Panach, JI.; Dieste, Ó.; Marín, B.; España, S.; Vegas, S.; Pastor López, O.; Juristo, N. (2021). Evaluating Model-Driven Development Claims with Respect to Quality: A Family of Experiments. IEEE Transactions on Software Engineering. 47(1):130-145. https://doi.org/10.1109/TSE.2018.2884706

Evaluating Model-Driven Development Claims with respect to Quality: A Family of Experiments

Jose Ignacio Panach, Óscar Dieste, Beatriz Marín, Sergio España, Sira Vegas, Óscar Pastor, Natalia Juristo

Abstract—Context: There is a lack of empirical evidence on the differences between model-driven development (MDD), where code is automatically derived from conceptual models, and traditional software development method, where code is manually written. In our previous work, we compared both methods in a baseline experiment concluding that quality of the software developed following MDD was significantly better only for more complex problems (with more function points). Quality was measured through test cases run on a functional system. **Objective:** This paper reports six replications of the baseline to study the impact of problem complexity on software quality in the context of MDD. **Method**: We conducted replications of two types: strict replications and object replications. Strict replications were similar to the baseline, whereas we used more complex experimental objects (problems) in the object replications. **Results:** MDD yields better quality independently of problem complexity with a moderate effect size. This effect is bigger for problems that are more complex. **Conclusions:** Thanks to the bigger size of the sample after aggregating replications, we discovered an effect that the baseline had not revealed due to the small sample size. The baseline results hold, which suggests that MDD yields better quality for more complex problems.

Index Terms— D.1.2 Automatic Programming; D.2.1.e Methodologies; D.2.1.i Validation

1. Introduction

Model-driven development (MDD) [14] claims that conceptual models can represent systems abstractly. These models can then be transformed into code through model-to-code transformations. As a result, the analyst (person that builds the system) can focus on conceptual models (the problem space), relegating the implementation (solution space) to highly automated transformations. There are several degrees of code generation from conceptual models; from stubs based on UML class diagrams to fully functional systems that do not require writing a single line of code. MDD aims at generating as much code as possible from conceptual models. So, when we talk in this paper about MDD, we refer only to such methods that generate a holistic system from models.

Since the early days of MDD, several researchers have highlighted the benefits of developing software using this method. According to literature, **MDD has several benefits** such as improving code quality [35], reducing developer effort [23], improving productivity [6] and enhancing developer satisfaction [34], among others. But there are few empirical validations of these claims. Most empirical studies on MDD focus on effort and use small chunks of code generated from a conceptual model. In order to extend empirical evidence to other less analyzed software characteristics, such as software quality, we conducted an experiment in 2012 [38] to study quality, developer effort, developer productivity and developer satisfaction. The **goal of that baseline experiment** was to compare MDD versus a traditional software development method, where code is implemented manually. MDD was operationalized using INTEGRANOVA [26], a MDD tool that generates fully functional systems from conceptual models without writing any code (see details in Appendix C). Experimental objects were two textual descriptions of software systems to develop from scratch. Results showed that even though differences between MDD and traditional development were observed for quality, they were significant only for more complex problems (with more function points). Differences between MDD and traditional methods for developer efficiency, developer productivity and developer satisfaction were not significant even for complex problems. The **contribution of this paper** is to mature results from the baseline experiment through a family of experiments. Of all the response variables used in the baseline experiment, we focus here on quality since it got the most promising results. By quality

- *J.I. Panach is with Escola Tècnica Superior d'Enginyeria, Departament d'Informàtica, Universitat de València, Avinguda de la Universitat, s/n 46100 Burjassot, València, Spain. E-mail: joigpana@uv.es*
- *O. Dieste, S. Vegas and N. Juristo, are with Escuela Técnica Superior de Ingenieros Informáticos, Universidad Politécnica de Madrid, Campus de Montegancedo, 28660 Boadilla del Monte, E-mail:{odieste, svegas, natalia}@fi.upm.es*
- *B. Marín is with Escuela de Informática y Telecomunicaciones, Facultad de Ingeniería, Universidad Diego Portales, Ejército 441, Santiago, Chile. E-mail: beatriz.marin@mail.udp.cl*
- *S. España is with Utrecht University, the Netherlands. E-mail: s.espana@uu.nl*
- *O. Pastor is with Centro de Investigación en Métodos de Producción de Software,Universitat Politècnica de València, Camino de Vera s/n, Edificio 1F, 46022 Valencia, Spain. E-mail: opastor@pros.upv.es*

we mean the percentage of passed test cases. Our replications aim to check and build upon the results of the baseline experiment:

- We conducted three identical replications at Universidad Diego Portales in Chile. These replications were as similar as possible to the baseline. Strict replications increase sample size and thus the statistical power. The power of any statistical test is defined as the probability of rejecting a false null hypothesis. The baseline experiment had 13 experimental units (by experimental unit we mean a pair of subjects, since students worked in groups). This posed a threat to conclusion validity. Experiments with low power (usually due to small sample sizes) have an increased risk of type II error (failure to detect a difference between treatments when there is one).
- We conducted three differentiated replications at the Universidad Politécnica de Valencia in Spain in which problem complexity was increased (with more function points). The aim was to increase the external validity of results in the problem space. Therefore, we conducted replications using different (and more complex) experimental objects than were used in the baseline experiment.

In order to aggregate the data from the family of experiments, we pool raw data from the six replications, adding new factors that represent the differences in settings (like time pressure and problem difficulty). We reanalyze the new data set using traditional inferential statistics to identify the impact of such moderator variables (variables considered as fixed effects to aggregate the data of several replications). This approach to experiment synthesis (pooling together individual participant data) is used in more mature experimental disciplines like medicine [40]. In software engineering (SE), experiments are very often synthesized using aggregated data from replications (mean and standard deviation) rather than using individual participant data [1, 21-22, 43, 46]. Pooling together individual participant data from several replications is more powerful than using group-level statistics, in particular, effect sizes, especially for identifying moderator variables [10]. Even though, there are papers such as the work of Shepperd [47], that question the benefits of pooling data.

Results from our family of experiments show that: (1) there are significant differences in quality between MDD and traditional methods irrespective of problem complexity; (2) we observed significant differences in quality for easy problems only when we pooled together data from all replications and the baseline experiment to get 52 experimental units. (3) Differences between treatments are clearer as problem complexity increases, where MDD yields better values for quality than traditional methods.

The paper is structured as follows. Section 2 analyzes the design of other families of experiments in the area of SE. Section 3 describes the research methodology. Section 4 defines the design of the replications, Section 5 shows the statistical results after analyzing the data extracted from each replication individually. Section 6 analyzes the aggregation of the results from several replications. Finally, Section 7 presents some relevant conclusions.

2. Related Work

In a previous work [38], we reported empirical studies on the use of MDD. The variables studied were: quality [3, 7], effort [5, 9, 27, 35, 39, 46], productivity [3, 31], developer satisfaction [34]; model characteristics [4, 27], reuse [7], and compatibility [34]. Our baseline experiment focused on quality, developer effort, developer productivity and developer satisfaction. All these variables were previously studied in experiments of the literature except for productivity, which was analyzed in a case study.

As Basili et al. [4] stated, replications contribute to building a body of knowledge combining and generalizing results. In the following, we report a number of replications performed in SE. For each replication, we analyzed a set of characteristics extracted from the research questions of two mapping studies ([33] and [12]) that reviewed replications in SE. Note that our search was not confined to the papers reported in the mapping studies since existing literature reviews were conducted many years ago and do not capture the details about each replication that was reviewed separately . We merely used the mapping studies to identify the set of characteristics reported in each paper. Of all these, we selected the characteristics related to the design and results of the replications. These characteristics, shown in Appendix A-Table 12, are:

- Goal of the experiment.
- Type of replication (internal: the original researchers performed the replication; or external: independent researchers performed the replication).
- Number of replications and number of subjects.
- Confirmation or rejection of the baseline results.

Additionally, we added two new characteristics to help to understand the changes made in each design: the purpose of the replication (opportunistic-replications without a specific goal- or goal-driven) and the changes made to the baseline experiment in the replications (sixth and seventh column in Appendix A-Table 12). By opportunistic we mean experiments that are not looking for specific results, for example, experiments that do not validate a new proposal but compare existing ones.

We conducted a limited informal literature search to look for recent related work and also to provide more details about previous replications. The **goal** to look for related works was: *What are the characteristics of experiment replications in software engineering?* The **search string** used to search for literature on Scopus was “software engineering" AND ("family of experiments” OR “experiment replication” OR “series of experiments”). The **inclusion criteria** were: (IC1) papers that compare a baseline experiment versus n replications; (IC2) papers that compare replications in a family of experiments. The **exclusion criteria** were: (EC1) papers that do not describe the results of the comparison; (EC2) papers that do not describe the design of each replication; (EC3) papers that do not deal with modeling or coding to develop software; (EC4) papers that report experiments not conducted with humans. The search was run in March 2017 on Scopus. The

primary studies selected from the set of papers retrieved by the search were classified depending on the type of aggregation used in the family of experiments:

- **Narrative synthesis:** aggregating the results qualitatively [30]. This approach is flexible and easy to apply, but it is also unsystematic, unreliable, and does not scale up [11, 44].
- **Meta-analysis using effect size:** aggregating data using studies of effect sizes by the sample sizes of the individual studies (although it can also include other factors) [24]. This analysis is usually used when there is no access to raw data.
- **Meta-analysis by pooling together individual data:** aggregating data from studies using inferential statistics and adding moderator variables to the analysis. The experimenter needs to have access to the raw data of the studies [18].

2.1. Narrative synthesis

Fucci and Turhan [17] conducted a replication to analyze the relationship between number of tests generated using TDD and code quality, as well as programmer productivity. Biffl et al. [5] conducted a family of three experiments to investigate the effect of tool support with respect to defect detection and inspection meetings. Gómez and Acuña [19] performed a replication to analyze how personality factors and team climate influence software development team effectiveness, product quality and team member satisfaction. Macedo Santos and Gomes de Mendonça [28] performed a family of three experiments to investigate factors affecting the human perception of code smells. Scanniello and Erra [42] performed a replication to evaluate a think-pair-square-based method for the distributed modeling of use case diagrams. Sfetsos et al. [45] performed a replication to analyze the impact of developer personalities and temperaments on pair performance. Albayrak and Carver [2] conducted one replication of an experiment to investigate the impact of individual factors on the effectiveness of requirements inspections. Scanniello et al. [44] conducted a family of four experiments to assess whether the type of documentation for design patterns affects its comprehensibility.

2.2. Meta-analysis using effect size

González-Huerta et al. [21] performed three replications to validate QuaDAI, a method for evaluating and improving model-driven software architectures. Cruz-Lemus et al. [11] performed a family of experiments to investigate whether the use of composite states improves the understandability of UML statechart diagrams derived from class diagrams. Abrahao et al. [1] conducted a family of five experiments (one baseline and four replications) to investigate whether the comprehension of functional requirements is influenced by the use of UML sequence diagrams. Canfora et al. [8] conducted a family of six experiments (three baselines and three replications) to validate metrics for software process models. Fernández-Sáez et al. [16] performed a family of experiments with two replications to investigate whether using class and sequence diagrams improves the maintainer's performance when modifying source code.

2.3. Meta-analysis by pooling together individual data

Our search string and search criteria found only one paper by Per Runeson et al. [41]. They conducted a family of three experiments (one baseline and two replications) to explore code inspection and structural unit testing.

2.4. Conclusions of the literature review

In sum, we draw attention to the fact that replications in SE mainly use narrative synthesis ([2, 5, 19, 28, 42, 44-45]) and meta-analysis using effect size ([1, 8, 11, 16, 21]) while the use of meta-analysis pooling data ([41]) is less frequent. Maybe this imbalance is because pooling data needs access to the raw data of all replications. This requirement is generally hard to meet (although rather easier in the case of families of experiments conducted by the same or related researchers). Although this meta-analysis by pooling data is relatively uncommon in SE, we chose this technique to aggregate our replications because we do have access to the raw data and it is a powerful technique for identifying moderator variables [10].

Another conclusion is that the number of replications in families of experiments is small (fifth column of Appendix A-Table 12). Of the studied replications, five papers report one replication ([2, 17, 19, 42, 45]), six papers report from two to three replications ([5, 16, 21, 28, 41, 44]) and three papers report from four to five replications ([1, 8, 11]). Again, this might not be the whole picture but it definitely defines a trend. Our family (composed of six replications) merits the consideration of a medium- to large-sized family.

We found that it is not uncommon for families of experiments to have large numbers of subjects. There are two papers with fewer than 50 subjects ([17, 42]), six papers with from 51 to 100 subjects ([1-2, 21, 28, 41, 44]), and six papers with more than 100 subjects ([5, 8, 11, 16, 19, 45]). Our family has 52 experimental units (104 subjects). This would appear to be a small number of subjects compared with previous works.

Note that, to the best of our knowledge, no previous family of experiments has compared MDD against a traditional software development method (see third column in Appendix A-Table 12). Of the related work, there is only one paper addressing MDD ([21]). The goal of that paper was to validate software architectures, which is unrelated to our aim.

3. RESEARCH METHOD

For our experiment, we used Design Science [49] as research method. Design Science *is the design and investigation of artefacts in context*. According to Design Science, artefacts are intended to interact with a problem context in order to improve something in that context. In our research, the artefact is the software development method, while the context is the product attributes.

The main **research goal** of our work is to replicate studies about the benefit of MDD. The **research question**

is: RQM: *How is MDD better than a traditional software development method?*

Mainly, in the Design Science method, two activities should be considered: Design and Investigation. In Design, we analyze the main benefits that previous studies grant to MDD and we design an experiment to check those benefits. In Investigation, we conduct an experiment, analyze its results and discuss the conclusions.

Fig. 1 shows the steps of the methodology in detail. In the activity Design we have one task to study the existing MDD advantages in the literature (T1), then we define the experiment to contrast such advantages empirically (T2). In the activity Investigation we have one task to conduct the baseline experiment (T3) and to analyze the results (T4). The work and results of these four tasks was already published [38]. The results of the baseline experiment gave us feedback to apply small changes in the experiment design in order to check in more detail the conclusions extracted in the baseline (T5). This task is the beginning of the contribution of this paper. Next, we conduct a family of experiments in order to improve the statistical power of the baseline and to analyze the impact of the changes in the design (T6). Once we have replicated the experiment through several years in different sites, we analyze the results of such replications (T7). We can analyze the replications individually or we can aggregate them studying moderator variables. Finally, we can contrast the results of the family of experiments versus the results of the baseline. Moreover, we can contrast the conclusions with the benefits of MDD claimed in the literature (T8).

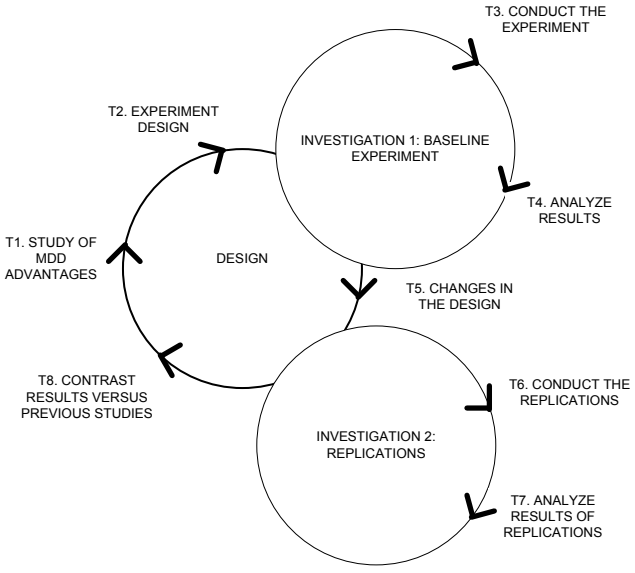

Fig. 1. Research Method according to Design Science

The family of experiments is composed of 7 experiments: 3 replications as similar as possible to the baseline; 3 replications with more complex problems, and the baseline experiment. Next, we describe the 6 replications.

4. Changes Made in the Replications

The details of the baseline experiment are published in a previous paper [38] and they can be seen in Appendix B. The baseline experiment aimed to compare MDD versus a traditional software development method through the response variables: quality, developer effort, developer productivity and developer satisfaction. Quality is a wide concept [27], we focused on the characteristic functional suitability and its sub-characteristic accuracy. We measured accuracy through the percentage of test cases run successfully. Each test case was defined as a sequence of steps; we considered each step as an item that the code must satisfy (see example in Appendix B-Table 13). We used four aggregation metrics to decide whether or not a test case was passed: All or Nothing (AN), the test is a success if every item is passed; Relaxed All or Nothing (RAN), the test is a success if at least 75% of the items are passed; Weighted Items (WI), each item has a weight depending on its importance, the test case is the addition of the weights of all the passed items; Same Weight (SM), the same as WI but each item has the same weight as the others. Developer effort was measured as time spent in the development; developer productivity as the accuracy to effort ratio; and developer satisfaction as a 5-point Likert questionnaire. All these metrics were calculated manually. The design of the baseline experiment was a paired design with the development method as factor with two treatments: MDD and traditional. The problem was a blocking variable (to avoid the learning effect between treatments). Subjects already knew a traditional method but they were trained with MDD before the experiment. From a textual description of the experimental problems, subjects had to develop a functional system from scratch (through a traditional method or through MDD). The MDD tool used in the experiment was INTEGRANOVA (Appendix C), while the choice of the traditional method was free, depending on subject' preferences. Raw data are in Appendix D and the problems in Appendix F. Results of the baseline showed that when problem complexity increases slightly, the accuracy remains stable only with MDD, obtaining better results than with a traditional method. There were no significant differences for developer effort, developer productivity and developer satisfaction.

Taking the baseline experiment as starting point, we performed two types of replications: replications that are as similar as possible to the baseline experiment (*strict replications* from now on) and replications changing the experimental objects (*object replications* from now on).

Our aim with **strict replications** is twofold: study whether results hold for different experimenters and increase sample size (therefore power). Strict replications were performed by different experimenters in a different site, but the original experimenters were involved in their coordination. Our aim with **object replications** is to study whether more complex experimental objects make differences of quality between treatments bigger. This goal is inspired by the results of the baseline experiment, where

MDD quality was less sensitive to small variations in the complexity of experimental objects. This led us to suspect that there might be significant differences between the quality of software output using a traditional method and MDD using more complex problems. The same experimenters that performed and analyzed the object replications conducted the baseline experiment.

Strict replications were conducted at Universidad Diego Portales (UDP) in Chile, while object replications were conducted at the same site as the baseline, namely the Universidad Politécnica de Valencia (UPV) in Spain. Each type of replication was run three times over three consecutive years. Table 1 shows the replications of each type by year. Raw data can be found in Appendix D.

Table 1. Experiments that form the family

Year	Baseline	Strict Replications	Object Replications
2012	Baseline	SR1	-
2013	-	SR2	OR1
2014	-	SR3	OR2
2015	-	-	OR3

Both types of replications share the essentials of the baseline experiment but we purposely made small changes. Table 2 shows the changes. To discuss these changes in the following sections, we classify experiment elements using the term *dimension* following the work of Gómez et al. [20]. A dimension is a configurable element that can be changed in a replication. The replication can, however, still be considered to be the same experiment as the baseline. We work with the four dimensions defined by Gómez et al.:

- Operationalization: Instantiation of constructs (MDD, traditional, and the quality effect) into variables (INTEGRANOVA, a traditional method based on a programming language, and accuracy).
- Population: Subjects and objects.
- Protocol: Apparatus, materials, forms and procedures.
- Experimenters: Researchers involved in conducting the experiment.

4.1. Strict replications

These three replications should be as similar as possible to the baseline experiment in order to be able to aggregate data and improve statistical power by increasing sample size. Next, we describe the characteristics of strict replications according to the four dimensions and the changes regarding the baseline experiment. Table 2 shows a summary.

Operationalization. *Factors*, *treatment definition, treatment transmission, treatment instructions, treatment resources, response variables*, *measurement procedure* and *metrics* are the same as for the baseline experiment (Appendix B). Note that even though we have the same response variables as for the baseline experiment, we only report quality here. We leave out developer effort, developer productivity and developer satisfaction for space reasons. We chose quality since the baseline experiment showed that there might be significant differences between treatments when problem complexity increases, while the other three variables did not show such trend. The sub-characteristic of quality we study is accuracy. Of the four metrics used to measure quality in the baseline (AN, RAN, WI and SW), we focused replications on SW because a comparative study of metrics is beyond the scope of this paper. We chose SW since it is the least demanding metric. For each test case, we considered the percentage of passed items (test case accuracy). System accuracy is the mean accuracy regarding the items of each test case. The *treatment application procedure* differs from the baseline experiment in that each session is 10 minutes shorter due to class schedule restrictions. This reduction in time is so small that it should not affect the results of the experiment.

Population. *Properties of experimental objects* and *subjects* were unchanged with respect to the baseline experiment. SR1 used 14 subjects, SR2 recruited 6 subjects and SR3 had 14 subjects. According to the information extracted from the demographic questionnaire, subjects had the same profile as in the baseline experiment, all of them had a background in Computer Science. Note that there are slight differences between SR2 and SR1 versus SR3; and between SR2 versus the baseline. These differences arise because SR2 is composed of only 3 experimental units, a very small number compared with the other replications. Small variations with a small sample may produce big changes in the average. Most of the subjects were novice programmers with a little experience in industry or students with no work experience at all. Most subjects had not used MDD often, even though most were acquainted with the technique. Therefore, we can conclude that all strict replications include different sample sizes of the same population used in the baseline experiment. Appendix E shows the details of subjects' previous experience with MDD and traditional development methods.

Protocol. *Experimental objects, experimental design, guides* and *measurement instruments* are the same as in the baseline experiment. The only change affects *data analysis technique,* where we used mixed model instead of General Linear Model (GLM). We purposely changed the statistical test used to analyze the data (data analysis technique, protocol dimension). The main reason behind this change is that, using a mixed model, we can deal with moderator variables. Besides, GLM only provides for one variable with repeated measures. In our design we have two variables with repeated measures: the development method and the problem. Note that all subjects apply both development methods to both problems. In the baseline experiment, we analyzed the variable problem as a covariable. Now we have decided to include such variable as a fixed variable in order to study possible significant interactions between the method and the problem. Hence, even though this change could affect the results, the advantages of using a mixed model makes the change worthwhile. We re-analyzed the results of the baseline using the mixed model to check that results were the same as for GLM and made them comparable with replications results.

Table 2. Summary of differences across experiment designs. Replications and changes are described following [20]

		Baseline Experiment	Strict Replications			Object Replications		
			SR1	SR2	SR3	OR1	OR2	OR3
Operationalization	Factor	Development method						
	Treatment Definition	Traditional development method and MDD						
	Treatment Transmission	Transport/Videoclub				Videoclub/ Transport	Transport/ Videoclub	Videoclub/ Transport
	Treatment Instructions	Free application of a traditional method vs 18h (theory classes (12 hours) and practical classes (6 hours)) of MDD						
	Treatment Application Procedure	Each treatment is applied in 2 2-hour sessions (2x2h)	Two sessions of 1:50 hours each treatment (2x1:50h)			2x2h		
	Treatment Resources	Free in a traditional method vs INTEGRANOVA in MDD						
	Metrics	Percentage of passed test cases, time, percentage of passed test cases/time, level of satisfaction	Percentage of passed test cases					
	Response Variables	Accuracy, developer effort, developer productivity, developer satisfaction						
	Measurement Procedure	Test cases, time and satisfaction questionnaire						
Population	Subjects	13 units	7 units	3 units	7 units	10 units	6 units	6 units
	Properties of Experimental Objects	Requirements specified according to IEEE standard 830-1998						
Protocol	Experimental Objects	OIP and OPP				EIP and EPP		
	Experimental design	Paired-blocked						
	Guides	INTEGRANOVA guide						
	Measurement Instruments	Test Cases/Web application/Satifaction Questionnaire						
	Data analysis techniques	GLM repeated measures	Mixed model					
Experimenters	Designer	Teachers from UPV-UPM						
	Trainer	Teachers from UPV	Teachers from UDP			Teachers from UPV		
	Monitor	Teachers from UPV	Teachers from UDP			Teachers from UPV		
	Measurer	Teachers from UPV	Teachers from UDP			Teachers from UPV		
	Analyst	Teachers from UPV						

Experimenters. *Designers* and *analysts* were unchanged with respect to the baseline experiment. There was a different *trainer, monitor* and *measurer* from the baseline experiment. The original experimenters were in contact with the replicating experimenters, and the design and instruments were the same as in the baseline. Moreover, the new experimenters were perfectly well acquainted with the MDD method and the INTEGRANOVA tool since they had been collaborating with the original experimenters for the last six years. Therefore, results should not be affected by this change.

4.2. Object replications

Since we concluded that MDD is more robust to small variations in object complexity in the baseline experiment, these three replications aim to study whether an increase in problem complexity affects results. According to this goal, we purposely made some changes with respect to the baseline experiment. Table 2 shows a summary of these changes by dimension.

Operationalization. The *treatments transmissions* are treatments to train the subjects but not useful to analyze data. Treatment transmission change affects only replications OR1 and OR3. We swapped the training problems in such a manner as "Transport" (a system to manage routes of public buses) and "Videoclub" (a system to manage films renting) were the training for MDD and for the traditional method, respectively. This change was done in order to prevent training problems from affecting the results of the experiment. OR2 follows the same order as the baseline experiment.

Population. OR1 used 20 subjects, OR2 recruited 12 subjects and OR3 had 12 subjects. There are slight differences between OR2 versus OR1 and OR3; and between OR2 versus the baseline. Subjects of OR2 had a more professional profile. Most of the subjects were novice programmers or students that had not yet developed a real system. We can conclude that object replications have the same population as the baseline experiment and strict replications. Appendix E shows the details of the subjects.

Protocol. We changed the *experimental objects* and *data analysis technique* (as in strict replications). Experimental objects (problems) were extended in the object replications, although the context of both problems is the same as in the baseline experiment: Invoice Problem and Photography Problem. In the baseline, Invoice Problem aims to manage a company of electrical appliance. Once reparation is finished, the system must create the invoice. Photography Problem aims to manage a company that works with freelance photographers. The system must register who is the owner of each photo and the amount of money to pay to each photographer. From now on, we refer problems of the baseline as Original Invoice Problem (OIP) and Original Photography Problem (OPP) respectively. Problems are described in Appendix F.

The extension in the object replications was designed to check experimentally our hypothesis (based on the findings of the baseline experiment): when object complexity increases, there are bigger differences between traditional development methods and MDD. Problems were divided into three parts in such a way that the first part is OIP and OPP. Other two parts were extensions. Subjects were not allowed to start the second part until they had completed the first (and the same applies for the second and third parts). Problems can be found in Appendix F.

The Photography Problem was extended with the functionality to support the management of delivery notes and scoops. From now on, we refer to this problem as Extended Photography Problem (EPP). In Appendix F-Fig. 8.b, the classes that support OPP are highlighted in grey while the classes that support EPP are highlighted in white. EPP has 199 function points versus the 94 function points of OPP, including CRUD operations.

The Invoice Problem was extended with functionality to manage repair training courses and manage audits. From now on, we refer to this problem as Extended Invoice Problem (EIP). Appendix F-Fig. 8.a shows the class diagram, where classes highlighted grey refer to OIP. Classes highlighted white refer to EIP. EIP has 272 function points versus 100 function points of OIP. Experimental objects description can be seen in Appendix F as they were shown to the subjects. Note that class diagrams of Appendix F are part of our solution and they were not available for subjects. Subjects built their own class diagram from the textual description and their model could be different from our solution.

Experimenters. There were no changes for this dimension with respect to the baseline experiment.

4.3. Analysis of replications

The statistical analysis was done applying the mixed model statistical test [48] with unstructured repeated covariance. The development method and the problem were defined as fixed-repeated variables (since we applied two levels of both variables to all subjects). The subjects were defined as random variables.

The assumption for applying the mixed model is normality of residuals. The normality of residuals can be tested with Saphiro-Wilk test applied to the residuals automatically calculated during the application of the mixed model test [36]. We checked the assumption of normality of residuals for all replications; details of the normality are shown in Appendix G-Table 20. There are some residuals whose p-values are lower than 0.05 but higher than 0.00. We can accept a weak normality for p-values between 0.001 and 0.05; and a high normality for p-values higher than 0.05. In the case of Row 13, we obtained a p-value of 0.00. In order to solve this problem, we applied a monotonous transformation named "reverse score" [32]. The transformation is log(Subtract X_i from highest score). After the transformation we get a p-value of 0.14, indicating that these residuals have a normal distribution.

We have used Cohen's d [9] to calculate effect size. Cohen's d is defined as the difference between two means divided by a standard deviation of the data. According to Cohen [9], the meaning of the effect size is as follows: more than 0.8 is a large effect; from 0.79 to 0.5 is a moderate effect; from 0.49 to 0.2 is a small effect.

Using the mixed model, we cannot calculate power statistically as we did in the baseline experiment (independently of the statistical tool used in the analysis). Still we used G*Power [15], finding that, for a repeated measures statistical test, we need a sample size of 16 units for an effect size of 0.8 (large effect) to get a power of 80%. The sample size of each replication is less than 16 units, which implies a low power. In order to deal with this issue, we analyze not only the data for each replication but also the aggregation of several replications, whose number of experimental units is greater than 16. From a practical point of view, small effects are not relevant since, if the difference between MDD and traditional methods is small, it is not worth the effort of learning MDD (which is usually an unknown method).

For each analysis, we report descriptive data using box-and-whisker plots to illustrate the differences between the two treatments. We apply a mixed model to calculate the p-value for method and method*problem and the mean for each treatment. If the p-value is less or equal[1] to 0.05, we assume that there are significant differences between treatments, and we calculate the effect size to analyze the magnitude of the differences. Problem is included in the analysis only when the method*problem interaction is significant. Being a blocking variable, the significance of the problem is not relevant without such an interaction.

5. Individual Replication Results

5.1. Strict replications

This analysis studies each strict replication separately. Fig. 2 shows the box-and-whisker plot for accuracy in SR1 (histogram is in Appendix H-Fig. 9). The line between the two boxes connects the means. The median is better for MDD, even though the first and third quartile values for accuracy are better using a traditional method.

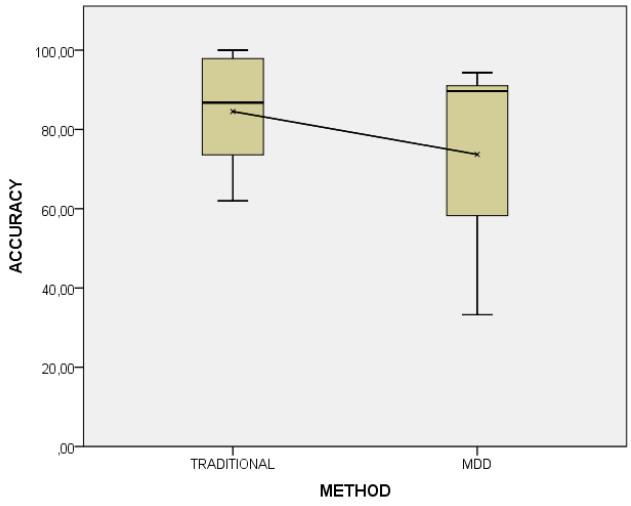

Fig. 2. Box-and-whisker plot for accuracy of SR1

[1] Traditionally, only p-values lower to 0.05 are considered as significant. We are also considering significant values equal to 0.05. This choice depends on how conservative we want to be with the analysis. We think that with this choice we are not rejecting significant results very close to 0.05 but lower, since SPSS rounds results.

Results for SR2 and SR3 are similar to SR1. Appendix H shows the histograms and box-and-whisker plots for both replications.

The power of each replication is low since all three have small sample sizes. Table 3 shows the p-values yielded by the mixed model and the means of both treatments. There are no significant results since all p-values are greater than 0.05. Moreover, we only have three units in SR2, which is not enough data to run the test.

Table 3. P-values for strict replications

Rep.[2]	E.U.[3]	Factor		Value
SR1	7	Method	p-value	0.37
			meanTraditional	84.5
			meanMDD	73.6
		Method*Problem	p-value	0.13
SR2	3	Method	p-value	-
			meanTraditional	65.9
			meanMDD	91.3
		Method*Problem	p-value	-
SR3	7	Method	p-value	0.12
			meanTrad	80.8
			meanMDD	90.8
		Method*Problem	p-value	0.98

We cannot reject the null hypothesis H_{01} (*The quality of software built using MDD or a traditional method is similar*). Note that the problem type does not affect the results since the method*problem interaction is not significant.

Comparing the results of strict replications with the baseline experiment, we find that **SR1, SR2 and SR3 get similar results to the baseline: there are no significant differences for accuracy using MDD or a traditional method for simple problems**. Since these replications were conducted at a different site and by different experimenters, we can state that *the results of the baseline experiment hold for other sites and experimenters***.**

5.2. Object replications with extended problems

Fig. 3 shows the box-and-whisker plot for accuracy in OR1 (histogram is in Appendix H). We find that the median, and the first and the third quartiles for accuracy are larger using MDD than a traditional method. This means that subjects who worked with MDD appear to achieve better results for accuracy. Note that OR1 is the replication with the biggest sample size (10 units). Even though the sample size of the baseline experiment was bigger (13 units), the problems used in OR1 are more complex. Therefore, we might expect to get significant differences even with a smaller sample size. Descriptive data for OR2 and OR3 is similar to OR1, plots can be seen in Appendix H.

The power of each replication is low. Table 4 shows the p-values, effect size and means. There are significant results for OR1 and OR3. In OR1 results are significant for the method factor. MDD has the effect of accuracy being moderately higher than for the traditional method.

[2] Replication
[3] Experimental units

In OR1 there is also a significant result for the method*problem interaction. This means that problem is affecting the method. Appendix H-Fig. 19 shows a profile plot. Accuracy for MDD is clearly better than for traditional methods with respect to both problems. However, the increase in quality for MDD is greater for the Invoice Problem than for the Photography Problem. As the result for method*problem was significant, we analyzed the blocking variable Problem in OR1. Results for Problem show that there are no significant differences between treatments.

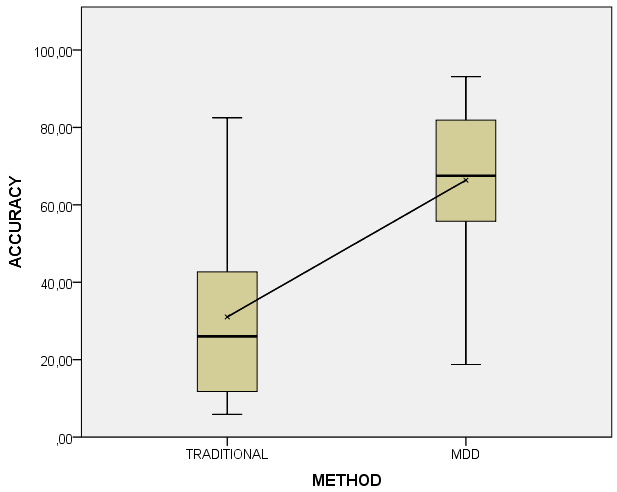

Fig. 3. Box-and-whisker plot for accuracy of OR1 with EIP and EPP

Table 4. P-values for object replications

Rep.	E.U.	Factor		Value
OR1	**10**	**Method**	**p-value**	0.00
			effect size	0.61
			meanTraditional	31
			meanMDD	66.3
		Method*Problem	**p-value**	0.01
		Problem	**p-value**	0.78
OR2	**6**	**Method**	**p-value**	0.94
			meanTraditional	30.5
			meanMDD	30.7
		Method*Problem	**p-value**	0.79
OR3	**6**	**Method**	**p-value**	0.05
			effect size	0.53
			meanTraditional	21.4
			meanMDD	47.7
		Method*Problem	**p-value**	0.14

In OR3 the significance between both treatments was still moderate but lower than in OR1. We found no significant results for the method*problem interaction. This may be due to the smaller sample size than OR1.

OR2 does not conclude significant differences between treatments. This could be caused by the low statistical power of the samples in OR2. A low power may involve accepting null hypothesis when they are false. Even though OR1 and OR3 have a low power, we identified significant differences, which lead us to think that differences in OR2 could appear with a higher power. Other reason that could justify the absence of significant differences in OR2 is the subjects' profile, that could be a confounding factor. Subjects of OR2 have more experience in the application of a traditional development method in industry than subjects of other replications.

So there are significant differences for accuracy in OR1 and OR3, where MDD is better than a traditional method. *This result confirms the finding from the baseline experiment that there appeared to be differences in accuracy in complex systems (i.e., in the object replications) that did not occur for simpler systems (i.e., in the baseline).* **Effect sizes are bigger for OR1 and OR3 than for baseline experiment**. Hence, we reject H_{01} for OR1 and OR3.

5.3. Object replications with original problems

This analysis studies each object replication considering only the part of the problems used in the baseline (OIP and OPP). The goal is to study whether or not time pressure affects the results. In object replications, subjects had more time pressure since they had 4 hours to solve EIP and EPP. Those 4 hours are the same time that subjects in the baseline had to solve OIP and OPP.

Table 5 shows the p-value, and mean of both treatments for the accuracy of the part of the problems shared by the baseline experiment and the replications. There are no significant results for any replication, even though descriptive data (Appendix H-Fig. 20 and Appendix H-Fig. 21) show MDD to be slightly better.

Table 5. P-values for object replications analyzing problem parts shared with the baseline experiment

Rep.	E.U.	Factor		Value
OR1	**10**	**Method**	**p-value**	0.12
			meanTraditional	81.7
			meanMDD	97.1
		Method*Problem	**p-value**	0.5
OR2	**6**	**Method**	**p-value**	0.15
			meanTraditional	66.5
			meanMDD	85
		Method*Problem	**p-value**	0.8
OR3	**6**	**Method**	**p-value**	0.29
			meanTraditional	75.6
			meanMDD	87.8
		Method*Problem	**p-value**	0.44

We conclude that, even though accuracy varies for each treatment, these differences are not significant. *This result matches the results of the baseline experiment: easy problems (OIP and OPP) do not reveal significant differences between MDD and a traditional method for accuracy.* Hence, we cannot reject H_{01}. Note that the results for replications with small sample sizes are not significant most likely because bigger samples sizes are needed to detect very small differences between treatments. Aggregations should address this issue.

6. Aggregated Results

In order to improve the power of the statistical test, we aggregated the results of the different replications [13], again applying the mixed model.

6.1. Aggregation of strict replications

By aggregating the three strict replications, we get 17 experimental units (7 units SR1, 3 units SR2 and 7 units SR3). Fig. 4 shows the box-and-whisker plot for accuracy (histogram is in Appendix H-Fig. 26). There is an overlap between both treatments, although the median of MDD yields better accuracy.

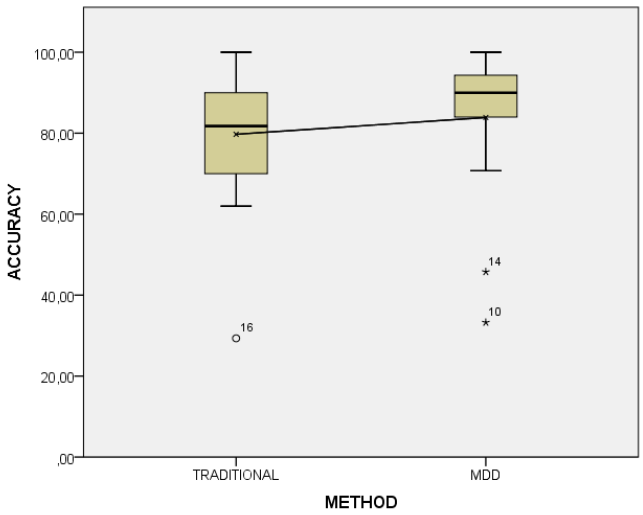

Fig. 4. Box-and-whisker plot for accuracy of SR1+SR2+SR3

Table 6 shows the p-values and means of both treatments for this aggregation. There are no significant results since all p-values are greater than 0.05.

Table 6. P-values for the aggregation of strict replications

Rep.	E.U.	Factor		Value
SR1+ SR2+ SR3	17	Method	p-value	0.54
			meanTraditional	79.7
			meanMDD	83.8
		Method*Problem	p-value	0.09

The results of the aggregation of strict replications conclude that **accuracy is slightly better for MDD than for a traditional method even though these differences are not significant**. *These results match the findings from the individual analysis of each strict replication, as well as of the baseline experiment*. Results hold even with a sample size greater than for the baseline experiment (13 units versus 17).

To improve statistical power, we aggregated strict replications together with the baseline experiment, resulting in a sample size of 30 experimental units. The results of this aggregation are the same as for the strict replications, that is, there are no significant differences between the two methods. Therefore, we have to conclude that power is not behind the fact that results are not significant.

6.2. Aggregation of object replications with extended problems

Aggregating the three object replications using complex problems (EIP and EPP), we have 22 experimental units (10 units in OR1, six units in OR2 and six units in OR3). Fig. 5 shows the box-and-whisker plot for accuracy (histogram is in Appendix H-Fig. 27). Medians, and the first and third quartiles show that MDD yields better results than traditional methods. Moreover, only MDD gets an important amount of samples with accuracy higher to 60%. MDD yields better accuracy than traditional methods.

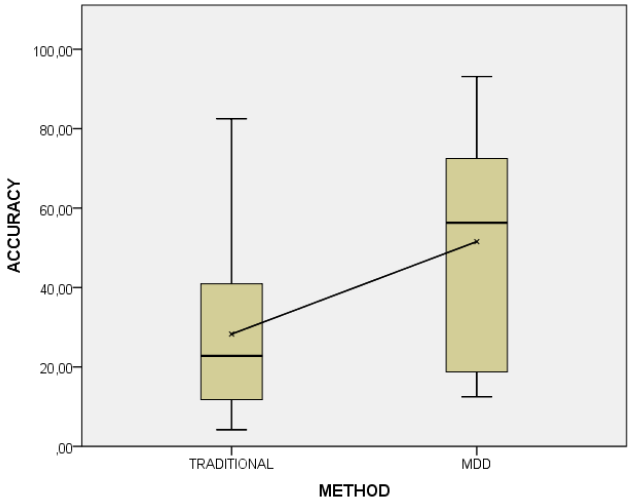

Fig. 5. Box-and-whisker plot for accuracy of OR1+OR2+OR3 studying EIP and EPP

Table 7 shows the p-values, effect sizes and means of both treatments for this aggregation. Since the p-value is less than 0.05, differences between treatments are significant. The value of 0.42 for effect size means that there are moderate differences between the two treatments. Accuracy with MDD is moderately higher than for a traditional method. There are no significant results for the method*problem interaction.

Table 7. P-values for the aggregation of object replications with EIP and EPP

Rep.	E.U.	Factor		Value
OR1+ OR2+ OR3	22	Method	p-value	0.00
			Effect size	0.42
			meanTraditional	28.2
			meanMDD	51.5
		Method*Problem	p-value	0.66

The results of the aggregation of object replications suggest that **accuracy for MDD applied to complex problems is significantly better than for a traditional method**. This result matches the findings of the analysis of each object replication individually. These results also confirm the hypothesis that we developed based on baseline outcomes: differences for accuracy between the two treatments are more evident for complex problems. After increasing the problem complexity with respect to the baseline, there are differences. Therefore, we reject H_{01}. *Hence, the small scale of the experimental objects used appears to be the main reason for not detecting significant results in the baseline experiment.*

6.3. Aggregation of object replications with original easy problems

This analysis pools together all object replications considering only the part of the problems that they have in common with the baseline.

Box-and-whisker plot and histogram can be found in Appendix H-Fig. 29, they are similar to the aggregation with extended problems. Table 8 shows the p-values of the aggregation of object replications considering the part of the problems that they have in common with strict replications and the baseline experiment (OIP and OPP). We get significant results for method. The effect size means that differences between treatments are moderate. Therefore, accuracy is better for MDD than for traditional methods.

Table 8. P-values for the aggregation of object replications analyzing OIP and OPP

Rep.	E.U.	Factor		Value
OR1+ OR2+ OR3	22	Method	p-value	0.01
			Effect size	0.73
			meanTraditional	75.9
			meanMDD	91.2
		Method*Problem	p-value	0.85

The aggregation of object replications reveals **significant differences between MDD and traditional methods for easy problems (OIP and OPP) developed under time pressure**. *These differences did not show up when we analyzed the same easy problems in strict replications or the baseline experiment (either individually or after aggregation)*. Some baseline observations led us to suspect that differences between traditional methods and MDD occur only for complex problems. However, the results of the aggregation of object replications considering easy problems developed under time pressure clarify the picture. Notice that this is a bigger sample size (22 experimental units). However, it is smaller than the size of the aggregated sample of strict replications and baseline experiment (30 experimental units), which did not reveal any significant effect. Therefore, the reason for detecting a non-significant effect could be a moderator variable rather than an increase in power. To get a reliable response, we aggregated all the experiments studying time pressure.

6.4. Aggregation of strict replications + object replications + baseline experiment to study time pressure

This analysis pools together the data of the whole family of experiments. The aggregation studies time pressure as a moderator variable. Subjects in the strict replications and baseline experiment were subject to less time pressure than subjects in the object replications.

We aggregate the whole family of experiments considering only the part of the problems common to strict replications, object replications and the baseline experiment (OIP and OPP). There are two groups of replications depending on the time pressure for subjects. Subjects of strict replications and the baseline experiment had four hours to develop easy problems (OIP and OPP), whereas subjects of object replications had four hours to develop problems that were three times more complex (EIP and EPP had 3 exercises and OIP and OPP had 1 exercise), of which OIP and OPP are part. We added time pressure as moderator variable to distinguish between the time pressure that subjects experienced in each replication. This moderator variable has two levels: high and low.

Fig. 6 shows the box-and-whisker plot for the aggregation of all the replications with the baseline. The median, the first and the third quartiles are better for MDD at both pressure levels. This means that subjects using MDD achieve better results than subjects working with a traditional method. Note that the median of MDD is better for a high time pressure.

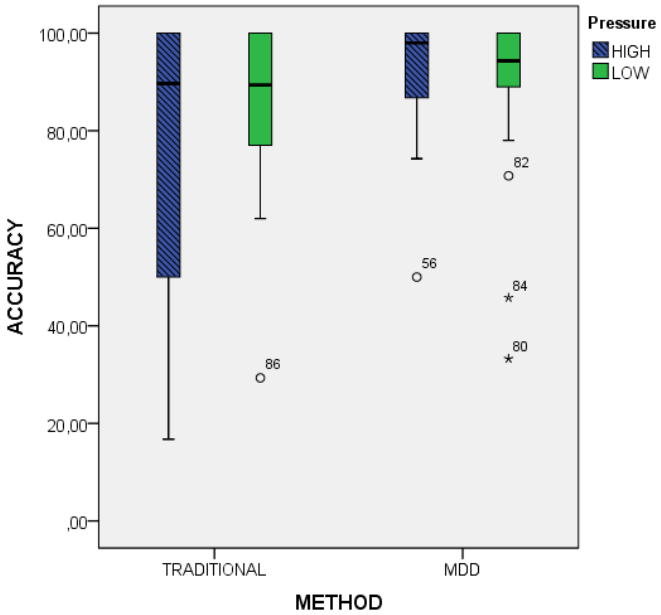

Fig. 6. Box-and-whisker plot for accuracy of strict replications+ object replications+baseline experiment for OIP and OPP

Table 9. P-values for replications of strict replication + object replications + baseline experiment considering the common part of the problems (OIP and OPP)

Rep.	E.U.	Factor		Value
SR1+ SR2+ SR3+ OR1+ OR2+ OR3+ Base.	52	Method	p-value	0.00
			Effect size	0.47
			meanTraditional	81.7
			meanMDD	90.3
		Method*Problem	p-value	0.31
		Pressure	p-value	0.73
		Method*Pressure	p-value	0.62
		Pressure*Problem	p-value	0.88

Table 9 shows the p-values of the aggregation between strict replications, object replications and the baseline experiment considering the common part of the problems (OIP and OPP). Note that these p-values have been calculated after applying the transformation log(Subtract X_i from highest score) since residuals did not followed a normal distribution. Results show that there are significant results for the method factor, with a moderate effect

in favour of MDD versus traditional development. Table 9 includes the pressure moderator variable and the method*pressure interaction. Neither is significant, although the interaction is not far off. Note that the sample size needed to detect significant interactions is bigger than what is required to detect main effects. We suspect that the result might be significant with a bigger sample size. Therefore, the moderator variable does not affect accuracy. The method*problem and pressure*problem interactions do not have significant results. This means that the blocking variable is not affecting accuracy.

We conclude that **even with easy problems (some solved under time pressure and others not), there is a small difference between MDD and traditional methods with a sample size of 52 experimental units**. *This outcome contradicts the baseline results*. After aggregating all replications and increasing statistical power, differences between treatments are bigger, even though problems are easy. Note that the effect size for method (0.47) is small, and therefore its detection requires a bigger sample size. However, we tend to think that the main reason is time pressure rather than power, although we have been unable to confirm this point. It might take a bigger sample size to appreciate the role played by time pressure. Therefore, we should expect its effect to be small.

6.5. Aggregation of Strict Replications + Object Replications + Baseline Experiment to Study Problem Complexity

This analysis pools together the whole family of experiments adding problem complexity as a moderator variable, since the complexity of problems was not the same in both types of replications.

We aggregated the replications differentiated by the problem complexity. We add problem difficulty as moderator variable to the data analysis to differentiate the complexity of the problem used in each replication. This moderator variable has two levels: easy and difficult.

Fig. 7 shows the box-and-whisker plot for accuracy. The median, the first and the third quartiles are different for MDD and traditional methods. This means that the accuracy values are better for MDD. While the medians for accuracy with respect to easy problems are similar in both MDD and traditional methods, the median with respect to complex problems is clearly better for MDD.

Table 10 shows the p-values for the aggregation using the difficulty moderator variable to differentiate the complexity of problems used in each type of replication. Results for method are significant with an effect size of 0.38, which is a small effect. The means of both treatments show that accuracy is better for subjects working with MDD than with a traditional method. The comparison of treatments taking into account problems of varying difficulty may be the reason for there being a smaller effect size than when aggregating object replications. Remember that by aggregating only complex problems (Table 7), there was a p-value of 0.00 and a moderate effect size of 0.42 for the method variable. When we mix easy problem replications with no significant effect and complex problem replications with a moderate effect in the same aggregation, the impact of MDD on accuracy becomes fuzzy.

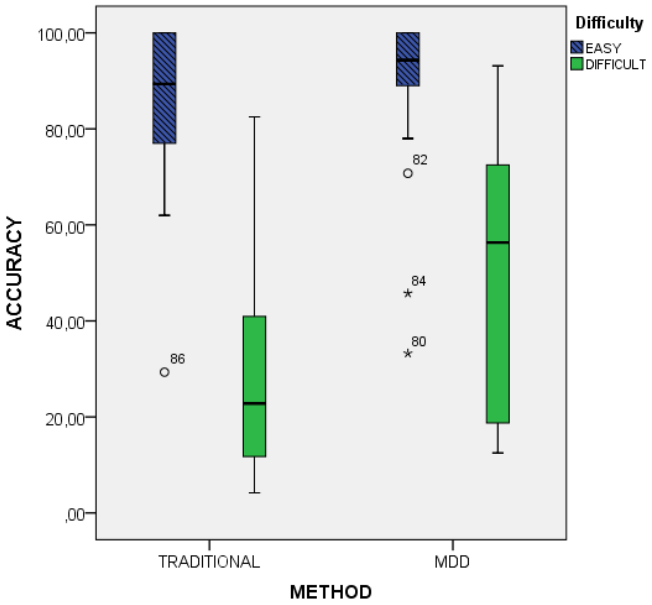

Fig. 7. Box-and-whisker plot for accuracy of strict replications+ object replications+baseline experiment for problems of different difficulty

Table 10. P-values for strict replication + object replications + baseline considering the problems as they were shown to subjects

Rep.	E.U.	Factor		Value
SR1+ SR2+ SR3+ OR1+ OR2+ OR3+ Base.	52	Method	p-value	0.00
			Effect size	0.38
			meanTraditional	61.5
			meanMDD	73.5
		Method*Problem	p-value	0.32
		Difficulty	p-value	0.00
			Effect size	2.1
			meanEasy	87.7
			meanDifficulty	39.9
		Method* Difficulty	p-value	0.00
		Difficulty*Problem	p-value	0.29

In Table 10, we also identified **a significant result for the difficulty moderator variable with a large effect size** (2.1). Looking at means, accuracy is better for easy problems (OIP and OPP). This makes sense since subjects had the same amount of time to develop software to solve both easy problems and complex problems. Therefore, the quality of the software developed to solve the easy problems should be much higher. There are also **significant results for the method*difficulty interaction.** They show that the improved accuracy achieved using MDD versus traditional methods depends on the problem difficulty. There are bigger differences between MDD and traditional methods when the problems to be solved are complex. Appendix H-Fig. 30 shows the profile plot for method*difficulty. The difference in accuracy between treatments is bigger for difficult problems, where MDD achieves better results. Therefore, MDD gets better accuracy results for difficult problems. This is consistent with

the findings for the baseline. This result makes sense since the analyst tends to make more mistakes using a traditional method for a more complex problem. The automation provided by MDD can avoid these mistakes.

Since experimental units are the same in Table 9 and Table 10, we can compare the roles of time pressure and difficulty. Note that both moderator variables are interrelated in our family of experiments. A low time pressure is consistent with an easy problem, whereas a high time pressure is consistent with a difficult problem. According to these results, time pressure appears to have a smaller impact than difficulty, since the sample size that detects method*difficulty does not detect the method*pressure interaction.

We conclude **that there are significant differences in accuracy between treatments, and the difficulty of the problem moderates the resulting accuracy.** *This result is consistent with the result of the baseline experiment*, although the differences identified in the aggregation of all the replications are more significant because statistical power is higher. The small effect size for method (0.38) is similar to the effect size resulting from the aggregation of object replications for complex and easy problems.

6.6. Aggregation of Strict Replications + Object Replications + Baseline Experiment Splitting by the Traditional Method Used

Following the study of the baseline experiment, this analysis pools together all the experiments within the family of experiments filtering the subjects by the traditional method used in the control treatment: model-based (subjects that in the traditional method drew any conceptual model before coding) and code-centric development (subjects that in the traditional method wrote the code without drawing any conceptual model). Each subject applied model-based or code-centric depending on her/his preferences.

We did not perform this analysis in each replication separately in order to preserve statistical power. However, when we aggregated all replications and the baseline experiment, we had a large enough sample size to perform a post-hoc analysis of this variable. There were 22 units that worked with model-based and 30 units that worked with code-centric development methods. Note that this is a post-hoc analysis and cannot detect causality since our study was not designed for this purpose. Therefore, the type of traditional development was not allocated randomly.

Appendix H-Fig. 31, Fig. 32, Fig. 33, Fig. 34 show histograms and box-and-whisker plots. Table 11 shows the p-values. There are significant results for the method factor where there is a small effect size (0.36) for the improvement of MDD over model-based development and a moderate (0.4) effect size if MDD is compared with code-centric development.

The results for the difficulty moderator variable are significant with a big effect size (2.79 and 1.68) in favour of MDD compared with both model-based and code-centric development. Method*problem and method* difficulty interactions do not achieve a significant result when MDD is compared against model-based development but the differences are significant if MDD is compared against code-centric development. Accuracy for the Invoice Problem is better than for the Photography Problem.

Table 11. P-values for strict replication + object replications + baseline experiment differentiating between problem complexity and filtering by model-based and code-centric development methods

Rep.	E.U.	Factor		Value
Model-Based				
OR1+ OR2+ OR3+ SR1+ SR2+ SR3+ Base.	22	Method	p-value	0.02
			effect size	0.36
			meanTraditional	54.3
			meanMDD	66.9
		Method*Problem	p-value	0.28
		Difficulty	p-value	0.00
			effect size	2.79
			meanEasy	91.4
			meanDifficulty	35
		Method*Difficulty	p-value	0.35
		Difficulty*Problem	p-value	0.27
Code-Centric				
OR1+ OR2+ OR3+ SR1+ SR2+ SR3+ Base.	30	Method	p-value	0.00
			effect size	0.4
			meanTraditional	66.7
			meanMDD	78.2
		Method*Problem	p-value	0.01
		Problem	p-value	0.09
		Difficulty	p-value	0.00
			effect size	1.68
			meanEasy	85.9
			meanDifficulty	45.7
		Method*Difficulty	p-value	0.00
		Difficulty*Problem	p-value	0.62

Appendix H-Fig. 35 shows the profile plot highlighting the differences between problems that are more evident for MDD. Accuracy is better for easy problems. Appendix H-Fig. 36 shows the profile plot. Differences between difficulties for accuracy are more evident for code-centric development.

We conclude that for **both model-based and code-centric development the differences are significant when compared with each other, where MDD yields better values for accuracy**. **Another result that model-based and code-centric development has in common is that there are significant differences for difficulty and method*difficulty**. *This contradicts the result for the baseline experiment where we did not get significant differences for either model-based or code-centric development*. The baseline experiment and strict replications yield better values for accuracy than object replications using MDD. This result makes sense since it was possible to solve the problems set in the baseline experiment and strict replications within the time scheduled for the experiment, whereas the problems set in the object replications were too large. The only difference between model-based and code-centric development is that **code-centric development**

yields significant results for problem and method*problem. The Invoice Problem is slightly easier than the Photography Problem since it does not require inheritance between classes. This difference is more evident when developers use code-centric development.

6.7. Interpretation of the Results

We can state that MDD obtains better values than a traditional method, and these differences are more evident when problem difficulty and time pressure increase. When we aggregate replications and we increase the power, differences between treatments appear independently of the problem difficulty and the time pressure. We also conclude that the use of a model-based or code-centric paradigm in the traditional method does not involve changes in the results. A summary of the results is in Appendix I.

We analyzed time pressure and difficulty as moderator variables. In our design, the two variables are related to each other since a low time pressure is consistent with an easy difficulty and vice versa. Therefore, we were unable to check all the combinations of both variables to study their effect on the factor. Even so, MDD appears to work better when both time pressure and difficulty are high. Note that the data aggregation results in 52 experimental units. This is equivalent to a large statistical power according to G*Power [15]. According to the results of the aggregation, we reject the null hypothesis H_{01} and accept that there are differences between MDD and traditional methods in terms of quality.

7. Threats to Validity

This section discusses the threats to validity that could affect the family of experiments. We have focused on the threats that appear when we replicate the experiment and when we aggregate results to improve the statistical power. Threats to the experimental design and how each experiment replication is run are not discussed in this paper since they are described in detail in the report on the baseline experiment [38]. In the following we describe the threats according to Wohlin's classification [50]. For each group of threats, we made a distinction between threats that we were unable to address, threats whose effect we managed to minimize, and threats that we solved.

Conclusion validity. This threat is concerned with issues that affect the ability to draw the correct conclusions about relations between the treatment and the outcome. Threats of this type to some replications are:

- *Low statistical power*: Replications of SR have fewer units than replications of OR, leading to lower statistical power. Low statistical power might result in null hypotheses being accepted when they are false. Moreover, in the case of SR2 (Table 3), the number of units is so low that the statistical test cannot be run.
- *Fishing*: This threat materializes when experimenters are looking for a definite result. The experimenters used in both replication types were MDD trainers. This could result in some level of researcher bias.

A threat that we have minimized in the replications is the *Reliability of treatment implementation*. This threat might materialize when treatments are taught (and then applied by experiment participants) by different experimenters. We tried to minimize this threat by using the same instruments in all replications and organizing meetings among all experimenters.

The threat that we avoided in the replications is *Random heterogeneity of subjects.* This threat materializes when subjects are heterogeneous. This could mean that differences between subjects are greater than differences between treatments. Since replications were run in several years, subjects might have a different profile in each replication. We solved this threat by recruiting subjects with similar profiles in all replications (software developers that are studying for a master's degree). Moreover, we used demographic questionnaires to detect differences between the profiles of subjects recruited in each replication. There were no differences with respect to experience with MDD and in industry.

Internal validity. This threat is concerned with influences that may affect the dependent variable with respect to causality of which researchers are unaware. There are two threats that we have tried to minimize:

- *History*: This threat materializes when treatments are applied at different times. This may lead to the experiment being run in different contexts. Our replications are open to this threat since each university ran a different replication every year. We tried to mitigate this threat using the same experimenters in each type of replication sharing the same materials across all replications.
- *Interactions with selection*: This threat materializes when there are different groups of subjects, and each group behaves differently from the others. Depending on the profile of the subjects recruited for each replication, subjects in some replications might have learned MDD better than others or might have more knowledge of traditional development. Different levels of expertise with respect to MDD or traditional methods in each replication may affect the results of the aggregation. We tried to minimize this threat by using the same training problems in all the replications. Even though we can guarantee that all subjects had a minimum knowledge of MDD and traditional methods before applying the treatments, we cannot be sure that the subjects' skills are the same across all the replications.

Construct validity. This threat is concerned with generalizing the results of the experiment to the concept or theory behind the experiment. The threat posed to the replications is *Mono-operation bias.* This threat materializes when there is only one factor, which may underrepresent the construct and thus not give the full picture of the theory. This is a potential threat when we aggregate the data of the strict replications and object replications using only easy problems. However, the other aggregations offset this threat. In this case where the analysis focuses exclusively on the method factor, there is no moderator variable to analyze the differences between the replication type, since we only considered the easy part of the prob-

lems. There are some other differences between the replication type, apart from the data: subjects were exclusively exposed to easy problems in strict replications and complex problems in object replications. Subjects participating in both types of replications had the same period of time to complete the tasks (approximately four hours). Therefore, subjects of strict replications took approximately four hours to complete easy problems and subjects of object replications spent the same amount of time on complex problems (i.e., the time taken to solve easy problems in object replications was shorter).

External validity. This threat is concerned with conditions that limit our ability to generalize the results of our experiments to industrial practice. The threat posed to the family of experiments is the *Interaction of settings and treatments.* Results are applicable only to subjects with little experience in industry (we conducted the experiment with students) considering toy experimental problems such as Photography and Invoice. More replications would have to be conducted with other types of subjects and problems to study whether the results of our family hold in other settings. Other threat that may suffer our family of experiments is *Confounding factors*. This happens when it is impossible to distinguish the effects of two factors from each other. Results may be affected also by other factors that were not considered in our analysis, such as, tools, subjects' profile, the experience of participants or teacher, among others.

Other threat suffered in all the aggregations is *Generalization of results*. Experimental objects of the replications are based on the development of a Web application; there is no development in other platform. This reduces the generalization of the results since the effort in a traditional development method may be different for other platforms, such as mobile systems. The threat to the aggregation of the replications is *Interaction of history and treatment*. This threat materializes when each replication is conducted on a different day, and the context of that day is likely to affect the results. Since each replication was conducted in a different year and each replication lasted for several weeks, we cannot be sure that the results would be unaffected by the context of the day on which the treatments were applied. Issues such as boredom, personal problems, stress, etc., can have a bigger or smaller impact on the results depending on the day when the treatment is applied.

8. CONCLUSIONS

This paper presents a family of experiments replicating a baseline experiment to analyze the quality of systems developed using MDD against traditional software development methods. The goal of the replication is twofold: (1) check whether the results of the baseline experiment hold; (2) further elaborate upon the preliminary finding of the baseline experiment that MDD has a bigger effect on quality when problem complexity increases. To do this, we defined two types of replications: strict replications, which should be as similar as possible to the baseline experiment, and object replications, in which we increased problem complexity. We ran three strict replications at Universidad Diego Portales in Chile and three object replications at Universidad Politécnica de Valencia in Spain.

The results of the replications answer our research question (RQM) confirming the results of the baseline experiment:

- **MDD yields better values for accuracy** than traditional development methods. In the baseline experiment, these differences were observed in the descriptive data, but the sample size was so small that we were unable to detect significant results. The replications addressed these shortcomings, and we were able to analyze 52 units by aggregating the data, avoiding type II error (failure to detect a difference when there is one).
- **Differences between MDD and the control are bigger when the problems to be solved are more complex** since results are significant and the effect size with complex problems is moderate even with low statistical power. The new results are consistent with the findings of the baseline experiment insofar as the differences between treatments are independent of the traditional development technique (model-based or code-centric) used.

We can extract two recommendations from the results. The first is that, to exploit the strengths of MDD, it should be applied to complex problems. The second is that MDD is more suitable when developers have previous knowledge of the MDD tool or have enough time to learn the technique. During the experiment, we found that only a few subjects knew anything about MDD before they took the course. In our setting, participants were not novice users of traditional methods, but they were new to MDD. Moreover, the teacher spent 12 hours training subjects on the use of the MDD tool and six hours solving a training problem. Therefore, the MDD learning period is not negligible. Note that subjects are knowledgeable enough to develop a system by the end of the learning period, but are not yet experts at the technique.

ACKNOWLEDGEMENTS

This work was developed with the support of the Spanish Ministry of Science and Innovation project DataMe (TIN2016-80811-P), TIN2014-60490-P and was co-financed by ERDF. It also has the support of Generalitat Valenciana with GISPRO project (PROMETEO/2018/176).

9. REFERENCES

[1] S. Abrahao, C. Gravino, E. Insfran, G. Scanniello, and G. Tortora, "Assessing the Effectiveness of Sequence Diagrams in the Comprehension of Functional Requirements: Results from a Family of Five Experiments," *Software Engineering, IEEE Transactions on,* vol. 39, pp. 327-342, 2013.

[2] Ö. Albayrak and J. C. Carver, "Investigation of individual factors impacting the effectiveness of requirements inspections: a replicated experiment," *Empirical software engineering,* vol. 19, pp. 241-266, 2012.

[3] P. Baker, S. Loh, and F. Weil, "Model-driven engineering in a large industrial context - motorola case study," in *8th International Conference of Model Driven Engineering Languages and Systems (MoDELS)*, Montego Bay, Jamaica, pp. 476–491. 2005

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

[5] S. Biffl, P. Grünbacher, and M. Halling, "A family of experiments to investigate the effects of groupware for software inspection," *Automated Software Engineering,* vol. 13, pp. 373-394, 2006/07/01 2006.

[6] M. Brambilla, J. Cabot, and M. Wimmer, *Model-Driven Software Engineering in Practice (Synthesis Lectures on Software Engineering)*, first ed.: Morgan & Claypool Publishers, 2012.

[7] C. Bunse, H.-G. Gross, and C. Peper, "Embedded System Construction --- Evaluation of Model-Driven and Component-Based Development Approaches," in *Models in Software Engineering*, R. C. Michel, Ed., ed: Springer-Verlag, pp. 66-77, 2009.

[8] G. Canfora, F. Garc\, \#237, M. Piattini, F. Ruiz, and C. A. Visaggio, "A family of experiments to validate metrics for software process models," *J. Syst. Softw.*, vol. 77, pp. 113-129, 2005.

[9] L. Cohen, *Statistical power analysis for the behavioral sciences*, 2nd. Edition ed.: Lawrence Earlbaum Associates, 1988.

[10] H. Cooper and E. A. Patall, "The relative benefits of meta-analysis conducted with individual participant data versus aggregated data," *Psychol Methods,* vol. 14, pp. 165-76, Jun 2009.

[11] J. 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,* vol. 14, pp. 685-719, 2009/12/01 2009.

[12] F. B. da Silva, M. Suassuna, A. C. França, A. Grubb, T. Gouveia, C. F. Monteiro, and I. dos Santos, "Replication of empirical studies in software engineering research: a systematic mapping study," *Empirical software engineering,* vol. 19, pp. 501-557, 2014/06/01 2014.

[13] O. Dieste, E. Fernández, R. García, and N. Juristo, "Hidden Evidence Behind Useless Replications," in *1st International Workshop on Replication in Empirical Software Engineering Research (Workshop in ICSE)*, Cape Town (South Africa), pp. 1-8. 2010

[14] D. W. Embley, S. Liddle, and Ó. Pastor, "Conceptual-Model Programming: A Manifesto," in *Handbook of Conceptual Modeling*, ed: Springer, pp. 3-16, 2011.

[15] F. Faul, E. Erdfelder, A.-G. Lang, and A. Buchner, "G*Power 3: A flexible statistical power analysis program for the social, behavioral, and biomedical sciences," *Behavior Research Methods,* vol. 39, pp. 175-191, 2007/05/01 2007.

[16] A. M. Fernández-Sáez, M. Genero, M. R. V. Chaudron, D. Caivano, and I. Ramos, "Are Forward Designed or Reverse-Engineered UML diagrams more helpful for code maintenance?: A family of experiments," *Information and Software Technology,* vol. 57, pp. 644-663, 2015.

[17] D. Fucci and B. Turhan, "On the role of tests in test-driven development: a differentiated and partial replication," *Empirical software engineering,* vol. 19, pp. 277-302, 2014/04/01 2014.

[18] G. V. Glass, "Primary, Secondary, and Meta-Analysis of Research," *Educational Researcher,* vol. 5, pp. 3-8, 1976.

[19] M. Gómez and S. Acuña, "A replicated quasi-experimental study on the influence of personality and team climate in software development," *Empirical software engineering,* vol. 19, pp. 343-377, 2014/04/01 2014.

[20] O. S. Gómez, N. Juristo, and S. Vegas, "Understanding replication of experiments in software engineering: A classification," *Information and Software Technology,* vol. 56, pp. 1033-1048, 2014.

[21] J. Gonzalez-Huerta, E. Insfran, S. Abrahão, and G. Scanniello, "Validating a model-driven software architecture evaluation and improvement method: A family of experiments," *Information and Software Technology,* vol. 57, pp. 405-429, 2015.

[22] I. Hadar, I. Reinhartz-Berger, T. Kuflik, A. Perini, F. Ricca, and A. Susi, "Comparing the comprehensibility of requirements models expressed in Use Case and Tropos: Results from a family of experiments," *Information and Software Technology,* vol. 55, pp. 1823-1843, 2013.

[23] B. Hailpern, Tarr, P., "Model-Driven Development: the Good, the Bad, and the Ugly," *IBM Syst. J.*, vol. 45, pp. 451-461, 2006.

[24] J. E. Hunter, F. L. Schmidt, and G. B. Jackson, *Meta-analysis: cumulating research findings across studies*: American Psychological Association. Division of Industrial-Organizational Psychology, 1982.

[25] IEEE, *IEEE standard computer dictionary. A compilation of IEEE standard computer glossaries.* Institute of Electrical and Electronics Engineers. New York, EE.UU., 1991.

[26] INTEGRANOVA, "INTEGRANOVA Technologies: http://www.integranova.com," ed.

[27] Iso/iec, "ISO/IEC 25000 - Software engineering - Software product Quality Requirements and Evaluation (SQuaRE) - Guide to SQuaRE," 2005.

[28] José, A. M. Santos, and M. G. d. Mendonça, "Exploring decision drivers on god class detection in three controlled experiments," presented at the Proceedings of the 30th Annual ACM Symposium on Applied Computing, Salamanca, Spain, pp. 1472-1479, 2015.

[29] N. Juristo and A. Moreno, *Basics of Software Engineering Experimentation*: Springer, 2001.

[30] N. Juristo and S. Vegas, "The role of non-exact

replications in software engineering experiments," *Empirical software engineering,* vol. 16, pp. 295-324, 2011.

[31] T. Kapteijns, S. Jansen, S. Brinkkemper, H. Houët, and R. Barendse, "A Comparative Case Study of Model Driven Development vs Traditional Development: The Tortoise or the Hare," presented at the Proc. of 4th European Workshop on From code centric to model centric software engineering: Practices, Implications and ROI (C2M), Enschede, The Netherlands, pp. 22-33, 2009.

[32] R. O. Kuehl., *Design of Experiments: Statistical Principles of Research Design and Analysis*: Second Edition. Brooks/ColeCengage Learning. , 2000.

[33] C. V. C. d. Magalhaes, F. Q. B. d. Silva, and R. E. S. Santos, "Investigations about replication of empirical studies in software engineering: preliminary findings from a mapping study," presented at the Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering, London, England, United Kingdom, pp. 1-10, 2014.

[34] Y. Martínez, C. Cachero, and S. Meliá, "MDD vs. traditional software development: A practitioner's subjective perspective," *Information and Software Technology,* vol. 55, pp. 189-200, 2013.

[35] T. O. Meservy, "Transforming software development : an MDA road map " *IEEE Computer,* vol. 38, 2005.

[36] L. S. Meyers, *Applied multivariate research : design and interpretation / Lawrence S. Meyers, Glenn Gamst, A.J. Guarino*. Thousand Oaks: SAGE Publications, 2006.

[37] D. L. Moody, "The method evaluation model: a theoretical model for validating information systems design methods," presented at the European Conference on Information Systems (ECIS 03), Naples, Italy pp. 1327-1336, 2003.

[38] J. I. Panach, S. España, Ó. Dieste, Ó. Pastor, and N. Juristo, "In search of evidence for model-driven development claims: An experiment on quality, effort, productivity and satisfaction," *Information and Software Technology,* vol. 62, pp. 164-186, 2015.

[39] N. Parkinson, *Parkinson's Law and Other Studies in Administration*. Boston: Houghton Mifflin Cornpony, 1957.

[40] R. D. Riley, P. C. Lambert, and G. Abo-Zaid, "Meta-analysis of individual participant data: rationale, conduct, and reporting," *BMJ,* vol. 340, 2010-02-05 13:38:57 2010.

[41] P. Runeson, A. Stefik, and A. Andrews, "Variation factors in the design and analysis of replicated controlled experiments," *Empirical software engineering,* vol. 19, pp. 1781-1808, 2013.

[42] G. Scanniello and U. Erra, "Distributed modeling of use case diagrams with a method based on think-pair-square: Results from two controlled experiments," *J. Vis. Lang. Comput.,* vol. 25, pp. 494-517, 2014.

[43] G. Scanniello, C. Gravino, M. Genero, J. A. Cruz-Lemus, and G. Tortora, "On the impact of UML analysis models on source-code comprehensibility and modifiability," *ACM Trans. Softw. Eng. Methodol.,* vol. 23, pp. 1-26, 2014.

[44] G. Scanniello, C. Gravino, M. Risi, G. Tortora, and G. Dodero, "Documenting Design-Pattern Instances: A Family of Experiments on Source-Code Comprehensibility," *ACM Trans. Softw. Eng. Methodol.,* vol. 24, pp. 1-35, 2015.

[45] P. Sfetsos, P. Adamidis, L. Angelis, I. Stamelos, and I. Deligiannis, "Investigating the Impact of Personality and Temperament Traits on Pair Programming: A Controlled Experiment Replication," in *Quality of Information and Communications Technology (QUATIC), 2012 Eighth International Conference on the*, pp. 57-65. 2012

[46] M. Shahin, P. Liang, and Z. Li, "Do architectural design decisions improve the understanding of software architecture? two controlled experiments," presented at the Proceedings of the 22nd International Conference on Program Comprehension, Hyderabad, India, pp. 3-13, 2014.

[47] M. Shepperd, "Replication studies considered harmful," presented at the ICSE - Track New Ideas and Emerging Results (NIER 2018), pp. 73-76, 2018.

[48] B. T. West, K. B. Welch, and A. T. Galecki, *Linear mixed models: a practical guide using statistical software*: CRC Press, 2014.

[49] R. Wieringa, *Design Science Methodology for Information Systems and Software Engineering*: Springer, 2014.

[50] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén, *Experimentation in Software Engineering: An Introduction*: Springer, 2012.

Jose Ignacio Panach is assistant professor at Universitat de València from 2011 and Assistant Researcher at Centro de Investigacion en Metodos de Produccion de Software (ProS) at the Universidad Politécnica de Valencia from 2005. Jose Ignacio holds a PhD in Computer Science (UPV 2010). His research activities focus on MDD, usability, and interaction modelling.

Oscar Dieste received his PhD from the University of Castilla-La Mancha. He is a researcher with the UPM's School of Computer Engineering. He was previously with the University of Colorado, the Complutense University of Madrid, and the Alfonso X el Sabio University. His research interests include empirical software engineering and requirements engineering.

Beatriz Marín is associate professor at Universidad Diego Portales (UDP) from 2012. She completed her PhD in Computer Science at Universitat Politècnica de València (UPV) at 2011. Her main research areas are MDD, quality of conceptual models, software testing, functional size measurement, and empirical software engineering.

Sergio España is assistant professor in Utrecht University. PhD degree in Computer Science from Universitat Politècnica de València. Author of over 70 scientific publications on requirements engineering and conceptual modelling. He works on how enterprise modelling and ICT can support responsible enterprises willing to increasingly improve their socio-environmental impact.

Sira Vegas received the PhD degree from the Universidad Politécnica de Madrid (UPM) in 2002. She is currently associate professor of software engineering at UPM. Her main research interests include experimental software engineering and software testing.

Oscar Pastor is Professor and Director of the Centro de Investigación en Métodos de Producciónde Software –ProS of the Universitat Politécnica de València. He got his PhD in 1992. Formerly he was a researcher in HP Labs, Bristol, UK. Research activities focus on web engineering, requirements engineering, information systems and MDD.

Natalia Juristo received the PhD degree from the Universidad Politecnica de Madrid (UPM) in 1991. She is currently a full professor of software engineering at UPM. She received a Finland Distinguished Professor Program (FiDiPro) professorship, starting in January 2013. Her main research interests include experimental software engineering, requirements, and testing.