Assessing Think-Pair-Square in Distributed Modeling of Use Case Diagrams

Ugo Erra
Department of Mathematic and Computer Science
University of Basilicata
Potenza, Italy
Email: ugo.erra@unibas.it

Giuseppe Scanniello
Department of Mathematic and Computer Science
University of Basilicata
Potenza, Italy
Email: giuseppe.scanniello@unibas.it

***Abstract*—In this paper, we propose a new method for the modeling of use case diagrams in the context of global software development. It is based on think-pair-square, a widely used cooperative method for active problem solving. The validity of the developed technology (i.e., the method and its 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 the time needed to model use case diagrams and the quality of the produced models. The data analysis indicates a significant difference in favor of the brainstorming session for the time, with no significant impact on the requirements specification.**

***Keywords*-Controlled experiments; Global software engineering; Functional modeling; Requirements engineering;**

I. Introduction

Splitting the development of a software product (e.g., system or service) among distributed sites is increasingly becoming a common practice in the software industry. Academy and industry community refers to this relevant phenomenon as global, distributed, or multi-site software development [1]. 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 [1].

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 customers. In order to model functional requirements, several approaches have been proposed in the past, and of these, behavioral modeling is a common part of the most broadly employed ones [2]. Behavioral modeling includes the requirements elicitation phase in which stakeholders communicate and cooperate to solve problems and to represent functional requirements in terms of use case diagrams and use cases [3].

In this work, we have modified the original definition of the think-pair-square method [4] to make it suitable for the global software engineering. In particular, we have modified this method to support the requirements elicitation phase. The new method has been implemented in a software environment, thus enabling both the distributed modeling and synchronous communication among team members. Two controlled experiments have been conducted to compare the developed technology (i.e., the proposed method and the supporting environment) with a brainstorming session based on face-to-face interaction. The comparison has been performed on the performances of the involved participants with respect to a traditional requirements elicitation task in which they were asked to develop a part of a functional model by employing use case diagrams (in the following we will also refer to it as *high level use case diagram*). The performances have been assessed with respect to the time needed to model a high level use case diagram and the overall quality of this diagram. The second experiment is a replication, where the same participants as the original experiment were involved on different modeling tasks. The rationale for conducting this replication was that we were interested in verifying whether better trained participants benefit more from a high level of familiarity with our technology.

The remainder of the paper is organized as follows. We discuss the proposed method in Section II. The experiments are introduced in Section III, while the results are presented in Section IV. The discussion of the results and threats that may affect the validity of the investigation are highlighted in Section V. Final remarks and future work conclude the paper.

II. Method and Environment

We propose a new method for global software development with application to the requirements elicitation. In the following, we first recall the general definition of the think-pair-square method and then how it has been modified and implemented in our modeling and communication environment.

A. Background

The think-pair-square method [4] has been conceived for promoting active learning to solve problems in a 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 heteroge-

978-1-4577-1076-6/11/$26.00 ©2011 IEEE

neous 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 step helps individuals 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 the ideas achieved in the previous step may be revised.

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 two or more pairs, thus forming a square and discussing again possible solutions for the original problem. This step allows identifying 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 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 pairs combine results and generate a more comprehensive solution.

B. Distributed Think-Pair-Square

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 customers. The most challenging and demanding phase of the requirements engineering process is the elicitation [2]. 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 distributed.

Based on our experience [5] 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. With respect to the original definition of the think-pair-square method, we have planned to replace face-to-face interaction with a computer-supported collaborative tool environment. In particular, face-to-face interaction has been simulated with a distributed environment composed of a text-based chat and a synchronous use case diagram modeling tool. The chat is in charge of simulating face-to-face interaction, while the modeling tool is used 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] [6] among users while performing modeling tasks.

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

Our environment has been implemented using CoFFEE [7]. It is mainly aimed at developing environments for cooperative learning and distributed synchronous cooperative activities. To this end, it provides three main tools: *(i)* Session Editor, *(ii)* Controller, and *(iii)* Discusser.

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 [3]. In our case, we defined a session composed of three steps (one for each phase of our method). In all these steps, the tool SDT is provided to specify a high level use case diagram. 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 they use SDT to present a graphical overview of the functionality of a system. To effectively support our method, at each step the model created by using SDT is made available to the subsequent one.

The Controller tool is used to execute sessions previously defined by the user. This tool has been used here to execute the session described above and to pass from a step to subsequent one. We have also used the Controller tool to compose groups of participants and to constantly monitor them while performing the modeling tasks.

Finally, the Discusser tool enables users exploiting the environment created by the Session Editor tool and executed by the Controlled tool. The communication among the users follows the session underlying the communication environment and is supported by the tools that compose this session.

III. Controlled Experiments

In this section, we present the definition, the design, and the planning of the original experiment (from here on, *Experiment I*) and its replication[2] (from here on, *Experiment II*), structured according to the templates suggested in [9] and [10]. For replication purposes, the experimental package and the raw data of both the experiments are available for downloading on the web[3]. A technical report is also available there.

A. Definition

The main goal of both the experiments is to investigate whether our method and its implementation are 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. In the following, we will refer to **TPS** as our developed technology (i.e., method and environment), while we use **F2F** to indicate a traditional face-to-face brainstorming session.

[2]It is a differentiated replication [8], since we introduced a variation in the original conditions.

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

978-1-4577-1076-6/11/$26.00 ©2011 IEEE

The goal of the study can be therefore defined by using the GQM (Goal Question Metric) [11] 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 diagrams 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 diagrams in the context of a global software development project*".

B. Context

The context of the experiments was constituted of second year Bachelor students and first year 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. All the participants to the experiments were volunteers and no selection was accomplished.

The Bachelor students were asked to accomplish the controlled experiments as a part of a series of optional laboratory exercises of the Software Engineering course. As mandatory laboratory activity of this course, the participants 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 participants 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, to cooperatively modeling the system, and to disseminate and share information among the team members. The participants 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 a Software Engineering Course. As mandatory activity of the Computer Graphic course, the students 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.

Bachelor and Master students were volunteers and were aware of the pedagogical purpose of the experiment, but they did not know the experimental hypotheses. Note also that all the participants 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.

C. Hypotheses Formulation

The classification proposed by Robert in [12] shows that the media considered in this investigation are synchronous (i.e., the communication happens in real time), while they have a different space dimension. According to this classification, the participants involved in experimentation are virtually and physically collocated when interact to accomplish a given task.

We are particularly interested in investigating whether virtually collocated participants may obtain similar performances (i.e., the quality of the software artifacts produced within the modeling tasks and the time needed to accomplish these tasks) than physically collocated ones. Hence, 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 high level use case diagrams with respect to TPS.
- **Hn2** There ***is not a significant difference*** with respect to the quality of the use case diagrams when using F2F or TPS.

The first null hypothesis is one sided since we expected that participants spent less time to accomplish the tasks using F2F. This is due to both the results presented in [13], [14] 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 participants 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 that can be easily derived.

We were also interested in assessing whether participants better trained on our technology achieve better performances. To this end, two further null hypotheses have been defined and tested:

- **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.

[4]It is a document developed by the project management and the clients 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.

978-1-4577-1076-6/11/$26.00 ©2011 IEEE

TABLE I
EXPERIMENT DESIGN

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

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

D. Design

Table I summarizes the adopted design. This design has been adopted to mitigate as much as possible learning/fatigue effect. It also 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 participants were sufficiently familiar with. The problem statements of the systems are available in the experimental package.

In each experiment the participants were asked to accomplish 2 tasks (i.e., *Task1* and *Task2*). For each task, the participants were asked to create a high level use case diagram on the basis of the provided problem statement. The participants had a break between the tasks.

Before the experiments, the students were asked to fill in a pre-questionnaire to get information on the industrial working experience and GPA (Grade Point Average). Since the greater part of the students did not have any remarkable working experience, we used their GPA to equally distributed high and low ability participants among 9 teams (the interested reader can found demographic information of the participants in the experimental package). Each team contained 3 Bachelor students and 1 Master student. The teams were randomly assigned to the groups A and B. In particular, 5 and 4 teams were assigned to the groups A and B, respectively.

The groups and the teams were the same for both Experiment I and Experiment II. Accordingly, we assumed that the participants passing from Experiment I to Experiment II increased their experience and familiarity with our technology. This design choice also allowed the participants to consolidate his/her working style and to develop a shared work habit with the team mate [15].

E. Selected Variables

In order to properly design and then analyze the results, for each experiment only the main factor *Method* was considered. It is a nominal variable with two possible values: F2F, TPS. Two additional variables (also named factors or co-factors) have been also considered and controlled in each experiment: *Task* ($\in \{Task1, Task2\}$) and *Object* ($\in \{Object1, Object2\}$). To analyze the effect of the participants' familiarity with the developed technology, we considered the performances of the participants when they used TPS to accomplish the tasks within the experiments. Therefore, to test the null hypotheses Hn3 and Hn4, we considered the nominal variable *Experiment* ($\in \{ExperimentI, ExperimentII\}$).

To statistically test the formulated null hypotheses, we selected the following dependent variables:

Time indicates the number of minutes that all the participants within a given team spent to accomplish a task;

Quality measures the quality of a model.

We used an approach based on the information retrieval theory [16] to get a quantitative assessment of the quality of a high level 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 participant or a given group of participants, while A_d is the set of dependencies between actors and use cases. Accordingly, $precision_{a/c}$ measures the correctness of both actors/use cases belonging to a given use case diagram. The correctness of the dependencies between actors and use cases is measured by using $precision_d$. On the other hand, $recall_{a/u}$ measures the completeness of a use case diagram with respect to its actors/use cases, while $recall_d$ is 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 [16] to get a balance between correctness and completeness of actors/use cases of a use case diagram (i.e., $F - Measure_{a/u}$). We also used the harmonic mean of precision and recall to get a balance between correctness and completeness of the dependences among objects in a use case diagram (i.e., $F - Measure_d$).

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 $F - Measure_d$ by computing their arithmetic mean. This mean represent the Quality dependent variable. In case this variable assumes a value close to 1,

978-1-4577-1076-6/11/$26.00 ©2011 IEEE

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. The greater the Quality value, the better is the model.

The authors developed the oracles before conducting the experiments and so without analyzing the high level use case diagrams produced by the participants. 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. However, it is 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 also affected by ambiguities in the associations between the names of the use cases of an oracle and the names of the use cases of a modeled diagram. 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 participants were not asked to model use cases. The rationale for this design choice was inspired by the need of achieving a trade-off between complexity of the tasks and the 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 participants, the authors conducted a meeting to solve all the possible issues and ambiguities.

F. 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 participants involved in the study attended a lesson in which we posed great care to explain the required granularity level in the specification of use cases. In particular, we explained that we were interested in the specification of use case diagrams based on summary level use cases rather than user-goal or sub-function use cases. Instructions on the experimental procedure to follow were provided together with details on the method and the environment.

The experiments were performed in a laboratory at the University of Basilicata. No interaction was allowed among the participants when they used TPS, so simulating as much as possible a modeling task in a global development scenario.

At the end of each experiment, the participants were asked to fill in a post experiment survey questionnaire. For space reasons, we did not present the results achieved. The reader can find details in the technical report available on the web.

Regarding the preparation of the PCs used in the experiments, we installed the CoFFEE Controller on 5 desktops (i.e., the server nodes), while the CoFFEE Discusser was installed on the participants' PCs. 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 participants' 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 participants to use only our environment in the experimentation.

G. Execution

In order to carry out each experiment, the participants were provided with a pencil, some paper sheets, and the following hard copy material: *(i)* The introductory presentation of both the experiment and the environment; *(ii)* The problem statements of the software systems to be used in the tasks (the problem statement of Task2 was provided after the participants accomplished Task1); *(iii)* The post-experiment survey questionnaire to be filled in at the end of the experiment.

Note that the size of the text within the problem statements of the systems was nearly the same. The problem statement of ECP (i.e., Object2 of Experiment II) included some screen mockups to clarify some functionality to implement [2]. This difference was deliberately introduced and properly controlled (see Section V-A).

H. Analysis Procedure

To analyze the collected data and to test the defined null hypotheses, statistical tests have been used. In all the tests, we decided (as usual) to accept a probability of 5% of committing Type-I-Error [9], i.e., of rejecting a null hypothesis when it is actually true.

We used the non-parametric paired Wilcoxon's test [17] to verify the null hypotheses. This test was employed due to the sample sizes and mostly non-normality of the data. Furthermore, it has been used in the literature for purposes similar to ours [18]. Other than testing the formulated hypotheses, it is of practical interest to estimate the magnitude of performance differences of the participants when using TPS and F2F. Similar to [18], we used the Cohen's d effect size [19] to understand how a factor effects a dependent variable. Typically, the 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 design, we applied the Cohen's d effect size for dependent samples (to be used in the context of paired analyses).

To assess the effect of Task on the dependent variables and to analyze its possible interactions with the main factor, we used a two-way Analysis of Variance (ANOVA) [20].

978-1-4577-1076-6/11/$26.00 ©2011 IEEE

TABLE II
EFFECT OF METHOD WITHIN THE INDIVIDUAL EXPERIMENTS

Experiment I				
Hypotheses	p-value	Cohen's *d*	F2F>TPS	F2F<TPS
Hn1 (Time)	**YES (0.006)**	Large (1.39)	1	8
Hn2 (Quality)	NO (0.677)	Small (-0.34)	4	5
Experiment II				
Hypotheses	p-value	Cohen's *d*	F2F>TPS	F2F<TPS
Hn1 (Time)	**YES (0.001)**	Large (1.72)	1	8
Hn2 (Quality)	NO (0.173)	Medium (0.53)	3	6

TABLE III
ANOVA RESULTS FOR THE INDIVIDUAL EXPERIMENTS

Experiment I		
Factor	Time	Quality
Method	**YES (<0.001)**	NO (0.703)
Task	**YES (0.001)**	NO (0.690)
Method vs Task	**YES (0.002)**	NO (0.414)
Experiment II		
Method	**YES (<0.001)**	NO (0.104)
Task	NO (0.108)	**YES (0.025)**
Method vs Task	NO (0.352)	NO (0.156)

TABLE IV
RESULTS OF THE EXPERIMENTS TOGETHER

Hypotheses	p-value	Cohen's *d*	ExperimentI> ExperimentII	ExperimentI< ExperimentII
Hn3 (Time)	**YES (<0.001)**	Large (2.2)	9	0
Hn4 (Quality)	NO (0.345)	Negligible (-0.09)	3	6

To estimate the magnitude of performance difference of the participants when using TPS in Experiment I and Experiment II, we used the Cohen's *d* effect size for dependent samples.

IV. RESULTS

A. Experiment I

Table II summarizes the results of the Wilcoxon test on the defined null hypotheses and the values of the effect size for both the experiments. Furthermore, this table also shows the number of teams that spent more time to accomplish the tasks with F2F with respect to TPS(i.e., F2F>TPS). The number of teams that spent less time to accomplish the tasks with TPS (i.e., F2F<TPS) is reported as well. The table also shows the number of teams that produced better models using F2F with respect to TPS (i.e., F2F>TPS). The number of teams that produced better models with TPS is reported as well (i.e., F2F<TPS).

The Wilcoxon test revealed that Hn1 can be rejected. This results indicate a significant difference in terms of time to accomplish the tasks (p-value = 0.006). The effect size is large (i.e., 1.39). Furthermore 8 out 9 teams spent less time to accomplish the task when using F2F (see Table II).

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

1) Other Factors: Table III summarizes the results achieved by applying ANOVA on Method and Task. The table also show the results on the interaction between Method and Task. The results indicate a positive effect of Method (p-value < 0.001) and Task (p-value = 0.001)on Time. Differently, the effect of Method and Task on Quality is not statistically significant since the p-values were 0.703 and 0.690, respectively. A significant interaction between Task and Method on Time is present (p-value = 0.002), while the interaction on these variables is not significant on Quality (p-value = 0.0414).

B. Experiment II

Hn1 can be rejected (see Table II) 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 is large (i.e., 1.72). Furthermore 8 out of 9 teams spent less time when using F2F. The results of the Wilcoxon test did not enable to reject Hn2 (p-value = 0.173). The effect size is medium as the Cohen's *d* value is 0.53. Moreover, 6 out of 9 teams modeled better diagrams when using TPS.

1) Other Factors: The ANOVA results indicate a positive effect of Method on Time (p-value < 0.001). On the other hand, the effect of Task on Time and the interaction between Task and Method on Time are not statistically significant (the p-values are 0.108 and 0.352, respectively). The effect of Method is not statistically significant on Quality (p-value = 0.104), while it is statistically significant on Task (p-value = 0.025). The ANOVA results also revealed that the interaction between Task and Method on Quality is not significant (p-value = 0.156).

C. The Effect of Experience on TPS

The results of the Wilcoxon test (see Table IV) show that the null hypothesis Hn3 can be rejected (p-value < 0.001) with a large effect size (i.e., 2.2). Therefore, using TPS the participants spent significantly less time to accomplish the tasks in the replication. 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). The effect size was negligible (i.e., -0.09). However, 6 out of 9 teams modeled better quality use case diagrams in Experiment II. It is worth noting that the observed difference between the two experiments could not be due to the increased experience of the participants when passing from the first experiment to the second one. This difference could be due to the experimental objects used in the experiments.

V. DISCUSSION AND THREATS TO VALIDITY

In the following, we discuss the results and the threats that may affect their validity.

A. 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. It is possible that the participants

978-1-4577-1076-6/11/$26.00 ©2011 IEEE

spent more time when modeling high level use case diagrams with our technology because they had to write a number of messages to both communicate each other and accomplish the modeling task. Further, the agreement within each team passed trough three steps. Despite more time is needed to accomplish a modeling task with our technology, the observed results have meaningful and practical implications. In fact, in case the time distance is not an issue, but moving people might be a problem, our proposal 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. Differently, the time needed to accomplish the second modeling tasks was not affected in case the participants 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 [21], 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 participants 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 III). A further analysis indicates that screen mockups in the problem statement did not properly support participants in the accomplishment of tasks using face-to-face interaction. In fact, the participants who used first our technology to accomplish the task starting from a problem statement without screen mockups produced worse quality use case diagrams when they successively used a brainstorming session on a problem statement with screen mockups (0.77 and 0.56, respectively). A slightly difference (0.81 and 0.75) in terms of use case diagram quality was observed in case the participants accomplished the task using a brainstorming session first with screen mockups and then with our technology and screen mockups. These findings together with the fact that participants 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 [14], this result may have practical implications. Further investigations are needed to understand the effect of employing our technology with additional material.

The results also indicated that the effect of the familiarity on the technology did not significantly affect the quality of the modeled use case diagram. Although we cannot conclude anything definitively as we were not able to reject Hn4, a further analysis indicated a possible positive effect of the participants' familiarity on the diagram quality. This result suggests that also participants, who are not perfectly trained may benefit from our technology. Also this result has a practical implication since a training phase could be shortened without affecting the quality of the produced diagrams.

B. Threats to Validity

The *Internal Validity* was mitigated by the experiment design since each group of participants 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 (see Table II). Moreover, we observed a significant interaction between Task and Method on Time (see Table III). Specifically, in the second task we observed a reduction of the mean time to complete it. This could be the consequence of the effect of the method on the cohesion of team members (see Section V-A).

The *External Validity* may be present when experiments are conducted with students, since they could not be representative as software professionals [22]. Indeed, students could be more comfortable than software practitioners with our modeling environment and less experienced with brainstorming session. In this scenario, the use of students as participants may bring out phenomena related to the learning of new communication media. They could be more prone to learn and master our environment. This is enough unusual for student experiments. However, the greater part of the involved undergraduate students will be integrated in the software industry market soon, while the graduate students are not far from junior industrial analysts (most of them have some working experience due to the internship they made in the industry as final part of their Bachelor degree). Threats belonging to this category can be also related to the complexity of the tasks. Another possible threat might concern cultural and background diversities among the participants. In fact, space and time are not the only issues in the global software development [15]. Work that takes place over long distances means that communication often involves different cultures. Future work is needed to investigate whether the achieved results can be generalized in case participants have cultural and background diversities.

The *Construct Validity* was mitigated by a proper design. The selection and the measurement of the dependent variables could threaten this validity. Regarding the Time variable and the tasks accomplished with a brainstorming session, we asked the participants to note down the start and stop time. These information was validated by the supervisors in both the experiments. Although this may not be very accurate, this is a widely adopted in the literature (e.g., [18]).

The selection of the population may affect the *Conclusion Validity*. However, the participants were not far from junior developers/analysts. To verify the null hypotheses non-parametric tests were used. In case differences were present but not significant, this was explicitly mentioned and discussed. We used ANOVA to detect possible interactions between Method and Task. Even if the assumptions of the

978-1-4577-1076-6/11/$26.00 ©2011 IEEE

test were not verified, it is quite robust and has been used extensively in the literature to conduct analysis similar to ours [18], [23]. The conclusion validity could be also affected by the sample size. Replications on a larger sample are needed.

VI. Conclusions and Future Work

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 [24], [25]. The effect of different communication media and synchronous/asynchronous modeling tools has been marginally investigated in the distributed requirements engineering process. For example, Damian *et al.* [24] show an empirical study to compare five physical group configurations: one face-to-face and four distributed. This study indicates that the highest group performance occurred when clients are separated and collocated with the system analyst. The effect of using mixed media in distributed requirements negotiations is proposed in [26]. The study reveals that the requirements negotiation is more effective when an asynchronous structured discussion is conducted before a synchronous meeting.

Recently, we have investigated [5] the effectiveness of preliminarily applying the method proposed here in the modeling of use cases with respect to a brainstorming session. The study reveals 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. Several are the differences between the investigation presented in this paper and the previous conducted experiment. The most remarkable differences are: *(i)* different participants; *(ii)* the method is here applied to model use case diagrams; *(iii)* the effect of familiarity with the our proposal is preliminarily assessed.

In this paper, we propose a new application field for the synchronous method proposed in [5], namely the distributed modeling of use case diagrams. The validity of our proposal has been assessed through two controlled experiments. The experiments have been conducted to compare the developed technology with a brainstorming session based on face-to-face interaction. The data analysis indicates a significant difference in favor of the brainstorming session for the time, with no significant impact on the quality of the produced models.

Future work will be devoted to conduct a comparison between our proposal and different 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 collocated. We also plan to assess whether our approach can be also applied to asynchronous communication and modeling. We also plan to investigate the effect of group dynamic issues [27] (e.g., lack of focus) of brainstorming sessions in requirements modeling.

References

[1] B. Sengupta, S. Chandra, and V. Sinha, "A research agenda for distributed software development," in *Proceedings of the 28th international conference on Software engineering*. New York, NY, USA: ACM, 2006, pp. 731–740.

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

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

[4] 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.

[5] U. Erra, A. Portnova, and 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)*, September 2010.

[6] A. Gupta, "24-hour knowledge factory paradigm and its role in surmounting organizational, national, and other barriers," 2009.

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

[8] V. R. Basili, F. Shull, and F. Lanubile, "Building knowledge through families of experiments," *IEEE Trans. Software Eng.*, vol. 25, no. 4, pp. 456–473, 1999.

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

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

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

[12] R. Johansen, *GroupWare: Computer Support for Business Teams*. New York, NY, USA: The Free Press, 1988. [Online]. Available: http://portal.acm.org/citation.cfm?id=542298

[13] U. Erra and G. Scanniello, "Assessing communication media richness in requirements negotiation," *IET Software*, vol. 4, no. 2, pp. 134–148, 2010.

[14] F. Ricca, G. Scanniello, M. Torchiano, G. Reggio, and 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)*, September 2010.

[15] J. S. Olson and G. M. Olson, "Culture surprises in remote software development teams," *Queue*, vol. 1, no. 9, pp. 52–59, 2004.

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

[17] W. J. Conover, *Practical Nonparametric Statistics*. John Wiley & Sons, December 1998.

[18] F. Ricca, M. D. Penta, M. Torchiano, P. Tonella, and M. Ceccato, "How developers' experience and ability influence web application comprehension tasks supported by uml stereotypes: A series of four experiments," *IEEE Transaction on Software Engineering*, vol. 36, no. 1, pp. 96–118, 2010.

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

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

[21] M. A. N. Ehsan, E. Mirza, "Impact of computer-mediated communication on virtual teams performance: An empirical study," vol. 32, no. August, pp. 833–843, 2008.

[22] J. Hannay and M. Jørgensen, "The role of deliberate artificial design elements in software engineering experiments," *IEEE Transaction on Software Engineering*, vol. 34, no. 2, pp. 242–259, 2008.

[23] E. Arisholm, L. C. Briand, S. E. Hove, and Y. Labiche, "The impact of uml documentation on software maintenance: An experimental evaluation," *IEEE Trans. Softw. Eng.*, vol. 32, no. 6, pp. 365–381, 2006.

[24] D. Damian, A. Eberlein, M. L. G. S., and B. R. Gaines, "Using different communication media in requirements negotiation," *IEEE Software*, vol. 17, no. 3, pp. 28–36, 2000.

[25] B. Nuseibeh and S. Easterbrook, "Requirements engineering: a roadmap," in *Workshop on The Future of Software Engineering at ICSE*. Limerick, Ireland: ACM Press, June 2000, pp. 35–46. [Online]. Available: http://dx.doi.org/10.1145/336512.336523

[26] D. Damian, F. Lanubile, and T. Mallardo, "On the need for mixed media in distributed requirements negotiations," *IEEE Transaction on Software Engineering*, vol. 34, no. 1, pp. 116–132, 2008.

[27] D. R. Forsyth, *Group Dynamics*. 4th Edition. Belmont, CA: Thomson Wadsworth, 2006.

978-1-4577-1076-6/11/$26.00 ©2011 IEEE