Comparing a Method based on the Think-Pair-Square Method with a Brain Storming Session on High-Level Functional Modeling: Results from Two Controlled Experiments

Giuseppe Scanniello, Ugo Erra, Ana Portnova

Dipartimento di Matematica e Informatica

University of Basilicata, Viale dell'Ateneo, I-85100, Italy

Abstract

Context: Some of the challenges related to the development in geographically distributed settings are due to the fact that the greater part of the available approaches and methods try to apply well established and consolidated practices of the traditional software engineering field.
Objective: In this paper, we propose a new cooperative problem solving method for the specification of functional requirements in the context of global software development. It is based on think-pair-square, a widely used cooperative method for active problem solving. The method has been then implemented in an integrated environment for the synchronous communication and modeling.
Method: The validity of the developed technology (i.e., the method and the supporting environment) has been assessed through two controlled experiments. In particular, the experiments have been conducted to compare the developed technology with a brainstorming session based on face-to-face interaction. The comparison has been performed with respect to both the time needed to produce the specification of functional requirements and their quality.
Results: The data analysis indicates a significant difference in favor of the brainstorming session for the time, with no significant impact on the requirements specification. The effect of the familiarity of the subjects with the

Email address: giuseppe.scanniello@unibas.it (Giuseppe Scanniello)

developed technology has been investigated as well. The results reveal a significant effect on the time spent to specify software requirements (more familiar subjects spent less time), with no significant impact on their quality. **Conclusion:** The results of the evaluation mainly suggest that our proposal can be considered attractive for academy and industry community. In fact, in case the time distance is not an issue, but moving people might be a problem, the proposed distributed cooperative method could represent a viable solution with respect to a traditional brain storming session in the specification of functional requirements. Another remarkable finding is that subjects not perfectly trained may benefit from the use of our method and tool. This indicates that training phases on our technology could be shortened or skipped to reduce project costs and and failure risks.

Keywords:
Experiment, Distributed Software Development, Requirements Specification

1. INTRODUCTION

Splitting the development of a software product (e.g., system or service) among globally distributed sites is increasingly becoming a common practice in the software industry [1]. Academy and industry community refers to this relevant phenomenon as global, distributed, or multi-site software development [2]. In the recent years, an increasing interest has been devoted to this field since it offers several benefits for software organizations, such as working cost reduction, enhanced availability of skilled development staff, proximity to the market, and flexibility and efficiency for in-house staff usage to adapt quickly to volatile business needs [2]. One of the drawbacks of this practice is that very often it creates challenges due to the impact of temporal, geographical, and socio-cultural differences [1, 3].

Both in the traditional and global software development contexts, requirements engineering aims at defining the features that the system must have (i.e., functional requirements) or constraints (i.e., quality or pseudo requirements) that it must satisfy to be accepted by the customer. The most challenging phase of the requirements engineering is the requirements elicitation [4] since it is mainly concerned with communication, cooperation, and problem-solving tasks. Hence, it seems reasonable to apply cooperation active problem solving methods, such as think-pair-square [5, 6].

In this work, we have modified the original definition of the think-pair-square method in order to make it suitable for the global software engineering and for the specification of functional requirements in a distributed requirements engineering process, in particular. A prototype of a supporting tool has been also proposed here.

In order to make the developed technology (i.e., the proposed method and the supporting tool) attractive in academy and industry community an empirical assessment has been conducted as well. To this end, we have conducted two controlled experiments to compare the developed technology with the use of a brainstorming session based on face-to-face interaction. The comparison has been performed on the subjects' performances with respect to a traditional requirements elicitation task in which they were asked to develop a part of a functional model by employing UML use case diagrams [7] (in the following we will refer to it as *high level use case diagram*). The subjects' performances have been assessed considering the time needed to accomplish the tasks. Furthermore, since the only assessment of the time could be meaningless, for a project manager interested in the adoption our technology to reduce costs and guaranty a good quality functional model, we have also took into account the overall quality of the developed models.

In the replication, we have introduced a variation in the original experimental conditions to verify whether better trained subjects benefit more from a high level of familiarity with the developed technology. To this end, the same subjects as the first experiment were involved on different requirements specification tasks.

The main contributions of the paper can be summarized as follows:

- A novel method for global software development with application to the specification of functional requirements. A prototype of a supporting software system has been also proposed to enable the communication among the team members and to specify functional requirements abstracted by using high level use case diagrams.

- An empirical evaluation to compare the method and the tool prototype with respect to a traditional brain storming session. The evaluation also aimed at assessing the effect of experience on subjects' performances when employing the method and tool on functional requirements modeling.

The remainder of the paper is organized as follows. In Section 2, we first

summarize research in the field of distributed design and then we present empirical studies on the media effect in the requirements engineering process. In Section 3, we highlight the think-pair-square method and present both the method and the supporting tool. The study definition and details on the adopted experiment design are presented in Section 4, while the results of the data analysis are presented in Section 5. The discussion of the results and threats that may affect the validity of our empirical investigation are presented in Section 6. Final remarks and future direction for our work conclude the paper.

2. Related Work

Similarly to the traditional software engineering, global software development requires that software engineers spend a large part of their time to communicate both indirectly, by means of software artifacts [8, 9, 10] and directly, through meetings and informal conversations [11]. In the following subsections, we first present tools and environments for supporting distributed and cooperative design of software artifacts and then empirical studies on the effect of communication media in the requirements engineering process are discussed.

2.1. Distributed and Cooperative Modeling

Computer supported cooperative work research is focused on supporting cooperation in many applications including distributed software engineering and modeling [12, 3]. This resulted in the development of a number of distributed synchronous and asynchronous visual modeling environments that enable geographically dispersed developers to edit and discuss about the same software model. In the literature, a large number of these tools have been proposed and analyzed [13, 14, 15, 16, 17, 18]. Accordingly, we cannot be exhaustive and then only widely known modeling environments have been presented. Dewan and Riedl in [13] describe an environment to support concurrent software engineering. The authors also present the results of a study to concurrent software engineering applied to different software development tasks. The study shows that this kind of development increases the effectiveness of teams working together. From a different perspective, an integrated environment to provide multiple textual and graphical views for constructing programs in an object-oriented language is presented in [19]. In [14] is

presented an extension of this environment to support the synchronous and asynchronous object oriented development in cooperative fashion.

Many UML tools provide some functionality for cooperative modeling. For instance, Rosetta [2] is a light-weight tool for web-based development of UML diagrams. Boulila [20] focuses on the specific problem of distributed brainstorming and the construction of UML models of software through successive distributed teamwork meetings. In [16] is presented a synchronous visual tool for cooperative modeling of UML diagrams. This tool also provides a communication infrastructure that enables the concurrent editing of the same diagram at the same time. Change and configuration management functionalities for both the diagrams and graphical objects are provided. The main difference with respect to our tool is that it does not support the method think-pair-square. Furthermore, any empirical study is performed to assess the effectiveness of the development modeling environment. The same authors in [17] present COMOVER, a tool for asynchronous modeling based on UML diagrams. This tool is also integrated in an artifact management system to enable the reuse of model elements across a software project.

In the literature approaches and tools for synchronously modeling UML diagrams based on E-whiteboard are also available [15, 21]. These have pen-based sketching interface to support cooperative design, including free-hand annotations and hand-written text.

2.2. The Effect of Communication Media on Requirements Engineering Tasks

A typical and meaningful example, where the communication plays a relevant role, both in the traditional and distributed software engineering, is the requirements engineering process [11, 22]. The effect of different communication media in the requirements engineering process has been marginally investigated in the past. For example, Damian *et al.* [11] show an empirical study to compare five physical group configurations: one face-to-face and four distributed. In case of distributed settings different locations of the stakeholders were considered. The stakeholders' communication was based on computer-conference communication. This study indicates that the highest group performance occurred when clients are separated from each other and collocated with the system analyst.

A study on the effect of using a synchronous unstructured text-based communication in distributed requirements workshops is proposed in [23]. The authors compare the communication media in terms of satisfaction with performance and comfort with communication mode. Results indicate that

the satisfaction with performance elicitation is a better task/technology fit than negotiation in case text-based communication is used. Moreover, subjects found more comfortable the face-to-face communication. Several are the differences with respect to our investigation. The main difference concerns the used tasks and the fact that the interaction among distributed subjects is based on a novel method for the modeling in cooperative fashion.

Erra and Scanniello in [24] present an empirical study in the requirements negotiation. They compare a traditional face-to-face meeting, an enhanced chat, and a three dimensional virtual environment (implemented in Second Life) with respect to the time needed to negotiate software requirements and to the quality of these requirements. Results indicate that there is a difference in favor of face-to-face meeting regarding the time, while the quality is not influenced by the used media. The main difference with respect to this study regards the use and the experimentation of a novel method to cooperatively model high level use case diagrams. More recently, in [25] the authors investigated the effectiveness of preliminarily applying the distributed cooperative method proposed here for the modeling of use cases with respect to a brainstorming session using face-to-face interaction. The study revels a significant difference in terms of time needed to create use cases in favor of the face-to-face interaction with no significant impact on their quality. The quality of the use cases is measured using a checklist based approach. Several are the differences between the investigation presented in this paper and the previous conducted experiment. The most remarkable one is that the method is applied on the modeling of use case diagrams representing high level views of functional requirements. Further on, we also investigate the effect of subjects' familiarity with the method and the modeling environment.

An interesting investigation on the effect of using mixed media (i.e., rich and lean) in distributed requirements negotiations is proposed in [26]. The study is conducted with graduate, undergraduate, master, and doctorate students. Differently from us, the students used an asynchronous text-based tool and a synchronous videoconferencing based communication tool. The study reveals that the requirements negotiation is more effective when an asynchronous structured discussion is conducted before a synchronous negotiation meeting. They also observe that asynchronous communication is useful to resolve issues related to uncertainty, thus enabling the videoconferencing negotiations to only concentrate on removing ambiguities. The results presented in our study are in line with the findings presented by the authors.

In the software requirements negotiation, empirical investigations differ-

ent from controlled experiments are also available. For example, Boehm and Egyed in [27] capture and analyze requirements negotiation behavior for groups of 90 undergraduate students. The subjects use an instrumented version of the WinWin groupware system on 15 different projects. The study evidences several real problems, such as fuzzy requirements, conflicts with resources and personnel, and so on.

2.3. Differences and Remarkable Results

The greater part of the these approaches try to apply well established and consolidated practices of the traditional software engineering field to the global software development, e.g., [11, 23, 26, 28]. So far, to the best of our knowledge, our proposal is the first attempt to define and apply a special conceived method to the modeling in the global software development field. The use of the method seems promising since the subjects produced on average better quality models when accomplished the task using our method and tool. Furthermore, the familiarity of the subjects with the method and the supporting tool does not significantly affect the quality of the modeled high level use case diagrams. On the other hand, better trained subjects significantly spent less time to accomplish the modeling tasks. Results are deeply discussed in Section 6.1.

3. The Proposed Method

We propose here a new method for global software development with application to the requirements elicitation field. The method is a customization of the think-pair-square method [5, 6]. In the following subsections, we first recall the general definition of the think-pair-square method and then how it has been modified and implemented in our tool prototype (in the following we will also refer to it as communication/modeling environment).

3.1. Background

The think-pair-square method has been conceived for promoting active learning to solve problems in cooperative fashion. In fact, it gives individuals the opportunity to discuss their ideas or possible solutions for a given problem and provides means to observe problem solving strategies of the others. Individuals are grouped in homogeneous or heterogeneous way and are asked to solve a given problem through the following sequential three steps or phases:

think is individually accomplished to approach a solution for a given problem. The prompt should be limited so that each individual can really focus on the start point. This allows for wait time and helps students control the urge to impulsively shout out the first answer that comes to mind.

pair is performed in pair, who share the possible solutions individually identified in the think step. In this step students may wish to revise or alter their original ideas.

square is performed by all the individuals, who work on the common solution of the given problem. Individuals share the work made in the pair phase by the two or more pairs, thus forming a square and discussing again possible solutions for the original problem. This should allow the students to identify a shared and more comprehensive solution.

Individuals perform the pair and square phases in the same physical setting through face-to-face interaction. The rationale for performing these steps relies on the fact that if one pair of individuals is unable to solve the problem, the other pair can often explain their answer and strategy. Finally, if a suitable solution for the problem is not identified, the two pairs combine their results and generate a more comprehensive solution.

In the literature, there are some variations of the think-pair-square method to facilitate class discussions and solve problems in cooperative fashion [5]. Examples are think-pair-share and think-pair-square-share. The former replaces the square step with the share step, where students are called upon to share with the rest of the class possible solutions of the problem to provide closure to the discussion. Finally, the think-pair-square-share method is based on both the steps square and share. The square step is performed before the square one.

3.2. Distributed Think-Pair-Square and Supporting Tool

Requirements engineering aims at defining the features (i.e., functional requirements) that a system must have or constraints (i.e., quality or pseudo requirements) that it must satisfy to be accepted by the customer. The most challenging and demanding phase of the requirements engineering process is the elicitation [4]. Requirement elicitation is a non-trivial task because you can never be sure you get all requirements from the user and customer by

just asking them what the system should do. It is concerned with several activities. Among these communication, cooperation, requirements modeling, and problem-solving are the most demanding. These activities become even more difficult in case software engineers are geographically distributed.

Based on our experience [25] and on the fact that the think-pair-square method is used to promote active discussions and to solve problems, it seemed reasonable to adapt it to support distributed requirements elicitation. Indeed, we propose here a variation of the original definition of the think-pair-square method to model software requirements in cooperative and distributed fashion. With respect to the original definition of the method, we have decided of replacing the face-to-face interaction with computer-supported collaboration tools. In particular, the face-to-face interaction has been replaced with a distributed communication/modeling environment composed of a text-based chat and a synchronous use case diagram modeling tool. The chat is in charge of simulating the face-to-face interaction, while the modeling environment is to take note of the possible software requirements models (identified at each step) abstracted by using high level use case diagrams. The latter tool also enables implicit communication[1] [29] among users while performing modeling tasks.

The rationale for adapting think-pair-square with respect to think-pair-share and think-pair-square-share relies on the fact that the share step makes the latter two methods not particularly suitable for modeling tasks. In fact, it is not vital for the success of a software project to share among all the involved software engineers all the possible alternatives for software models. This could introduce useless overhead and delay the software development. Differently, the share step could be more appropriate in software engineering tasks where all the possible solutions of an issue have to be shared among the greater part of the members of a project, such as the rationale management [30].

A prototype of a supporting system has been implemented using the CoFFEE system [31]. Although this system is mainly aimed at improving cooperative learning in the context of computer support work-groups, it also offers tools to support distributed synchronous cooperative activities during discussions. In particular, it provides three main tools: *(i)* Session Editor,

[1]It is a knowledge transfer process based on communication through a shared mental or abstract model.

(ii) Controller, and *(iii)* Discusser. In the following subsections, we provide details on how we used these tools to create our communication/modeling environment.

3.2.1. The Session Editor Tool

The Session Editor tool is used to define a communication/discussion environment by means of a session that is in turns composed of steps. The activities that can be accomplished within each step are defined combining one or more basic tools. We have used here two tools: the Chat Tool (CT) and the Shared Drawing Tool (SDT). The Chat Tool is a traditional synchronous text-based chat, while SDT is a synchronous environment for modeling UML diagrams [7] (only use case and class diagrams were supported).

We have created here a communication/modeling environment, whose underlying session is composed of three steps (one for each phase of the proposed method) using the CoFFEE Session Editor (see the tab Session Details of Figure 1). In all the defined steps, the tool SDT is provided to specify the high level use case diagram of a given software system. Differently, the CT tool is included in the second and third steps to enable the communication among the users. The users employ CT to communicate and discuss with the others, while SDT was used to present a graphical overview of the functionality provided by a system in terms of actors, use cases, and dependencies between use cases and actors.

Figure 1 shows in the tab Step Details the tools used in the Pair step. The CoFFEE Session Editor also provides a Layout preview (see on the bottom right hand side of Figure 1) to highlight how the tools will be show to the users. To effectively support our method, at each step the model created by using SDT is made available to the subsequent one.

3.2.2. The Controller Tool

The Controller tool enables to execute the session behind the communication/discussion environment. The first step that a coordinator has to accomplish concerns the composition of groups of users and then he/she has to start the session . The coordinator can also constantly monitor the interactions among the users within communication/discussion environment.

The session behind our method and how it is shown within the Controller tool is depicted in Figure 2. On the top left hand side of this figure, it is shown the tab Control Panel that enables the coordinator to associate each user (e.g., RUSSO) to a team (e.g., 1) within the Think step (see on the

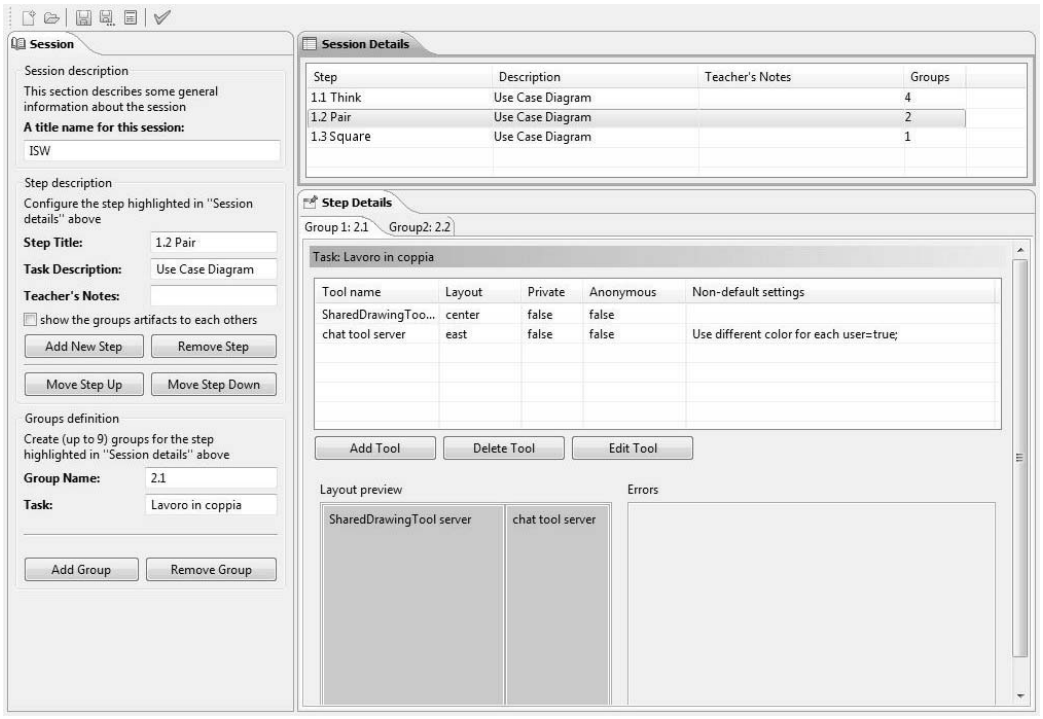

Figure 1: The CoFFEE Session Editor. .

middle of the tab SessionControl). In this case, each user is assigned to a different group since the Think step has to be individually accomplished. The team composition has to be performed also for the subsequent steps of the method. For the Pair step, the teams will be composed of two users, while a team of four will be formed in the last step. Once the composition of the teams for each step is accomplished, the session is executed pressing the button on the top of the tab SessionControl. The experimenters managed the Controller tool in the experiments to start the modeling tasks, to pass from a step to another, and to compose the teams in the Pair and Square steps. It is worth mentioning that the teams in the latter two steps of the method were randomly composed.

3.2.3. The Discusser Tool

The Discusser tool enables users exploiting the environment created by the Session Editor tool and executed by the Controlled tool. In particular,

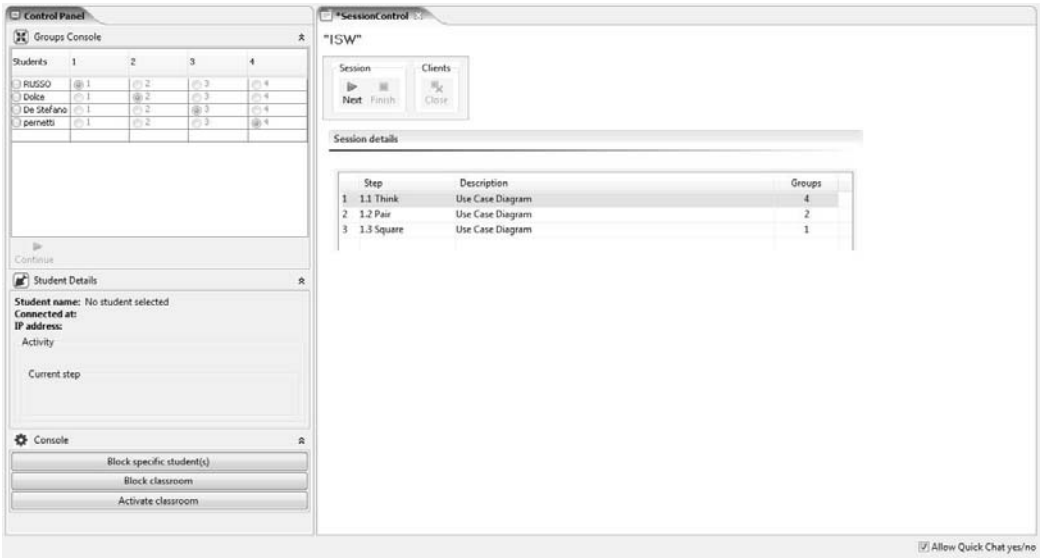

Figure 2: The CoFFEE Controller.

the communication among the users follows the session underlying the communication environment and is supported by the tools that compose this session.

Figure 3 and Figure 4 show the integrated communication/modeling environment implementing our method with respect to the Think and Square steps, respectively. Regarding the Think step, the discusser only provides features to compose a use case diagram (i.e., the SDT tool). Differently, the environment provides both the tool CT and SDT in the Pair and Square steps. The diagram produced in the previous accomplished step is made available (see on the left hand side) in the SDT tool. In the Square step the users can also brows the CT messages of the Pair step. The access to both the use case diagrams and the chat messages is possible by using the tabs on the top of the environment (see Figure 4).

4. The Experiments

In this section, we present the definition, the design, and the planning of the original experiment (*Experiment I* in the rest of the paper) and its

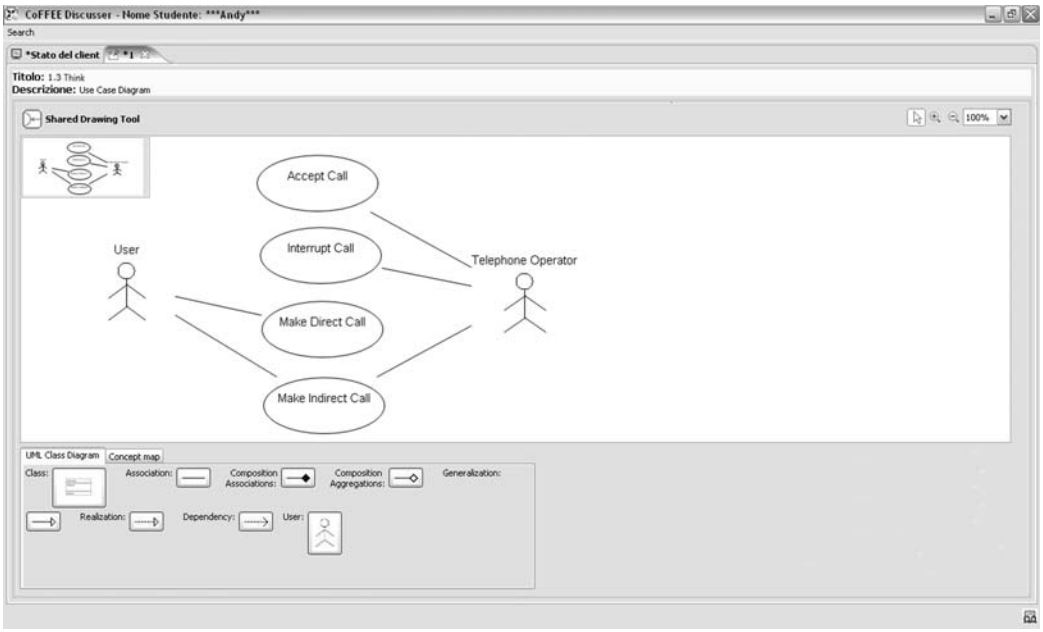

Figure 3: The CoFFEE Discusser tool for the Think step.

replication[2] (i.e., *Experiment II*), structured according to the templates suggested in [33] and [34]. For replication purposes, the experimental package and the raw data of both the experiments are available for downloading on the web[3].

4.1. Definition

The main goal of the experiments is to investigate whether our method and its implementation is effective as a traditional brainstorming session based on face-to-face interaction in the modeling of functional requirements abstracted by employing high level use case diagrams. Here and in the following, we will refer to **TPS** as our method and the tool prototype implemented in CoFFEE (see Section 3.2). On the other hand, **F2F** is used to indicate a brainstorming session based on face-to-face interaction.

[2]It is a differentiated replication [32], since we introduced a variation in the original conditions.

[3]www.scienzemfn.unisa.it/scanniello/TPS_UCDiagrams/

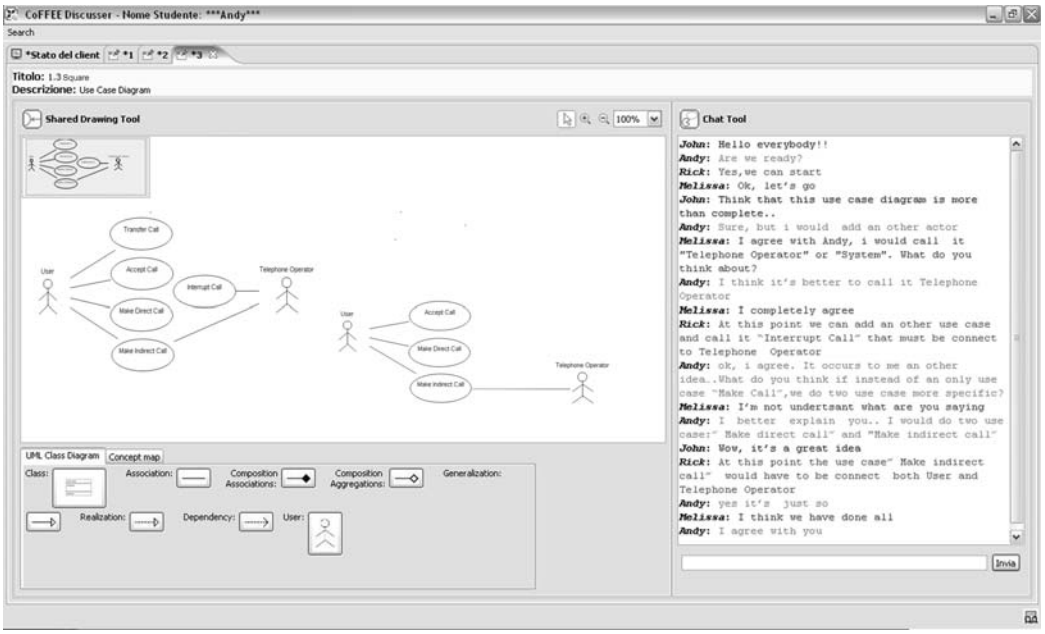

Figure 4: The CoFFEE Discusser tool for the Square step

The goal of our experimentation can be therefore defined by using the GQM (Goal Question Metric) [35] template: "*Analyze the use of TPS for the purpose of comparing it with respect F2F from the point of view of the researcher, evaluating the possibility of adopting TPS in the modeling of high level use case diagram in the context of students in Computer Science, and from the point of view of the project manager, evaluating the possibility of adopting TPS to model high level use case diagram in the context of a global software development project*".

4.2. Context

The context of the experiments was constituted of Bachelor and Master students in Computer Science at the University of Basilicata. In particular, 27 were students of a Software Engineering course of the Bachelor program, while 9 were students of a Computer Graphic course of the Master program. Before conducting the experiments, the students were asked to fill in a pre-questionnaire to get some information on the industrial working experience and grade point average. The results of this questionnaire were used

to equally distributed high and low ability subjects among the teams (see Section 4.4).

The Bachelor students were asked to accomplish the controlled experiments as a part of a series of optional laboratory exercises. As mandatory laboratory activity of the Software Engineering course, the subjects attended lectures and performed assignments about the development of an object oriented software system adopting an incremental process. Moreover, they performed a complete requirements analysis and high level design of a software system to be implemented and then they incrementally developed each subsystem of the designed system. Regarding the requirements elicitation, the subjects were asked to define functional models based on summary level, user-goal, and sub-function use cases. The execution of scheduled and unplanned meetings based on brainstorming sessions was encouraged all along the development process to solve issues and to disseminate and share information among the team members. The subjects were not graded on the performances achieved on the experiments and were asked to perform the tasks in a professional way.

The Master students have previously passed the same Software Engineering Course as the Bachelor students have been attending while carrying out this empirical investigation. Furthermore, as laboratory activity of the Computer Graphics course the subjects were asked to develop either video games (e.g., doom-like game) or three dimensional environments for software and scientific visualization. Students worked individually or in small teams. The controlled experiments presented in this paper represented an optional laboratory exercise of the course. It is also worth mentioning that the students have analysis, development and programming experience, and they are not far from junior industrial analysts since most of them have some working experience due to the internship they made in the industry as final part of their Bachelor degree.

Both Bachelor and Master students were volunteers and were aware of the pedagogical purpose of the experiment, but they did not known the experimental hypotheses. Note also that all the subjects in their academic career have used unstructured meetings and brainstorming sessions within the laboratory activities of programming and modeling, while they have never employed TPS before accomplishing Experiment I.

4.3. Hypotheses Formulation

We designed our investigation with two goals. The first goal concerned the analysis of the effect of TPS and F2F on the time needed to model a high level use case diagram (i.e., a part of a function model) starting from the problem statement[4] of the system to develop. Second, we wanted to investigate whether the use of TPS and F2F influences the quality of the produced diagrams. Accordingly, the following null hypotheses have been defined and investigated:

Hn1 The use of F2F ***does not significantly reduce*** the time needed to model a high level use case diagram with respect to TPS.

Hn2 There ***is not a significant difference*** of the use case diagram quality when using F2F or TPS.

The fist null hypothesis is one sided since we expected that subjects spent less time to accomplish the tasks using F2F. This is due to both the results presented in [24, 28] and the fact that it is intuitively clear that people using a chat tool and a shared editor needs more time to reconcile their different perspectives compared to subjects in a brainstorming session based on face-to-face interaction. The second null hypothesis is two sided since we expected that the quality of the produced high level use case diagrams is not affected by the means used to accomplish modeling tasks. In case the considered null hypotheses can be rejected with relatively high confidence, it is possible to formulate the corresponding alternative ones:

Ha1 The use of F2F ***significantly reduces*** the time needed to model a high level use case diagram with respect to TPS.

Ha2 There ***is a significant difference*** of the use case diagram quality when using F2F or TPS.

Since we are also interested in assessing whether subjects better trained on our method and tool achieve better performances, two further null hypotheses have been defined and tested:

[4]A problem statement is a document developed by the project management and the client as a mutual understanding of the problem to be addressed by the system. The problem statement describes the current situation, the functionality it should support, and environment in which the system will be deployed.

Hn3 A higher level of familiarity with TPS ***does not significantly reduce*** the time needed to model a high level use case diagram.

Hn4 A higher level of familiarity with TPS ***does not significantly decrease*** the use case diagram quality.

All these null hypotheses are one sided as we expected that better trained subjects obtained higher performances (i.e., spent less time to accomplish modeling tasks and produce higher quality models). Similarly to Hn1 and Hn2, the corresponding alternative hypotheses (i.e., Ha3 and Ha4) can be easily derived.

4.4. Design

The experiments were performed in a laboratory according to the design summarized in Table 1, which was adopted to mitigate as much as possible learning/fatigue effects. The adopted design ensured that each team worked on two different objects: *Object1* and *Object2*. For Experiment I, Object1 is *Library* (a software system to manage books and users of a library), while Object2 is *Film Collection* (a software system for the selling and the rental of films within a shop). Two different objects were selected for Experiment II. In particular, Object1 and Object2 were *Rent* (a car rental software to manage available cars, customers, and reservations) and *ECP* (an E-Commerce Platform to order CDs and books via the Internet from an on line catalogue), respectively. The software systems of Experiment I and Experiment II are similar in complexity and refer to application domains on which the subjects were sufficiently familiar with, thus reducing as much as possible guessing effect and mitigating external validity threats. Also, the complexity of the problem statements of the used software systems may be considered similar. The problem statements of the systems are available in Italian and English in the experimental package.

In each experiment the subjects were asked to accomplish 2 tasks (i.e., *Task1* and *Task2*). For each task, the subjects were asked to create a high level use case diagram using the problem statement and exploiting either TPS or F2F. The subjects had a break of half an hour between the tasks.

According to the results of the pre-questionnaire, we equally distributed high and low ability subjects among 9 teams (the interested reader can found demographic information of the subjects in the experimental package). Each team contained 3 Bachelor students and 1 Master student. The teams were

Table 1: Experiment Design

Groups	**Task1**	**Task2**
A	Object1, TPS	Object2, F2F
B	Object1, F2F	Object2, TPS

randomly assigned to the groups A and B. In particular, 5 and 4 teams were assigned to the groups A and B, respectively. The rationale for choosing 4 as size of each team relies on the fact that this size represents the smallest as possible to apply the considered distributed problem solving method.

The groups and the teams were the same for both Experiment I and Experiment II. Accordingly, we assumed that the subjects passing from Experiment I to Experiment II increased their experience and familiarity with our method and tool. This also enabled each subject to consolidate his/her working style and to develop a shared work habit with the other subjects of the team [36].

4.5. Selected Variables

The main factor is *Method* and it is a nominal variable with two possible values: F2F, TPS. Two additional variables (also named factors or co-factors) have been however considered and controlled in each experiment: *Task* ($\in \{Task1, Task2\}$) and *Object* ($\in \{Object1, Object2\}$). To analyze the effect of the subjects' familiarity with the proposed method and the developed supporting tool, we considered the performances of the subjects when they used TPS to accomplish the modeling tasks. Therefore, to test the null hypotheses Hn3 and Hn4 the *Experiment* ($\in \{ExperimentI, ExperimentII\}$) nominal variable has been considered as well.

To statistically test the formulated null hypotheses the following dependent variables were taken into account:

Time indicates the number of minutes that all the subjects within a given team spent to model a high level use case diagram;

Quality indicates the quality of a high level use case diagram.

Regarding Quality, we used an approach based on the information retrieval theory [37] in order to get a quantitative assessment of a use case diagram with respect to an oracle (i.e., the correct use case diagram) in terms of

actors/use cases (a/u) and dependencies (d) between them:

$$precision_{a/u} = \frac{|A_{a/u} \cap O_{a/u}|}{|A_{a/u}|} \quad recall_{a/u} = \frac{|A_{a/u} \cap O_{a/u}|}{|O_{a/u}|}$$

$$precision_{d} = \frac{|A_{d} \cap O_{d}|}{|A_{d}|} \quad recall_{d} = \frac{|A_{d} \cap O_{d}|}{|O_{d}|}$$

$O_{a/u}$ indicates the known correct set of expected actors/use cases that can be easily derived by an oracle. Similarly, O_d indicates the correct set of dependences among actors and use cases. Instead, $A_{a/u}$ represents the set of actors/use cases in the use case diagram modeled by a subject or a given group of subjects, while A_d is the set of dependencies between actors and use cases. Accordingly, $precision_{a/c}$ measured the correctness of both actors/use cases belonging to a given use case diagram. The correctness of the dependencies between actors and use cases was measured by using $precision_d$. On the other hand, $recall_{a/u}$ measured the completeness of a use case diagram with respect to its actors/use cases, while $recall_d$ was used to get a quantitative assessment of the completeness of a use case diagram with respect to the dependencies among actors and use cases.

Since precision and recall quantitatively summarize two different concepts, we used their harmonic mean [37] to get a balance between correctness and completeness of actors/use cases of a use case diagram:

$$F - Measure_{a/u} = \frac{2 * precision_{a/u} * recall_{a/u}}{precision_{a/u} + recall_{a/u}}$$

Similarly, we employed the harmonic mean of precision and recall to get a balance between correctness and completeness of the dependences among the objects of a use case diagram:

$$F - Measure_{d} = \frac{2 * precision_{d} * recall_{d}}{precision_{d} + recall_{d}}$$

To get a single value ($\in [0..1]$) for summarizing the overall quality of a high level use case diagram we combined $F-Measure_{a/u}$ and $F-Measure_d$ as follows:

$$Quality = \frac{F - Measure_{a/u} + F - Measure_{d}}{2}$$

Table 2: Measures obtained on the use case diagram shown in Figure 5

Measure	Value
$precision_{a/u}$	1
$recall_{a/u}$	0.857
$precision_d$	0.8
$recall_d$	0.666
$F-Measure_{a/u}$	0.923
$F-Measure_d$	0.727
$Quality$	0.825

In case Quality assumes a value close to 1, it means that the students modeled the high level use case diagram very well, while a value close to 0 indicates that the diagram is far from an oracle. This variable has been defined to give the same relevance to the correctness and completeness of a diagram with respect to both actors/use cases and dependences between them. In order to better comprehend how the defined measure works, we provide an example of its application. In particular, let the high level use case diagrams shown in Figure 3 and Figure 5 be respectively the diagram modeled by a group of subjects and an oracle, the Quality measure assumes 0.825 as value. Table 2 summarizes how this value has been obtained.

The authors developed the oracles before conducting the experiments and so without analyzing the high level use case diagrams produced by the subjects. Successively, an independent expert reviewed the oracles to detect possible defects. The obtained oracles enabled us to reduce as much as possible the computation of Quality in favor of either F2F or TPS. However, the reader may object to the fact that there could be more ways to model the use case diagram of a given software system. This may be true for tasks accomplished with both F2F and TPS. It is however worth pointing out that this issue is not so strength for the modeling of high level use case diagrams based on summary level use cases, since these diagrams aim at providing a description of the high level functionality that the system is intended to deliver.

The computation of a value for Quality may be affected by ambiguities in the associations between the use case names of an oracle and the use case names of the diagram modeled by the subjects. To effectively carry out this association the use cases of a use case diagram should be deeply analyzed and comprehended. This was not possible since the subjects were not asked to model use cases. The rationale for this design choice was inspired by the

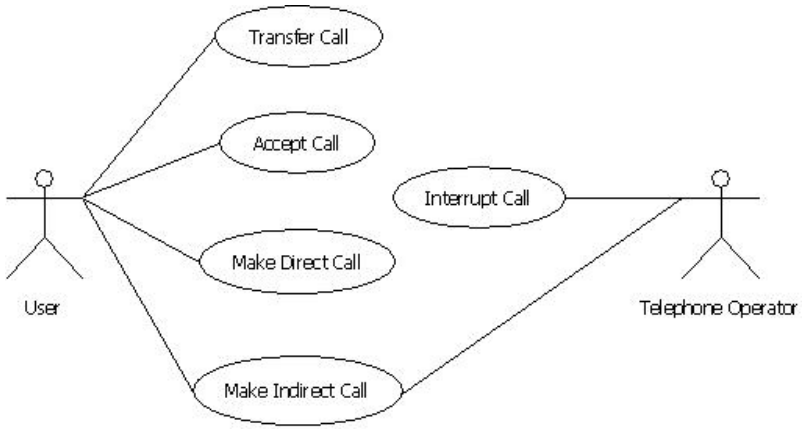

Figure 5: Oracle

need of achieving a trade-off between complexity of the tasks and fatigue to accomplish them. To control as much as possible threats related to the association between the use case names of an oracle and the use case names of the diagram modeled by the subjects, the authors conducted a meeting to solve all the possible issues and ambiguities.

4.6. Pilot, Procedure, and Preparation

Before Experiment I, we conducted a pilot experiment with four students of the Bachelor program in Computer Science at the University of Basilicata. They were volunteers and were not successively involved in the controlled experiments. The goals of the pilot were: *(i)* getting some indications on the complexity and the needed time to accomplish the tasks; *(ii)* testing the used integrated environment. The results indicated that the tasks were well suited and that one hour and a half was sufficient to accomplish them.

The subjects involved in the controlled experiments attended a training lesson. We posed great care to explain the granularity level required for the specification of use cases. In particular, we explained that we were interested in the specification of summary level use cases rather than user-goal and sub-function use cases. Instructions on the experimental procedure to follow were provided together with details on the method and the supporting tool.

At the end of each experiment, the subjects were asked to fill in a post experiment survey questionnaire (see Table 3). This survey mainly aimed at getting the perceived: *(i)* quality of the provided material; *(ii)* effort to

Table 3: Post experiment questionnaire

ID	Question
1	I had enough time to accomplish the tasks.
2	The objectives of the experiment were perfectly clear.
3	The face-to-face interaction made problematic the interaction with the other team members.
4	The modeling of a high level use case diagram using a brainstorming session was problematic.
5	I found useful the communication/modeling environment
6	The used communication/modeling environment made problematic the interaction with my colleagues.
7	The modeling of a high level use case diagram through the used communication/modeling environment is problematic.
8	I had some problems to apply the distributed cooperative problem solving method to accomplish the modeling task.
9	The use of the distributed cooperative problem solving method is too much expensive to apply.
10	The distributed problem solving method and the used communication/modeling environment allowed me to properly accomplish the modeling task.
11	The experiment was useful from the pedagogical point of view

perform the tasks and their clarity; *(iii)* satisfaction about the used technology and face-to-face based brainstorming session to accomplish the modeling tasks. All the statements expected closed answers according to a five-point Likert scale [38]: 1 (strongly agree); 2 (agree); 3 (neither agree nor disagree); 4 (disagree); 5 (strongly disagree). To avoid that subjects answered without paying attention to the statements, we formulated them in a positive and in a negative form (see for example Q2 and Q3).

Regarding the preparation of the PCs used in the experiments, we installed the CoFFEE Controller on 5 desktops (i.e., the server nodes), while we installed on the subjects' PCs the CoFFEE Discusser. All the PCs were equipped by a 3.0 GHz Intel Pentium 4 with 1 GB of RAM and Windows XP Professional SP 3 as operating system. The subjects' PCs were connected to the corresponding servers by a LAN. The size of each monitor was 15 inches with a resolution of 1024x768. The discusser was run in full-screen mode and the iconizable button was disabled. This forced the subjects to use only our tool during the experimentation.

4.7. Execution

In order to carry out each experiment, the subjects were provided with a pencil, some paper sheets, and the following hard copy material:

1. The introductory presentation of both the experiment and the communication/modeling environment;
2. The problem statements of the software systems to be used in the tasks. The problem statement of Task2 provided once the subjects accomplished the first task;
3. The post-experiment survey questionnaire to be filled in at the end of the experiment. This was furnished at the end of each experiment.

Note that the size of the text within the problem statements of the systems was nearly the same. Indeed, they differ as the problem statement of ECP (i.e., Object2 of Experiment II) included some screen mockups to clarify some functionality to implement [4]. This difference was deliberately introduced and properly controlled (see Section 6.1).

4.8. Analysis Procedure

To analyze the collected data and to test the defined null hypotheses, statistical tests have been used. In all our statistical tests, we decided (as usual) to accept a probability of 5% of committing Type-I-Error [34], i.e., of rejecting a null hypothesis when it is actually true.

We applied the non-parametric paired Wilcoxon's test [39] to verify the null hypotheses. We employed this test due to the sample sizes and mostly non-normality of the data. Furthermore, it has been applied in the literature for purposes similarly to ours [40, 41].

Other than testing the formulated hypotheses, it is of practical interest to estimate the magnitude of performance differences of the subjects when using TPS and F2F. Similarly to [41], we used the Cohen d effect size [42] to understand how a factor effects a dependent variable. Typically, effect size is 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 $|d| \geq 0.8$. Due to the experiment design, we applied the Cohen d effect size for dependent samples (to be used in the context of paired analyses). This is defined as the difference between the means of the samples (M_{TPS} and M_{F2F}) divided by the standard deviation of the (paired) differences between samples (i.e., $\sigma_{(TPS,F2F)}$):

$$d = \frac{M_{TPS} - M_{F2F}}{\sigma_{(TPS,F2F)}}$$

To assess the effect of Task on the dependent variables and to analyze its

possible interactions[5] with the main factor (i.e.,TPS or F2F), we used a two-way Analysis of Variance (ANOVA) [43]. ANOVA was also used to analyze the effect of the subjects' familiarity with the developed technology on the dependent variables. To this end, the main factor is Experiment (i.e., Experiment I and Experiment II). On the other hand, the factor to be controlled is Task. This factor has to be controlled since the complexity of the tasks used in the both the controlled experiments may affect the validity of the results. Therefore, we used ANOVA to test the effect of the main factor on the dependent variables and to analyze possible interaction between Experiment and Task. It is worth mentioning that ANOVA is used in the literature also when the sample size is small (as in our case) and without verifying its assumption [44, 41]. This is possible thanks to the test robustness.

To estimate the magnitude of performance difference of the subjects when using TPS in Experiment I and Experiment II, we employed the Cohen d effect size for dependent samples:

$$d = \frac{M_{ExperimentI} - M_{ExperimentII}}{\sigma_{(ExperimentI,ExperimentII)}}$$

where $\sigma_{(ExperimentI,ExperimentII)}$ is the standard deviation of the (paired) differences between the samples.

We also used here a visual representation widely employed in exploratory data analysis, i.e., the boxplots [45]. They summarize data using five numbers: the median, upper and lower quartiles, minimum and maximum values, and outliers. In particular, the thick horizontal line is the median, the box represents the observation between the 2^{nd} and 3^{rd} quartiles, and the whiskers are the 1^{st} and 4^{th} quartiles.

5. Results

In the following subsections, we present the results and highlight the outcomes of the post experiment survey questionnaires.

5.1. Experiment I

Some descriptive statistics (i.e., median, mean, and standard deviation) of Experiment I on the dependent variables Time and Quality (grouped by

[5]We have interaction when the combined effect of two independent variables on the dependent one is non additive.

Table 4: Descriptive Statistic on Time (in minute) for Experiment I

Method	Object1 (Library), Task1			Object2 (FilmCollection), Task2		
	Med.	Mean	Std. Dev.	Med.	Mean	Std. Dev.
All	260	285.66	43.21	148	198.22	88.76
TPS	333	302.2	48.65	304	294	22.89
F2F	260	265	21.79	120	121.6	23.54

Table 5: Descriptive Statistic on Quality for Experiment I

Method	Object1 (Library), Task1			Object2 (FilmCollection), Task2		
	Med.	Mean	Std. Dev.	Med.	Mean	Std. Dev.
All	0.78	0.78	0.10	0.73	0.76	0.11
TPS	0.75	0.75	0.13	0.76	0.77	0.05
F2F	0.82	0.82	0.05	0.73	0.75	0.14

Object and Method) are shown in Tables 4 and 5, respectively. The descriptive statistics generally show that the average time to accomplish the tasks using TPS was much more than the mean time to accomplish the same tasks with F2F. On Quality, we can observe that the subjects achieved on average slightly better scores (0.07) on Task1 using F2F, while slightly better scores (0.02) were obtained on average on Task2 using TPS.

5.1.1. Effect of Method

Table 6 summarizes the results of the Wilcoxon test on the defined null hypotheses and the values of the effect size. Furthermore, it also shows the number of teams that spent more time to accomplish the tasks with F2F (F2F>TPS). The number of teams that spent less time to accomplish the tasks with TPS (F2F<TPS) is reported as well. Similarly, we report the results on Quality.

The Wilcoxon test revealed that Hn1 (i.e., the null hypothesis used to assess the influence of Method on Time) can be rejected as a significant difference in terms of time to accomplish the tasks was present (p-value = 0.011) with a large effect size (i.e., 1.39). This result was further corroborated by the fact that 8 teams spent less time to accomplish the task when using F2F (see Table 6).

No significant difference on Quality was observed (p-value = 0.677) in case subjects used F2F and TPS (i.e., Hn2 cannot be rejected). The effect size is small as the Cohen d value is -0.34 (i.e., a small quality difference of the modelled high level use case diagram was present in favor of F2F). However, 5 teams modeled better quality diagrams using TPS (5 out of 9).

Table 6: Results of the Wilcoxon test for Experiment I

Hypotheses	p-value	Cohen d effect size	F2F>TPS	F2F<TPS
Hn1 (Time)	**YES (0.006)**	Large (1.39)	1	8
Hn2 (Quality)	NO (0.677)	Small (-0.34)	4	5

Table 7: ANOVA Results for Experiment I

Factor	Time	Quality
Method	**YES (0.000)**	NO (0.703)
Task	**YES (0.001)**	NO (0.690)
Method vs Task	**YES (0.002)**	NO (0.414)

5.1.2. Other Factors

Table 7 summarizes the results achieved by applying ANOVA on Method and Task. The results on the interaction between Method and Task are shown as well. The results of ANOVA show a positive effect of Task on Time. Differently, the effect of Task on Quality is not statistically significant. A significant interaction between Task and Method on Time is present, while the interaction on these variables is not significant on Quality.

5.2. Experiment II

The descriptive statistics of Experiment II on the two dependent variables are shown in Tables 8 and 9, respectively. These statistics are grouped by Object and Method. Similarly to Experiment I, the mean time to accomplish the tasks using TPS was much more than the mean time needed for performing the tasks with F2F. As far as Quality is concerned, the average quality of the developed high level use case diagrams is higher when subjects employed TPS. A larger difference is observable on Object2, namely 0.18 with respect to 0.01 for Object1.

Table 8: Descriptive Statistic on Time (in minute) for Experiment II

Method	Object1 (Rent), Task1			Object2 (ECP), Task2		
	Med.	Mean	Std. Dev.	Med.	Mean	Std. Dev.
All	149	126.55	43.01	140	143.55	51.65
TPS	149	154.2	18.39	195	194	21.8
F2F	74	92	39.89	112	103.2	27.52

Table 9: Descriptive Statistic on Quality for Experiment II

Method	Object1 (Rent), Task1			Object2 (ECP), Task2		
	Med.	Mean	Std. Dev.	Med.	Mean	Std. Dev.
All	0.81	0.78	0.11	0.6	0.64	0.11
TPS	0.77	0.79	0.13	0.75	0.74	0.10
F2F	0.81	0.78	0.07	0.56	0.56	0.05

Table 10: Results of the Wilcoxon test for Experiment II

Hypotheses	p-value	Cohen d effect size	F2F>TPS	F2F<TPS
Hn1 (Time)	**YES (0.001)**	Large (1.72)	1	8
Hn2 (Quality)	NO (0.173)	Medium (0.53)	3	6

5.2.1. Effect of Method

The results of the Wilcoxon test on the null hypotheses Hn1 and Hn2 and the Cohen d effect size values are reported on Table 10. Similarly to Experiment I, this table also shows the number of teams that spent less time to accomplish the tasks using F2F (F2F<TPS) and the number of teams that spent more time using TPS (F2F>TPS). Similarly, the results on Quality are shown.

The null hypothesis Hn1 can be rejected since a significant difference in terms of time to accomplish the tasks was shown by the Wilcoxon test (p-value = 0.001). The effect size of Method on Time is large (i.e., 1.7). This result was further confirmed as 8 out of 9 teams spent less time when using F2F to accomplish the modeling tasks. On the other hand, the Wilcoxon test did not enable to reject Hn2 (p-value = 0.173). The effect size is medium as the Cohen d value is 0.53 (i.e., a medium quality difference of the modelled high level use case diagram was present in favor of TPS). Moreover, as shown in Table 10, 6 out of 9 teams modeled better use case diagrams when using TPS.

5.2.2. Other Factors

The results of ANOVA on Method and Task are summarized in Table 11. This table also shows the results of the interaction between Method and Task. The results indicate a positive effect of Method on Time, thus confirming the Wilcoxon results. On the other hand, the effect of Task on Time and the interaction between Task and Method on Time are not statistically significant. The effect of Method is not statistically significant on

Table 11: ANOVA Results for Experiment II

Factor	Time	Quality
Method	**YES (0.000)**	NO (0.104)
Task	NO (0.108)	**YES (0.025)**
Method vs Task	NO (0.352)	NO (0.156)

Table 12: Results of the Wilcoxon test for Experiment I and Experiment II

Hypotheses	p-value	Cohen *d* effect size	ExperimentI>ExperimentII	ExperimentI<ExperimentII
Hn3 (Time)	**YES (0.000)**	Large (2.2)	9	0
Hn4 (Quality)	NO (0.345)	Negligible (-0.09)	3	6

Quality (confirming the results of the Wilcoxon test), while it is statistically significant on Task. ANOVA also revealed that the interaction between Task and Method on Quality is not significant.

5.3. The Effect of the Subjects' Experience on TPS

Te results of the Wilcoxon test (see Table 12) show that the null hypothesis Hn3 can be rejected (p-value = 0.000) with a large effect size (i.e., 2.2). Therefore, subjects spent significantly less time to accomplish the tasks in Experiment II. Note also that all the 9 teams spent less time to accomplish the tasks in Experiment II with respect to Experiment I. On the other hand, the results of the Wilcoxon test did not allow rejecting Hn4 (p-value = 0.345) with a negligible effect size (i.e., -0.09). However, a further analysis showed that 6 out of 9 teams modeled better quality use case diagrams in Experiment II.

The results of ANOVA on Experiment and Task and their interaction are summarized in Table 13. This test shows a positive effect of Experiment on Time, while this is not statistically significant on Quality. No significant interaction between Experiment and Task on both Time and Quality was revealed. Note that the ANOVA test confirmed the results of the Wilcoxon test on the effect of Experiment on Time and Quality.

Table 13: ANOVA Results for Experiment I and Experiment II

Factor	Time	Quality
Experiment	**YES (0.000)**	NO (0.912)
Task	NO (0.362)	NO (0.810)
Experiment vs Task	NO (0.175)	NO (0.525)

5.4. Post-Experiment Survey Questionnaires

The data collected from the post-experiment survey questionnaires of Experiment I and Experiment II are visually summarized in Figure 6 and Figure 7, respectively. The agreement level on answers of all the statements can be generally considered concordant for each experiment. Furthermore, for both the experiments the distributions of each statement, as expected, are nearly identical except for the presence of some outliers for Experiment I. Accordingly, we describe the results of the post-experiment survey questionnaires for Experiment I and Experiment II together. Differences are highlighted if present.

The analysis of the answers of the statement Q1 reveals that the time to accomplish the tasks was considered adequate, thus showing that any time constraint was perceived by the subjects. The boxplots of Q2 shows that the subjects found the objectives of the experiment clear. As shown by the boxplots of Q3, the subjects did not have any problem to conduct and manage the face-to-face interaction in brainstorming sessions. Similarly, the subjects declared that face-to-face interaction did not present any concerns in the execution of the modeling tasks (see the boxplots of Q4).

The boxplots of Q5 show that the subjects expressed a positive judgment on the usefulness of the proposed communication/modeling environment. The distribution of the answers of Q6 for Experiment I is more skewed than the distribution of the answers of Experiment II. However, both the boxplots show that the greater part of the subjects did not find problematic the interaction when using our environment to accomplish the tasks. Further on the subjects stated that they did not have any problems to model the use case diagram using the our environment (see the boxplots of Q7). The boxplots of Q8 show that the subjects of both the experiments did not have any problem to accomplish the modeling tasks with our method. The analysis of the answers of the statement Q9 indicates that the method was considered neither cheap nor expensive. The boxplots of Q10 show that the method and the supporting environment were considered appropriate to model use case diagrams. Finally, the greater part of the subjects found very useful the experiments from the pedagogical point of view (see the boxplots of Q11).

As expected, the subjects generally manifested a larger satisfaction level when using the face-to-face interaction. In particular, the interaction and the requirements specification within a brain storming session based on face-to-face interaction was considered simpler. However, a slight difference between

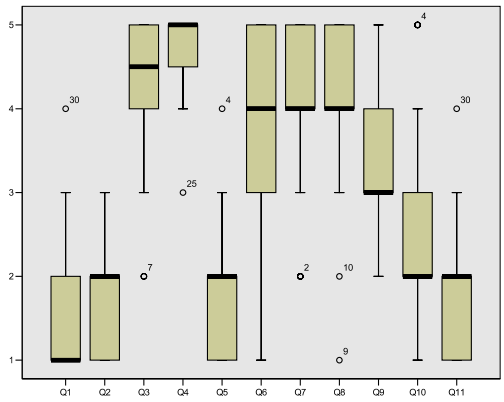

Figure 6: Boxplots of the post experiment survey questionnaire of Experiment I

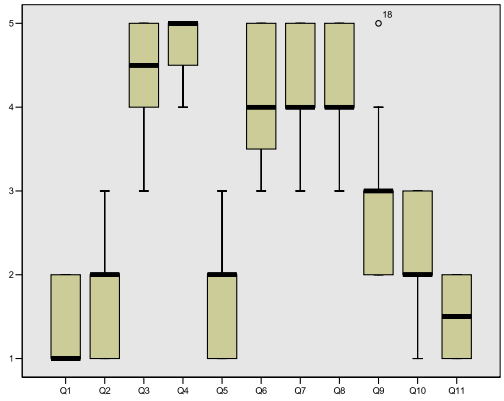

Figure 7: Boxplots of the post experiment survey questionnaire of Experiment II

the medians of the boxplots corresponding to Q3 and Q6 is observable. Similarly, the subjects found simpler the modeling of the high level use case diagrams when using the the face-to-face brainstorming session (see the boxes of Q4 and Q7).

6. Discussion and Threats to Validity

In the following we first discuss the results of the experimentation presented in this paper and then the threats that could affect the validity of these results.

6.1. Discussion

The data analysis conducted on Experiment I and Experiment II indicates a significant difference of the time needed to model high level use case diagrams using brainstorming sessions based on face-to-face interaction, with no significant impact on the quality. This has a meaningful and practical implication for the software modeling in the distributed requirements engineering process. In fact, in case the time distance is not an issue, but moving people might be a problem, the proposed distributed cooperative method could represent a viable solution.

The data analysis on Experiment I alone suggested that the members within a team need less time to accomplish the second task (i.e., Task2) when using a traditional brainstorming session (see descriptive statistics in Table 4 and consider the interaction between Method and Task). Also, this result may have practical implications. In fact, our environment may be used before a brainstorming session to reduce the effort to execute a requirements modeling task. Differently, the time needed to accomplish the second modeling tasks was not affected in case the subjects used brainstorming before. This result enables us to believe that the execution of a traditional brainstorming session is less prone to improve the cohesion among team members with respect to our method. This result is on line with the findings present in [46], but needs caution since in the literature related to computer mediated communication contrasting results are available on the effect of team cohesion when using communication media with different richness level. Accordingly, we plan to conduct special conceived investigations to increase our awareness on this point. Note that the order the subjects used the methods to accomplish the tasks did not affect the quality of the modeled use case diagrams.

Regarding Experiment II, we observed that the presence of screen mockups in the problem statement of ECP did not affect the overall quality of the modeled high level use case diagrams. This is because no interaction between Method and Task is present (see Table 11). Furthermore, the descriptive statistics of Table 9 indicate that the screen mockups did not properly supported subjects in the accomplishment of a modeling tasks using face-to-face interaction within a brainstorming session. In fact, the subjects who used first our method and tool to accomplish the task starting from a problem statement without screen mockups produced worse quality high level use case diagrams when they successively used a brainstorming session on a problem statement with screen mockups (0.77 and 0.56, respectively).

Table 14: Descriptive Statistic on the TPS steps

Variable	Step	ExperimentI			ExperimentII		
		Med.	Mean	Std. Dev.	Med.	Mean	Std. Dev.
Time	Think	30	29	3.1	15.5	15.5	4.2
	Pair	20	19.3	2.5	13	12.7	3.5
	Square	28	26	7.8	14	14.6	4.1
Quality	Think	0.6	0.578	0.162	0.584	0.584	0.19
	Pair	0.673	0.641	0.14	0.698	0.698	0.173
	Square	0.75	0.759	0.1	0.77	0.77	0.127

A slightly difference (0.81 and 0.75) in terms of use case diagram quality was observed in case the subjects accomplished the task using a brainstorming session first with screen mockups and then with our method and tool and screen mockups. These findings together with the fact that subjects achieved nearly the same results using TPS (0.77 and 0.75) with and without screen mockups suggest that the use of our technology in the modeling was slightly affected by the additional provided material. Since the creation of additional material (i.e., the design and the development of screen mockups) may have a cost [28], this result may have practical implications. Further investigations are needed to understand the effect of employing our method and tool with additional material and screen mockup, in particular.

The experimental results also indicated that the effect of the familiarity on the method and tool does not significantly affect the quality of the modeled high level use case diagram. Although we cannot conclude anything definitively as we were not able to reject the null hypothesis Hn4, a further analysis on the data indicated a possible positive effect of the subjects' familiarity (see Section 5.3) on the overall quality of the high level use case diagram modeled within the tasks. This result suggests that also subjects, who are not perfectly trained on the method and tool may benefit from their use. Also this result has a practical implication since a training phase could be shortened without affecting the overall quality of the produced software models.

A further analysis was conducted on both the experiments to investigate how the overall quality of the modeled high level use case diagrams changes across the subsequent steps of our method. In particular, Table 14 reports some descriptive statistics (i.e., median, mean, and standard deviation) on the two considered dependent variables. These statistics are grouped by experiment and the step of the method.

The results indicate that given a step of the method in the replicated

experiment the subjects spent on average less time. This result could be due to the increased familiarity of each subject with the method and the supporting tool. The statistics also shows that the average time to accomplish a step is almost the same in each experiment. Indeed, the Pair step required less time with respect to the steps Think and Square. A possible motivation is that in the Think step the subjects spent time to understand the functionality to be implemented in the system. On the other hand, in the Square step the additional time could be due to the communication among the team members to get an agreement and to model the diagram.

The descriptive statistics (median, mean, and standard deviation) on Quality indicate that the quality of the produced models increased through the subsequent steps of our method in both the experiments. In order to further investigate this point, we used boxplots to graphically represent the distribution of the Quality values obtained by the subjects in the first step, in pair at the end of the second step, and in groups of four in the last step. In particular, Figure 8 summarizes the values of Quality attained in Experiment I. The boxplots show that the quality increased through the subsequent steps of our method. The boxplots are also less skewed when passing through the three steps, thus indicating that the differences among the high level use case diagrams decrease in the latest steps (i.e., the gap between the subjects decreases when working individually, in pairs, and in groups of four). Further on we attained similar findings also considering each task alone. This trend has been also confirmed by the results of Experiment II (see Figure 9). From the methodological point of view, the positive effect of passing through the subsequent steps seems relevant. However, the reader may object to the fact this could be related to the lower number of data points of the latter two steps of the method. This could be true and so requires further and special conceived empirical investigations.

6.2. Threats to Validity

The *Internal Validity* threat is relevant in studies that try to establish a causal relationship. This threat was mitigated here by the experiment design since each group of subjects within Experiment I and Experiment II worked over two tasks, on two different objects. Even if the learning effect should be mitigated by this design, we found in Experiment I a significant effect of Task on Time (ANOVA $p = 0.001$). Moreover, we observed a significant interaction between Task and Method on Time (ANOVA $p = 0.002$). Specifically, in the second task we observed a reduction of the mean time to complete it. This

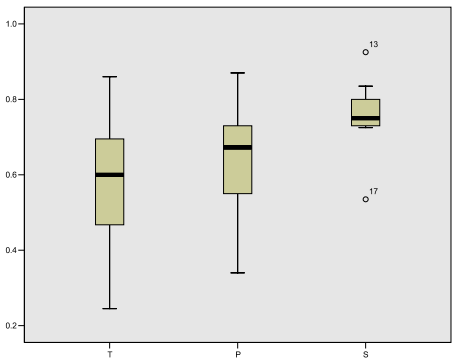

Figure 8: Quality for Experiment I

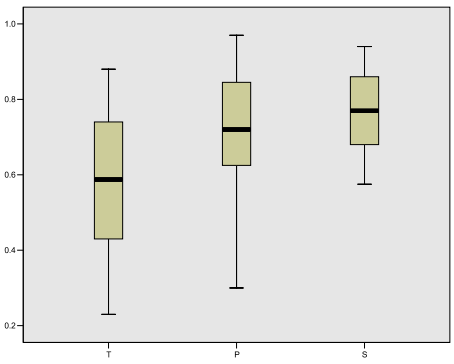

Figure 9: Quality for Experiment II

could be the consequence of the effect of the method on the cohesion of team members (see Section 6.1). On the other hand, to mitigate the fatigue effect, a break between the two tasks of each experiment was imposed to the subjects. Another issue concerns the possible information exchanged among the subjects while performing the tasks. We prevented this in several ways. Finally, to reduce internal validity, the subjects did not know exactly the hypotheses of the experiment and were not evaluated on their performances.

External Validity refers to the generalization of the results within different contexts. This threat may present when experiments are conducted with students, since they could not be representative as software professionals [47]. Indeed, students could be more comfortable than software practitioners with the developed communication/modeling environment and less experienced with brainstorming session. In this scenario, the use of students as subjects may bring out phenomena related to the learning of new communication media. In fact, they could be more prone to learn and master the tool prototype developed in our research laboratory. This is enough unusual for student experiments. However, the greater part of the involved Bachelor students will be integrated in the software industry market soon, so they will become the standard. Furthermore, the Masters students are not far from junior industrial analysts since most of them have some working experience due to the internship they made in the industry as final part of their Bachelor degree.

Even the fact that the subjects had a sufficient familiarity with the application domain of the software systems used within the tasks mitigated the external validity. Threats belonging to this category can be also related to

the tasks. Regarding the complexity of the tasks, we can argue that the functional model of the software systems the subjects were asked to model in terms of high level use case diagrams are realistic for small size projects and they are not trivial. Clearly, further investigations with larger/real objects, more demanding tasks, and more experienced subjects are needed to confirm or contradict the presented results. Further on it will be worth investigating the effect of using the defined method and the supporting tool to fully create functional models. Another possible threat to the external validity concerns the fact that in our investigation we did not involved subjects with cultural and background diversities. In fact, as described in [36] space and time are not the only issues in the global software development. Work that takes place over long distances means that communication will often involve different cultures. Accordingly, future work will be devoted to investigate how cultural and background diversities among the subjects affects subjects performances when using TPS. Regarding the external validity, it is worth mentioning that all the subjects successfully accomplished the requirements specification tasks.

Construct Validity threats concern the possibility that the relationship between cause and effect is causal. This validity was mitigated by a proper design. Also, the selection and the measurement of the dependent variables could threaten the construct validity. Quality was measured using an information retrieval based approach in order to avoid as much as possible any subjective evaluation (see Section 4.5). The values of this measure may be conditioned by the granularity level of the use cases of the modeled diagrams. This issue equally affects high level use case diagrams developed using both our method and tool and a traditional brainstorming session, thus not biasing the obtained results. Regarding the Time variable and the tasks accomplished with a brainstorming session, we asked the subjects to note down the start and stop time. These information was validated by the supervisors in each experiment. Although this may not be very accurate, this is a widely adopted way to measure performance, see for example [41, 28].

The experiment design also mitigated the *Conclusion Validity* threats, which concern relationships between the treatment and the obtained findings. In our experiments, the selection of the population may affect this experimental validity. However, the selected subjects were not far from junior developers/analysts since in each team there were students of the Bachelor and Master programs. Moreover, to verify the defined null hypotheses non-parametric tests were used. In case differences were present but not sig-

nificant, this was explicitly mentioned and discussed. Two-way ANOVA was used to detect possible interactions between Method and Task. Even if all the assumptions for using ANOVA were not verified, this test is quite robust and has been used extensively in the literature to conduct analysis similar to ours [40, 41, 48]. The conclusion validity could be also affected by the sample size. Accordingly, further replications on a larger sample are needed. Finally, the used post experiment survey questionnaire was designed using standard ways and scales [38].

7. Conclusions and Future Work

This work is in the direction of the media-effects theories applied to the software engineering processes. In particular, most of these theories indicate the face-to-face communication as the richest medium with respect to all the other communication media (including computer conferencing) [49] and assert that the subjects' performances decrease when leaner media are used [50]. As shown in [26], this can also be applicable to the context of a distributed requirements engineering process. These claims were not perfectly supported by our previous experiences (e.g., [24, 25]) and then we decided to define a novel method with the aim of filling the gap with the face-to-face communication. In particular, we adapt a widely and well known method for problem solving (i.e., think-pair-square [5, 6]) to the context of the requirements specification in a distributed requirements engineering process.

A prototype of a supporting tool has been also implemented to support the proposed method. A remarkable characteristic of this prototype is that it is simple to set up, to maintain, and to extend as compared to the widely employed communication infrastructures (e.g., videoconferencing communication tools) and modeling tools used in the global requirements engineering field [11, 26].

In order to investigate whether our method and its supporting tool reduce the global software development challenges related to the impact of temporal and geographical differences we have conducted two controlled experiments. In particular, these experiments aimed at investigating whether our proposal is effective as a traditional brainstorming session based on the face-to-face communication. The results indicate that the subjects grouped in teams significantly spent less time to accomplish a modeling task when using a traditional brainstorming session. Differently, the subjects produced on average

better quality models when accomplished the task using our method and tool. Their effect is not statistically significant.

We exploited controlled experiments since they allow controlling a number of confounding and uncontrollable factors that could be present in case studies conducted both in research laboratory and industrial settings [34, 48]. In fact, it may be quite impossible to control factors such as learning/fatigue effects and to select specific tasks. The use of controlled experiments may arise however questions about the external validity. Nevertheless, controlled experiments can be conducted in the early steps of an empirical investigation plan and before conducting empirical investigations in more realistic settings, see for example [44]. Accordingly, we will conduct case studies with practitioners to further investigate the efficacy and the effectiveness of the proposed method to different modeling activities, such as conceptual and detailed modeling of software systems.

The conducted experiments were designed considering quantifiable benefits from the use of both a traditional brainstorming session based face-to-face session and the method and the tool presented here. On the other hand, concerns and advantages deriving from the work in groups [51] have been ignored as well as cultural and background diversities [36]. Despite this design choices, we were able to conduct a very preliminary qualitative analysis on how the charisma and the authority of the subjects conditioned the performances of the teams. This was possible due to our personal knowledge of each subject involved in the experiments. The results revealed that more charismatic and authoritative subjects condition more a traditional brainstorming session. The effect of these two characteristics is less evident when subjects used our proposal. As future direction for our research, we are going to conduct a further study to get a better understanding of how charisma, authority, and ability affect team performances, when using or not the method.

Future work will be also devoted to conduct a comparison between our proposal and other distributed communication media (e.g., video conferencing). It would be also worth analyzing the effect of the size of the teams on the quality of the modeled high level use case diagrams when team members are virtually or physically collocated. We also plan to investigate the effect of group dynamic issues [52] (e.g., lack of focus) of brainstorming sessions on requirements modeling tasks.

A future possible direction for our research could consist in comparing our proposal with the traditional think-pair-square method. This kind of

investigation could be useful to better understand the penalty distribution brings.

8. Acknowledgments

The authors would like to thank the ISISlab (www.isislab.it) for the support provided to install and customize the CoFFEE system. Special thanks are due to the students that participated and to Carmine Gravino for his comments and suggestions.

References

[1] J. D. Herbsleb, A. Mockus, An empirical study of speed and communication in globally distributed software development, IEEE Trans. Software Eng. 29 (6) (2003) 481–494.

[2] B. Sengupta, S. Chandra, V. Sinha, A research agenda for distributed software development, in: Proceedings of the 28th international conference on Software engineering, ACM, New York, NY, USA, 2006, pp. 731–740. doi:http://doi.acm.org/10.1145/1134285.1134402.

[3] A. H. Dutoit, J. Johnstone, B. Brügge, Knowledge scouts: Reducing communication barriers in a distributed software development project, in: APSEC, 2001, pp. 427–430.

[4] B. Bruegge, A. H. Dutoit, Object-Oriented Software Engineering: Using UML, Patterns and Java, Second Edition, Prentice Hall, 2003.

[5] F. T. Lyman, The responsive classroom discussion: The inclusion of all students, in: In A. Anderson (Ed.), Mainstreaming digest, College Park, MD: University of Maryland Press, 1981, pp. 109–113.

[6] B. J. Millis, P. G. Cottell, Cooperative learning for higher education faculty, Phoenix, Ariz. : Oryx Press, 1998.

[7] OMG, Unified modeling language (UML) specification, version 2.0, Tech. rep., Object Management Group (July 2005).

[8] N. Bos, J. S. Olson, D. Gergle, G. M. Olson, Z. Wright, Effects of four computer-mediated communications channels on trust development, in:

Int'l ACM Conference on Human Factors in Computing Systems, 2002, pp. 135–140.

[9] E. J. Davidson, Joint application design (jad) in practice, Journal of Systems and Software 45 (3) (1999) 215–223.

[10] B. Brügge, A. D. Lucia, F. Fasano, G. Tortora, Supporting distributed software development with fine-grained artefact management, in: International Conference on Global Software Engineering (ICGSE), 2006.

[11] D. Damian, A. Eberlein, M. L. G. S., B. R. Gaines, Using different communication media in requirements negotiation, IEEE Software 17 (3) (2000) 28–36. doi:http://dx.doi.org/10.1109/52.896247.

[12] J. Grudin, Computer-supported cooperative work: history and focus, Computer 27 (5) (1994) 19–26. doi:10.1109/2.291294.

[13] P. Dewan, J. Riedl, Toward computer-supported concurrent software engineering, IEEE Computer 26 (1) (1993) 17–27.

[14] J. C. Grundy, J. R. Venable, J. G. Hosking, W. B. Mugridge, Supporting collaborative work in integrated information systems engineering environments (1996).

[15] Q. Chen, J. Grundy, J. Hosking, An e-whiteboard application to support early design-stage sketching of uml diagrams, in: Proceedings of the IEEE Symposium on Human Centric Computing Languages and Environments, IEEE Computer Society, Washington, DC, USA, 2003, pp. 219–226.

[16] A. De Lucia, F. Fasano, G. Scanniello, G. Tortora, Enhancing collaborative synchronous uml modelling with fine-grained versioning of software artefacts, Journal of Visual Languages Computing 18 (5) (2007) 492–503.

[17] A. De Lucia, F. Fasano, G. Scanniello, G. Tortora, Concurrent fine-grained versioning of uml models, in: Proceedings of Conference on Software Maintenance and Reengineering, 2009, pp. 89–98.

[18] N. Boulila, Group support for distributed collaborative concurrent software modeling, in: ASE '04: Proceedings of the 19th

IEEE international conference on Automated software engineering, IEEE Computer Society, Washington, DC, USA, 2004, pp. 422–425. doi:http://dx.doi.org/10.1109/ASE.2004.39.

[19] J. C. Grundy, W. B. Mugridge, J. G. Hosking, R. W. Amor, Support for collaborative, integrated software development, in: Proceedings of the 1995 Software Engineering Environment Conferences, IEEE Computer Society, Washington, DC, USA, 1995, p. 84.

[20] N. Boulila, Computer supported cooperative software engineering: A framework for supporting distributed concurrent group modeling of software, in: Proceedings of the International Conference on Applied Computing, 2004.

[21] C. Damm, K. Hansen, M. Thomsen, Tool support for object-oriented cooperative design: Gesture-based modeling on an electronic whiteboard, Proceedings of ACM Conference on Human Factors in Computing Systems, CHI Letters 2 (1) (2000) 518–525.

[22] B. Nuseibeh, S. Easterbrook, Requirements engineering: a roadmap, in: Workshop on The Future of Software Engineering at ICSE, ACM Press, Limerick, Ireland, 2000, pp. 35–46. doi:10.1145/336512.336523.
URL `http://dx.doi.org/10.1145/336512.336523`

[23] F. Calefato, D. Damian, F. Lanubile, An empirical investigation on text-based communication in distributed requirements workshops, in: Proceedings of the International Conference on Global Software Engineering, IEEE Computer Society, Washington, DC, USA, 2007, pp. 3–11. doi:http://dx.doi.org/10.1109/ICGSE.2007.9.

[24] U. Erra, G. Scanniello, Assessing communication media richness in requirements negotiation, IET Software 4 (2) (2010) 134–148.

[25] U. Erra, A. Portnova, G. Scanniello, Comparing two communication media in use case modeling: Results from a controlled experiment, in: IEEE Int'l Symp. on Empirical Software Engineering and Measurement (ESEM 2010), 2010.

[26] D. Damian, F. Lanubile, T. Mallardo, On the need for mixed media in distributed requirements negotiations, IEEE Transaction on Software Engineering 34 (1) (2008) 116–132.

[27] B. Boehm, A. Egyed, Software requirements negotiation: Some lessons learned, in: Proceedings of the 20th International Conference on Software Engineering, 1998, pp. 503–507.

[28] F. Ricca, G. Scanniello, M. Torchiano, G. Reggio, E. Astesiano, On the effort of augmenting use cases with screen mockups: Results from a preliminary empirical study, in: IEEE Int'l Symp. on Empirical Software Engineering and Measurement (ESEM 2010), 2010.

[29] A. Gupta, 24-hour knowledge factory paradigm and its role in surmounting organizational, national, and other barriers.

[30] A. Dutoit, R. McCall, I. Mistrík, B. Paech, Rationale Management in Software Engineering, 2006. doi:10.1007/978-3-540-30998-7.
URL http://dx.doi.org/10.1007/978-3-540-30998-7

[31] R. De Chiara, A. Di Matteo, I. Manno, S. V., Coffee : Cooperative face2face educational environment, in: CollaborateCom, 2007, pp. 243–252.

[32] V. R. Basili, F. Shull, F. Lanubile, Building knowledge through families of experiments, IEEE Trans. Software Eng. 25 (4) (1999) 456–473.

[33] N. Juristo, A. Moreno, Basics of Software Engineering Experimentation, Kluwer Academic Publishers, 2001.

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

[35] V. Basili, G. Caldiera, D. H. Rombach, The Goal Question Metric Paradigm, Encyclopedia of Software Engineering, John Wiley and Sons, 1994.

[36] J. S. Olson, G. M. Olson, Culture surprises in remote software development teams, Queue 1 (9) (2004) 52–59. doi:http://doi.acm.org/10.1145/966789.966804.

[37] W. B. Frakes, R. Baeza-Yates, Information Retrieval: Data Structures and Algorithms, 1992.

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

[39] W. J. Conover, Practical Nonparametric Statistics, John Wiley & Sons, 1998.

[40] L. C. Briand, Y. Labiche, M. D. Penta, H. D. Yan-Bondoc, An experimental investigation of formality in uml-based development, IEEE Transaction on Software Engineering 31 (10) (2005) 833–849.

[41] F. Ricca, M. D. Penta, M. Torchiano, P. Tonella, M. Ceccato, How developers' experience and ability influence web application comprehension tasks supported by uml stereotypes: A series of four experiments, IEEE Transaction on Software Engineering 36 (1) (2010) 96–118.

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

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

[44] M. Colosimo, A. De Lucia, G. Scanniello, G. Tortora, Evaluating legacy system migration technologies through empirical studies, Information & Software Technology 51 (2) (2009) 433–447.

[45] R. McGill, J. W. Tukey, W. A. Larsen, Variations of box plots, The American Statistician 32 (1) (1978) 12–16.

[46] M. A. N. Ehsan, E. Mirza, Impact of computer-mediated communication on virtual teams performance: An empirical study 32 (August) (2008) 833–843.

[47] J. Hannay, M. Jørgensen, The role of deliberate artificial design elements in software engineering experiments, IEEE Transaction on Software Engineering 34 (2) (2008) 242–259.

[48] E. Arisholm, L. C. Briand, S. E. Hove, Y. Labiche, The impact of uml documentation on software maintenance: An experimental evaluation, IEEE Trans. Softw. Eng. 32 (6) (2006) 365–381. doi:http://dx.doi.org/10.1109/TSE.2006.59.

[49] A. R. Dennis, J. S. Valacich, Rethinking media richness: Towards a theory of media synchronicity, in: HICSS, 1999.

[50] A. R. Dennis, S. T. Kinney, Testing media richness theory in the new media: The effects of cues, feedback, and task equivocality, Information Systems Research 9 (3) (1998) 256–274.

[51] B. Kitchenham, S. L. Pfleeger, L. Pickard, P. J., D. C. Hoaglin, K. E. Emam, J. Rosenberg, Preliminary guidelines for empirical research in software engineering, IEEE Trans. Software Eng. 28 (8) (2002) 721–734.

[52] D. R. Forsyth, Group Dynamics, 4th Edition. Belmont, CA: Thomson Wadsworth.