and countries. With regard to Spain 2 and Spain 3, the subjects could not exchange information because the experiments were conducted in two different contexts and subjects did not know each other.

6.2 External validity

External validity refers to the approximate truth of conclusions involving generalizations within different contexts. It may be threatened when experiments are performed with students [18], [37]. Other threats to external validity could be related to the subjects' representativeness with regard to software professionals. In our study, the subjects of the original experiment (Italy 1) had some knowledge of UML and dynamic modeling, as is the case of the vast majority of young software professionals working in small/medium companies in Italy and Spain. On the other hand, the Master's students in Computer Science (i.e., the subjects from Italy 2 and Spain 1) and the PhD students (i.e., the subjects from Spain 2) were probably better trained in UML and dynamic modeling than most of professionals from small/medium software companies. However, an increasing number of graduates with UML modeling skills are being integrated into the software industry market and should therefore become the standard. Further replications in different contexts are, however, needed to confirm the results attained.

A further threat that might affect the external validity concerns the size and complexity of the tasks used. In our study, for the first three experiments (i.e., Italy 1, Italy 2, and Spain 1), we used tasks that required about 30 minutes, thus conditioning their complexity. However, this threat was somewhat alleviated in the last two experiments since more complex tasks were used. They were chosen to be completed within 45 minutes. Replications using different and more complex tasks are therefore necessary.

6.3 Construct validity

Threats to construct validity could greatly affect the results of the family of experiments. In this study, construct validity may be influenced by the metrics used to obtain a quantitative evaluation of the comprehension of the software requirements, the comprehension questionnaires, and the post-experiment survey questionnaire. However, a proper design of the experiments has allowed us to mitigate these threats. Note that the measures used to obtain a

quantitative evaluation of the comprehension have been selected to achieve a balance between the correctness and completeness of the responses provided by the subjects on a given comprehension questionnaire. Furthermore, this measure has been employed in several previous controlled experiments [2], [56], [65].

The comprehension questionnaire was designed by one of the authors not involved in the modeling of software requirements. The questions in this questionnaire were defined to be of a sufficient complexity that they would not be obvious. This point was also perceived by the subjects of the controlled experiments, as the post-experiment survey questionnaires revealed. The questions in the comprehension questionnaires were also defined in order to avoid affecting the results in favor of one of the treatment. The questions were also phrased to be answered as multiple choice answers. This reduced possible ambiguities in answering the questions and enabled the subjects to accomplish the tasks within the expected time.

Multiple-choice answers also enabled us to easily obtain the subjects' comprehension scores in the comprehension tasks. We should also point out that at the end of each task these questionnaires were collected by the experiment supervisors. In order to obtain the software requirements comprehension, two researchers in each experiment worked together to analyze the comprehension questionnaires and compute the precision, recall, and F-measure values.

The post-experiment survey questionnaire was designed to capture the subjects' perception of the tasks. Our objective was principally to support and explain the quantitative results of the experiments by providing qualitative insight from the questionnaire data. Hence, the post-experiment questionnaire was not intended to statistically analyze the impact of the considered dependent variable on the factor under study. Furthermore, the post-experiment questionnaire was designed by using standard methods and scales [53].

To avoid social threats caused by evaluation apprehension, the students were not graded on the results they obtained. Moreover, the subjects were not aware of the experimental hypothesis. Finally, it is important to point out that the experimentation discussed here was carried out to evaluate whether dynamic models improve the comprehension of software requirements. Ease of comprehension/interpretation of the requirements specification was thus the sole criterion examined, since it represents a key issue for establishing agreement among stakeholders.

6.4 Conclusion validity

Threats to conclusion validity concern the issues that affect the ability to draw a correct conclusion. In this study, threats to conclusion validity concern the selection of the subjects, the data collection, the measurement reliability,

and the validity of the statistical tests. With regard to the selection of the populations, we drew fair samples and conducted our experiments with subjects belonging to these samples. Proper tests were performed in order to statistically reject the null hypothesis. In these cases, differences were found, but they were not significant, as has been explicitly mentioned and discussed. Moreover, the aggregate measure used allowed us to objectively assess the comprehension achieved by the subjects on the software requirements. Further replications on a larger dataset, preferably using professionals with different levels of experiences, are nevertheless needed to confirm the results of the family of the experiments presented here.

7. Related Work

This section discusses the related literature concerning empirical studies aimed at: *(i)* assessing the use of UML behavioral diagrams (also considering interaction diagrams) [52] in comprehension and maintenance tasks; *(ii)* evaluating the influence of subjects' ability and experience.

7.1 UML Behavioral Diagrams

Several empirical studies, with different purposes, have recently been proposed to assess both the comprehension of behavioral diagrams (e.g., statechart and sequence) and the support that they provide in the execution of maintenance tasks. For example, Cruz-Lemus *et al.* [26] presented a controlled experiment on UML statechart comprehension. This study revealed that the use of composite states in a UML statechart improves the understandability of the diagrams when subjects had previous used them.

Dzidek *et al.* [30] reported on the results of a controlled experiment conducted with professional developers to investigate the costs and benefits of using UML documentation (including, for example, use case diagrams and sequence diagram) in the maintenance and evolution of existing software systems. The main result was that the use of UML made a significant impact on the functional correctness of the maintenance operations, even if it did not make a significant impact in terms of the time needed to perform the tasks. In contrast, the impact of UML models on software maintenance and evolution was investigated in [4]. This study suggested that the availability of design documentation based on UML may significantly improve the functional correctness of both changes and the design quality when complex tasks have to be accomplished.

With regard to the works that study the comprehension of dynamic models, from the point of view of the modeling technique employed, Britton *et al.* [14] performed an experiment to investigate the relationship between

user preferences in the UML sequence or collaboration diagrams with their accuracy in understanding information modeled using these diagrams. The context of the study consisted of first year Computer Science students. The study revealed that the effect of the diagram type was not significant. Moreover, user preference for sequence diagram before performing the tasks was not related to significantly better performances with sequence diagrams.

The use of stereotypes to improve comprehension in UML class and collaboration diagrams was analyzed by Kuzniarz *et al.* [47]. The data analysis revealed that the use of stereotypes significantly helps both students and industrial developers to improve comprehension. Genero *et al.* [33] also investigated the use of stereotypes in the comprehension of sequence diagrams. The goal of this experiment was to provide empirical evidence on the influence of stereotypes when modelers, developers, and maintainers have to comprehend UML sequence diagrams. The results of the study showed a slight tendency in favor of the use of stereotypes in facilitating the comprehension of sequence diagrams. In [66], Xie *et al.* studied the improvement in comprehension of UML sequence diagrams with adornments to support understanding of thread interactions in concurrent programming. The results of the experiment showed that these diagrams, as opposed to purely textual representations, help students to better understand concurrent executions and concurrency concepts.

In [35], Glezer *et al.* performed a controlled experiment in which both the comprehensibility and quality of sequence and collaboration diagrams were investigated in two application domains: Management Information Systems (MIS) and Real-Time (RT) systems. The results indicated that collaboration diagrams were easier to comprehend than sequence diagrams in RT systems, but that there was no difference in comprehension of the two diagram types in MIS. Irrespective of the diagram type, it was easier to comprehend MIS interaction diagrams than those of RT systems. With regard to diagram quality, in the case of MIS, analysts created collaboration diagrams of a better quality than sequence diagrams, but there was no significant difference in the quality of the diagrams created in RT systems.

A comparison of the semantic comprehension of three dynamic UML notations, namely sequence, collaborations, and statechart diagrams was performed by Otero and Dolado in [54]. The empirical study revealed that the comprehension of dynamic models generally depended on the diagram type and on the complexity of the functionality. However, they also found that software design documents were more comprehensible when sequence diagrams were used to model dynamic behavior. In a subsequent study [55], the same authors observed that the specification of the dynamic behavior using OML (Open Modeling Language) was faster to comprehend and easier to interpret than when using UML.

Finally, it is worth mentioning the growing interest of both academia and industry in Domain-Specific Modeling (DSM) languages. The proponents of these languages claim that a key driver of DSM is easier comprehension of system structure and behavior, which should make building, evaluating, and maintaining models easier. In this context, Cao *et al.* [17] investigated how the use of a DSM language or UML affects model comprehension, the correctness of changes, and the degree of changes made during a maintenance task. The experimental results showed that structural and behavioral maintenance can be significantly easier and faster with a DSM language than with UML.

7.2 Subjects' Ability and Experience

Briand *et al.* [12] established that training is required in order to achieve better results when UML is completed with the use of OCL (Object Constraint Language) [51]. The authors found that OCL has the potential to improve an engineer's ability to understand, inspect, and modify a system modeled with UML. A significant interaction between ability and treatment in the case of defect detection was also observed.

With regard to the use of stereotypes to enhance comprehension, Ricca *et al.* [56] presented the results of four experiments carried out to assess the effectiveness of the UML stereotypes proposed by Conallen [22] to design Web applications. The experiments involved subjects with different levels of experience and ability. The data analysis revealed that it was not possible to conclude that the use of stereotypes significantly improves the subjects' comprehension. However, the analysis highlighted that ability significantly interacts with the treatment considered (i.e., using or not using Conallen's stereotypes). Indeed, subjects with a low ability achieved significant benefits from the use of stereotypes, while subjects with a high ability obtained a comparable comprehension level with or without using stereotypes. The authors thus concluded that the use of stereotypes reduces the gap between novice and experienced software engineers.

Canfora *et al.* [16] conducted an empirical investigation with students with different backgrounds, i.e., non computer science and computer science. The goal was to evaluate how the students' backgrounds affect the knowledge built when tasks are performed in pairs. The results indicated that pairs composed of subjects with different backgrounds performed worse than pairs composed of subjects with similar backgrounds.

8. Conclusion and Final Remarks

We have reported the results of a family of five experiments conducted to assess whether dynamic models improve stakeholders' comprehension when dealing with software requirements. The experiments were performed in different locations and with subjects including students and professionals of different abilities and levels of experience with UML.

The context of the original experiment was a group of second-year Computer Science undergraduate students from the University of Basilicata in Italy. The results of this experiment showed that the subjects judged the use of dynamic models to be useful, although the statistical tests revealed that there was no significant difference in their comprehension of software requirements when dynamic models were used. Possible reasons for this might be that the subjects were familiar with the domain of the specifications employed or that the subjects had no adequate previous experience in modeling with UML. In order to verify these findings, we carried out four replications of this experiment with more experienced students and professionals from Italy and Spain. The goal of these replications was to identify potentially important environmental factors that may affect the experimental results.

The results of all the replications revealed that dynamic models significantly improved the comprehension of software requirements. The data analysis also indicated the presence of a possible experience threshold, signifying that subjects with a high ability obtain greater benefit from dynamic models. On the other hand, in those cases in which dynamic models were not used, low ability subjects comprehended software requirements better. One possible explanation for this might be the fact that high ability subjects may make several interpretations of the requirements specifications that only include the static structure of the software systems (represented by the functional and analysis models). This is an interesting issue from the methodological point of view, and further and more in-depth empirical investigations into the comprehension of software requirements are needed.

Although the results obtained for the family of experiments indicated that dynamic models improved the way in which subjects comprehend requirements specifications, we performed a meta-analysis to aggregate the results of the different empirical studies in order to obtain stronger experimental evidence. The overall results of the meta-analysis showed that dynamic models significantly improved the comprehension of software requirements.

Although the experiments involved trained students and professionals as subjects, if we are to confirm and generalize our results, it is necessary to replicate the experiments on other subjects, including a large group of professionals with different levels of experience. Future work will be also devoted to investigating how the system

domain affects the subjects' comprehension. It would also be worth analyzing whether the additional effort and cost involved in creating a dynamic model is paid back by the improved comprehension (from 15% for Spain 2 and Spain 3 to 22% for Italy 2) of the software requirements. In fact, from a managerial point of view, the adoption of dynamic modeling should take into account the costs it implies.

Acknowledgments

The authors would like to thank all the subjects who participated in the experiments.

References

[1] B. Anda, D. I. Sjøberg, and M. Jorgensen, "Quality and understandability of use case models", *Proceedings of European Conference on Object-Oriented Programming*, LNCS 2072, pp .402–428, 2001. Springer-Verlag.

[2] B. Anda, K. Hansen, I. Gullesen, and H.K. Thorsen, "Experiences from Using a UML-Based Development Method in a Large Safety-Critical Project", *Empirical Software Engineering* 11(4), pp. 555-581, 2006.

[3] G. Antoniol, G. Canfora, G. Casazza, A. De Lucia, and E. Merlo, "Recovering traceability links between code and documentation". *IEEE Trans Software Engineering*, 28(10), pp. 970–983, 2002.

[4] 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*, 32(6), pp. 365–381, 2006.

[5] R. Baeza-Yates and B. Ribeiro-Neto, "*Modern Information Retrieval",* Addison-Wesley, 1999.

[6] V.R. Basili, "The Experimental Paradigm in Software Engineering", *Proceedings of the International Workshop, Experimental Software Eng. Issues: Critical Assessment and Future Directions*, LNCS 706, 1993. Springer-Verlag.

[7] V.R. Basili, R.W. Selby, and D.H. Hutchens, "Experimentation in Software Engineering", *IEEE Transaction on Software Engineering*, 12(7), pp. 733-743, 1986.

[8] V.R. Basili, "The Role of Experimentation in Software Engineering: Past, Current, and Future", *Proceedings of International Conference on Software Engineering*, pp. 442-449, 1996.

[9] V.R. Basili, F. Shull and F. Lanubile, "Building Knowledge through Families of Experiments," *IEEE Transactions on Software Engineering,* 25, pp. 456-473, 1999.

[10] Biostat, Comprehensive Meta-Analysis v2. 2006.

[11] B. W. Boehm, "*Software Engineering Economics",* Englewood, Cliffs, NJ: Prentice-Hall, 1981.

[12] L. Briand, Y. Labiche, M. Di Penta, and H. Yan-Bondoc, "An experimental investigation of formality in UML-based development", *IEEE Transactions on Software Engineering*, 31(10), pp. 833–849, 2005.

[13] L. Briand, M. Di Penta, and Y. Labiche, "Assessing and improving state-based class testing: A series of experiments", *IEEE Transaction on Software Engineering*, 30(11), pp. 770–793, 2004.

[14] C. Britton, M. Kutar, S. Anthony, T. Barker, S. Beecham, and V. Wilkinson, "An empirical study of user preference and performance with UML diagrams", *IEEE Symposia on Human Centric Computing Languages and Environments*, pp. 31–33, 2002. IEEE press.

[15] B. Bruegge and A. Dutoit, "*Object-Oriented Software Engineering Using UML, Patterns, and Java*", Prentice Hall, 2004.

[16] G. Canfora, A. Cimitile, F. Garcia, M. Piattini, and C.A. Visaggio, "Confirming the Influence of Educational Background in Pair- Design Knowledge through Experiments," *Proc. of ACM Symposium Applied Computing*, pp. 1478-1484, 2005.

[17] L. Cao, B. Ramesh, and M. Rossi, "Are Domain-Specific Models Easier to Maintain Than UML Models?", IEEE Software, 26(4), pp.19–21, 2009.

[18] J. Carver, L. Jaccheri, S. Morasca, and F. Shull, "Issues in using students in empirical studies in software engineering education", *Proceedings of International Software Metrics Symposium*, pp. 239–249, 2003. IEEE press.

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

[20] M. Ciolkowski, F. Shull, and S. Biffl, "A family of experiments to investigate the influence of context on the effect of inspection techniques". *Proceedings of the 6th International Conference on Empirical Assessment in Software Engineering*, Keele, UK, pp. 48–60, 2002.

[21] M. Colosimo, A. De Lucia, G. Scanniello, and G. Tortora "Evaluating Legacy System Migration Technologies through Empirical Studies" *International Journal on Information and Software Technology*, Elsevier, 51(12), pp. 433-447, 2009.

[22] J. Conallen, "*BuildingWeb Applications with UML*", Addison-Wesley Object Technology Series, 1999.

[23] W. J. Conover, "*Practical Nonparametric Statistics*", Wiley, 3rd edition, 1998.

[24] G. Costagliola, A. De Lucia, F. Ferrucci, C. Gravino, and G. Scanniello "Assessing the Usability of a Visual Tool for the definition of E-learning Processes", *International Journal of Visual Languages & Computing, Elsevier*, (available at http://dx.doi.org/10.1016/j.jvlc.2008.01.003).

[25] K. Cox, A. Aurum, and R. Jeffery, "An Experiment in Inspecting the Quality of Use Case Descriptions", *Journal of Research and Practice in Information Technology*, 36(4), pp. 211–229, 2004.

[26] J. A. Cruz-Lemus, M. Genero, M. E. Manso, S. Morasca, and M. Piattini, "Assessing the understandability of UML statechart diagrams with composite states - A family of empirical studies". *Empirical Software Engineering*, 14 (6), pp. 685-719, 2009.

[27] J. L. Devore and N. Farnum, *Applied Statistics for Engineers and Scientists*, Duxbury, 1999.

[28] G. A. Di Lucca and A. R. Fasolino, "Testing Web-based applications: The state of the art and future trends", *Information & Software Technology*, 48(12), pp. 1172-1186, 2006.

[29] A. Davis, "*Software Requirements: Objects, Functions and States*", Prentice Hall, 1993.

[30] W. J. Dzidek, E. Arisholm, and L. C. Briand, "A Realistic Empirical Evaluation of the Costs and Benefits of UML in Software Maintenance", *IEEE Transactions on Software Engineering*, 34(3), pp. 407-432, 2008.

[31] N. Fenton, "How Effective Are Software Engineering Methods?" *Journal of Systems and Software,* 22(2), pp. 141-146, 1993.

[32] A. Finkelstein, "Requirements engineering: an overview", *Proceedings of Asia-Pacific Software Engineering Conference*, 1993.

[33] M. Genero, J.A. Cruz-Lemus, D. Caivano, S. Abrahão, E. Insfran, and J.A. Carsí, "Assessing the Influence of Stereotypes on the Comprehension of UML Sequence Diagrams: A Controlled Experiment", *Proceedings of International Conference on Model Driven Engineering Languages and Systems,* LNCS Volume 5301, Springer Berlin / Heidelberg, pp. 280-294, 2008.

[34] G.V. Glass, B. McGaw, and M.L. Smith, Meta-Analysis in Social Research: Sage Publications, 1981.

[35] C. Glezer, M. Last, E. Nachmany, and P. Shoval, "Quality and comprehension of UML interaction diagrams: an experimental comparison", *Information and Software Technology*, 47(10), pp. 675–692, 2005.

[36] C. Gravino, G. Scanniello, and G. Tortora, "An Empirical Investigation on Dynamic Modeling in Requirements Engineering", *Proceedings of International Conference on Model Driven Engineering Languages and Systems*, LNCS Volume 5301, Springer Berlin / Heidelberg, pp. 615-629, 2008.

[37] J. E. Hannay and M. Jørgensen, "The Role of Deliberate Artificial Design Elements in Software Engineering Experiments", *IEEE Transactions on Software Engineering*, 34(2), pp. 242–259, 2008.

[38] L. V. Hedges and I. Olkin, Statistical Methods for Meta-Analysis: Academia Press, 1985.

[39] S.E. Hove, and B.C.D. Anda, "Experiences from Conducting Semi-Structured Interviews in Empirical Software Engineering Research, *Proceedings of International Software Metrics Symposium,* pp. 1-10, 2005. IEEE CS Press.

[40] M .Höst, B. Regnell, and C. Wholin, "Using students as subjects—a comparative study of students and professionals in lead-time impact assessment". *In Proceedings of the 4th Conference on Empirical Assessment and Evaluation in Software Engineering*, Keele University, UK, pp. 201–214, 2000.

[41] M. Jackson, "*Software Requirements and Specifications: A Lexicon of Practice, Principles and Prejudices*", Addison Wesley, 1995.

[42] N. Juristo, N. and A.M. Moreno, "Basics of Software Engineering Experimentation", Kluwer Academic Publishers, 2001.

[43] V. Kampenes, T. Dybå, J.E. Hannay, and D.I.K. Sjøberg, "A Systematic Review of Effect Size in Software Engineering Experiments". *Information and Software Technology*, 49 (11-12), pp. 1073-1086, 2007.

[44] B.A. Kitchenham, "Guidelines for performing Systematic Literature Reviews in Software Engineering", Version 2.3, EBSE-2007-01 Technical Report, Keele University & University of Durham, July 2007.

[45] B. A. Kitchenham, S. Pfleeger, D. C. Hoaglin, K. El Emam, and J. Rosenberg, "Preliminary Guidelines for Empirical Research in Software Engineering", *IEEE Transactions on Software Engineering*, 28(8), 721–734, 2002.

[46] R. Koschke, "Survey of Research on Software Clones", *Duplication, Redundancy, and Similarity in Software*, 2006, http://drops.dagstuhl.de/opus/volltexte/2007/962.

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

[48] R.M. Lindsay and A.S.C. Ehrenberg, "The Design of Replicated Studies", *The Am. Statistician*, 47, pp. 217-228, 1993.

[49] T. Nakajo and H. Kume, "A case history analysis of software error cause-effect relationships", *IEEE Transactions on Software Engineering*, 17(8), pp. 830–838, 1991.

[50] B. Nuseibeh and S. Easterbrook, "Requirements engineering: a roadmap", *Proceedings of the Conference on the Future of Software Engineering*, pp. 35–46, 2000.

[51] OMG. *Object constraint language (OCL) specification*, version 2.0, 2005.

[52] OMG. *Unified Modeling Language (UML) Specification*, version 2.0, 2005

[53] A. N. Oppenheim, "*Questionnaire Design, Interviewing and Attitude Measurement*", Pinter Publishers, 1992.

[54] M. C. Otero and J. J. Dolado, "An initial experimental assessment of the dynamic modelling in UML", *Empirical Software Engineering*, 7(1), pp. 27–47, 2002.

[55] M. C. Otero and J. J. Dolado, "An empirical comparison of the dynamic modeling in OML and UML", *Journal of Systems and Software*, 77(2), pp. 91–102, 2005.

[56] F. Ricca, M. Di Penta, M. Torchiano, P. Tonella, and M. Ceccato, "How Developers' Experience and Ability Influence Web Application Comprehension Tasks Supported by UML Stereotypes: A Series of Four Experiments," *IEEE Transactions on Software Engineering*, 36(1), pp. 96-118, 2010.

[57] R. Rosenthal, Meta-Analytic Procedures for Social Research: Sage Publications, 1986.

[58] P. Royston, "An extension of Shapiro and Wilk's test for normality to large samples", *Applied Statistics*, 31(2), pp.115-124, 1982.

[59] C. B. Seaman, "Qualitative Methods in Empirical Studies of Software Engineering", *IEEE Transaction on Software Engineering*, 25(4), pp. 557-572, 1999.

[60] D. I.K. Sjøberg, J. E. Hannay, O. Hansen, V. B. Kampenes, A. Karahasanovic, N. Liborg, and A. C. Rekdal, "A Survey of Controlled Experiments in Software Engineering", *IEEE Transaction on Software Engineering*, 31(9), pp. 733-753, 2005.

[61] D. I. K. Sjøberg, B. Anda, E. Arisholm, T. Dybå, M. Jørgensen, A. Karahasanovic, and M. Vokác. "Challenges and recommendations when increasing the realism of controlled software engineering experiments". ESERNET 2001-2003, LNCS 2765, pages 24-38, 2003.

[62] J. A. Sutton, R. K. Abrams, R.D. Jones, A.T. Sheldon, and F. Song, Methods for Meta-Analysis in Medical Research: John-Wiley & Sons, 2001.

[63] R. J. Wieringa, "*Requirements Engineering: Frameworks for Understanding*", Wiley, 1996.

[64] C.Wohlin, P. Runeson, M. Host, M. C. Ohlsson, B. Regnell, and A.Wesslen, "*Experimentation in Software Engineering - An Introduction",* Kluwer, 2000.

[65] T. Zimmermann, P. Weissgerber, S. Diehl, and A. Zeller, "Mining version histories to guide software changes", *IEEE Transactions on Software Engineering*, 31(6), pp. 429–445, 2005.

[66] S. Xie, E. Kraemer, and R.E.K. Stirewalt, "Empirical Evaluation of a UML Sequence Diagram with Adornments to Support Understanding of Thread Interactions", *Proceedings of 15th IEEE International Conference on Program Comprehension*, pp. 123-134, 2007.