Empirical Software Engineering, 8, 367–395, 2003.
© 2003 Kluwer Academic Publishers. Manufactured in The Netherlands.

An Externally Replicated Experiment for Evaluating the Learning Effectiveness of Using Simulations in Software Project Management Education

DIETMAR PFAHL pfahl@iese.fhg.de
Fraunhofer IESE, Germany

OLIVER LAITENBERGER oliver_laitenberger@droege.de
Droege & Comp., Germany

JÖRG DORSCH joerg.dorsch@accenture.com
Accenture, Germany

GÜNTHER RUHE ruhe@ucalgary.ca
University of Calgary, Canada

Editor: Marc I. Kellner

Abstract. The increasing demand for software project managers in industry requires strategies for the development of the management-related knowledge and skills of the current and future software workforce. Although several approaches help teach the required skills in a university setting, few empirical studies are currently available to characterize and compare their effects.

This paper presents results of an externally replicated controlled experiment that evaluates the learning effectiveness of using a process simulation model for educating computer science students in software project management. While the experimental group applies a system dynamics (SD) simulation model, the control group uses the well-known COCOMO model as a predictive tool for project planning.

The results of the empirical study indicate that students using the simulation model gain a better understanding about typical behavior patterns of software development projects. The combination of the results from the initial experiment and the replication corroborates this finding. Additional analysis shows that the observed effect can mainly be attributed to the use of the simulation model in combination with a web-based role-play scenario. This finding is strongly supported by information gathered from the debriefing questionnaires of subjects in the experimental group. They consistently rated the simulation-based role-play scenario as a very useful approach for learning about issues in software project management.

Keywords: COCOMO, learning effectiveness, replicated experiment, software project management education, system dynamics simulation.

1. Introduction

Software development is a dynamic and complex process since many interacting factors throughout the lifecycle impact the costs and schedule of the development project as well as the quality of the developed software product. To monitor and control software development projects, management experience and knowledge on how to balance the various influential factors are required. However, the growing pervasiveness of software and the increasing number of software development projects result in a lack of well-trained and experienced managers.

To address these issues, process simulation techniques have been applied to the domain of software engineering during the last decade, starting with the pioneering work of Kellner and Hansen (1989) and Abdel-Hamid and Madnick (1991). But although the potential of simulation models for the training of managers has long been recognized (Graham et al., 1992; Milling, 1995; Morecroft, 1988), experience with process simulation as a means for software project management education and training has rarely been published (examples are Drappa and Ludewig, 1999; Madachy and Tarbet 1999).

In fact, only few experimental studies have been conducted involving the use of models that simulate the typical behavior of software projects (Lin, 1993; Lin et al., 1997; Smith et al., 1993). The results of these experiments indicate that a natural one-way causal thinking could be detrimental to the success of software managers. They must rather adopt a multi-causal or systems thinking. Moreover, they must be aware of (unexpected) feedback to their management decisions. These findings highlight the need for new learning and education strategies.

The first strategic step for teaching software project management must already be included in the curriculum of students. University education must teach computer science and software-engineering students not only technology-related skills but also a basic understanding of typical management phenomena occurring in industrial (and academic) software projects. However, practical constraints of a course usually limit the exposure of students to realistic, large-scale industrial software development projects in which they could make their own experiences. This could be partially compensated by making students use software process simulation models that reproduce the behavior of realistic (i.e., complex) software development projects. The question is whether this approach is viable.

This paper presents the results of a controlled experiment and its first external replication to investigate the effectiveness of computer-based training in the field of software project management using a system dynamics (SD) simulation model. While the experiment was originally performed at the University of Kaiserslautern, Germany (Pfahl et al., 2001) its replication took place at the University of Oulu, Finland. Both empirical studies are viewed as exploratory. The intention is to identify and refine important hypotheses and to investigate them in further detail. More replications are planned for the future.

The paper is structured as follows: Section 2 presents the experimental details of the study. Section 3 summarizes the results of the data analysis. Section 4 discusses the various threats to the validity of the study. The paper concludes with improvement suggestions for the experimental design and proposes directions for future research.

2. Description of the Experiment

The main objective of developing and applying a simulation-based training module has been to facilitate effective learning about certain topics of software project management for computer science students. This was done by providing a scenario-driven interactive single-learner environment that can be accessed through the internet by using a standard web-browser. An additional goal was to raise interest in the topic of software project management among computer science students, and to make them aware of some of the difficulties associated with controlling the dynamic complexity of software projects.

The training module used in the study is composed of course material on project planning and control. The core element of the training module is a set of interrelated project management (i.e., planning) models, represented by a simulation model that was created by using the SD simulation modeling method (Forrester, 1971; Richardson and Pugh, 1981). This model simulates typical behavior of software development projects.

In order to investigate the effectiveness of computer-based training in the field of software project management using a SD simulation model, a controlled experiment applying a pre-test–post-test control group design was conducted. The subjects who were willing to participate in the experiment had to complete two tests, one before the training session (pre-test) and one after the training session (post-test). The effectiveness of the training was then evaluated by comparing within-subject post-test to pre-test scores, and by comparing the scores between subjects in the experimental group, i.e., those who used the SD model, and subjects in the control group, i.e., those who used a conventional project planning model instead of the SD model. In the study, the well-known COCOMO model (Boehm, 1981) was used by the control group since this model is quite comprehensive and can be considered as state-of-the-practice in many industrial software organizations.

The various possibilities of conducting a training session is described in Figure 1 as a three-layered graph. The first layer defines the learning goal, i.e., software project management with focus on project planning and control. The second layer defines the type of project planning model used in the training session, i.e., COCOMO model versus SD simulation model. Finally, the third layer defines the learning mode as another dimension to characterize the training session, i.e., inclusion or exclusion of a web-based interactive role-play. The combination of the distinctions made in layers two and three yield four different treatments. Our empirical investigations compare the effectiveness of two of them called T_A (SD model-based learning with web-based interactive role-play scenario) and T_B (standard COCOMO-based learning without web-based interactive role-play).

The following dimensions were used to characterize “effectiveness” of the training session:

1. Interest in software project management issues.

2. Knowledge about typical behavior patterns of software development projects.

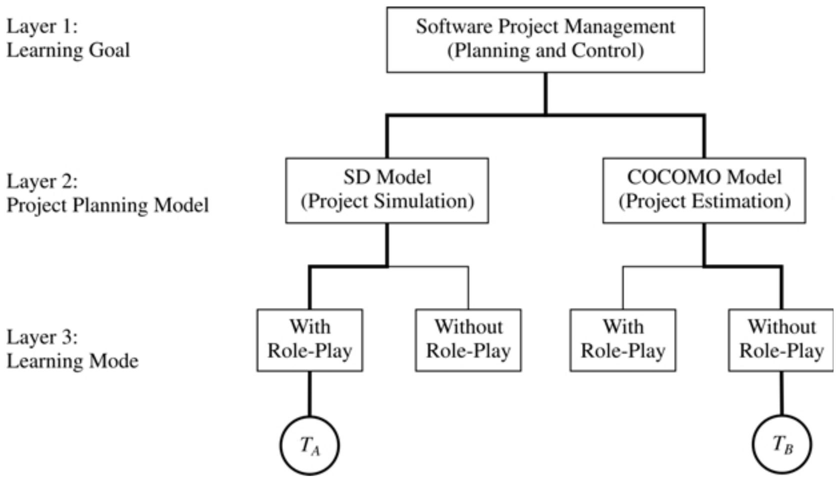


Figure 1. Possibilities for arranging the training session.

3. Understanding of "simple" project dynamics.
4. Understanding of "complex" project dynamics.

In the study, these dimensions were represented by dependent variables ($Y.1$ to $Y.4$).

2.1. Experimental Hypotheses

The hypotheses of the experiments were stated as follows:

1. There is a positive learning effect in both groups (A = experimental group, B = control group). Using the notations in Table 1, this expectation can be formulated as follows:

$$\text{score}_{\text{post}}(Y.i; \text{A}) > \text{score}_{\text{pre}}(Y.i; \text{A}), \text{for } i = 1, \ldots, 4$$
and
$$\text{score}_{\text{post}}(Y.i; \text{B}) > \text{score}_{\text{pre}}(Y.i; \text{B}), \text{for } i = 1, \ldots, 4$$

Table 1. Notation.

Term	Definition
$\text{score}_{\text{pre}}\ (Y.i;\ X)$	Pre-test scores for dependent variable $Y.i$ ($i = 1, \ldots, 4$) of subjects in group X(X = A or B).
$\text{score}_{\text{post}}\ (Y.i;\ X)$	Post-test scores for dependent variable $Y.i$ ($i = 1, \ldots, 4$) of subjects in group X(X = A or B).
$\text{score}_{\text{diff}}\ (Y.i;\ X)$	Difference scores for dependent variable $Y.i$ ($i = 1, \ldots, 4$) of subjects in group X(X = A or B). The difference scores are calculated as follows: $\text{score}_{\text{diff}}\ (Y.i;\ X) = \text{score}_{\text{post}}\ (Y.i;\ X) - \text{score}_{\text{pre}}\ (Y.i;\ X)$

2. The learning effect in group A is higher than in group B, either with regard to the performance improvement between pre-test and post-test (relative learning effect), or with regard to post-test performance (absolute learning effect). The absolute learning effect is of interest because it may indicate an upper bound of the possible correct answers depending on the type of training (A or B). Using the notations in Table 1, this expectation can be formulated as follows:

    a. "Relative learning effect":
    $\text{score}_{\text{diff}}\ (Y.i; \text{A}) > \text{score}_{\text{diff}}\ (Y.i; \text{B})$, for $i = 1, \ldots, 4$

    b. "Absolute learning effect":
    $\text{score}_{\text{post}}\ (Y.i; \text{A}) > \text{score}_{\text{post}}\ (Y.i; \text{B})$, for $i = 1, \ldots, 4$

Note that it is not expected that both relative and absolute learning effect will always occur simultaneously. This reflects on the fact that higher relative learning effects in group A compared to group B are less likely to occur when pre-test scores of group A are significantly higher than those of group B. Similarly, higher absolute learning effects in group A compared to group B are less likely to occur when pre-test scores of group A are significantly lower than those of group B. Standard significance testing was used to analyze expectations. The related null hypotheses were stated as follows:

- $\text{H}_{0,1}$: There is no difference between pre-test scores and post-test scores within group A and group B, i.e., $\text{score}_{\text{pre}}\ (Y.i; \text{A}) = \text{score}_{\text{post}}\ (Y.i; \text{A})$ and $\text{score}_{\text{pre}}\ (Y.i; \text{B}) = \text{score}_{\text{post}}\ (Y.i; \text{B})$ for all $i = 1, \ldots, 4$.

- $\text{H}_{0,2\text{a}}$: There is no difference in relative learning effectiveness between group A and group B, i.e., $\text{score}_{\text{diff}}\ (Y.i; \text{A}) = \text{score}_{\text{diff}}\ (Y.i; \text{B})$ for all $i = 1, \ldots, 4$.

- $\text{H}_{0,2\text{b}}$: There is no difference in absolute learning effectiveness between group A and group B, i.e., $\text{score}_{\text{post}}\ (Y.i; \text{A}) = \text{score}_{\text{post}}\ (Y.i; \text{B})$ for all $i = 1, \ldots, 4$.

2.2. Subjects

The initial experiment was conducted with graduate computer science students at the University of Kaiserslautern (KL), Germany, who were enrolled in the advanced software engineering class. Although none of them had concluded his or her Master degree in computer science or a related field, their skill level was comparable to a Bachelor degree. Twelve students expressed their interest in participation but eventually only nine of them completed all steps of the controlled experiment. The replication of the initial study was conducted during a summer school with twelve graduate and post-graduate students (one Master degree, one PhD) of the University of Oulu, Finland, having their major in computer science, information technology, information engineering, micro electronics or mathematics.

Table 2. Personal characteristics.

	KL students	Oulu students
Average age [years]	27.0	31.3
Share of women	11%	50%
Share of subjects majoring in:	100%	67%
• computer science		
• information technology		
• software engineering		
• information engineering		
• inf. processing science		
Preferred learning style(s):		
• reading (with exercise)	89%	33%
• web-based training	11%	8%
• in-class lecture (with exercise)	22%	25%
• working group (with peers)	33%	42%
Opinion about most effective learning style(s):	Not asked	
• reading (with exercise)		25%
• web-based training		17%
• in-class lecture (with exercise)		33%
• working group (with peers)		67%

In both studies, before performing the pre-test, information was captured on the personal background of the students and their experience with software development and software project management. In addition, the students answered questions about personal characteristics (age, gender), university education (number of terms, major, minor), and preferred learning style on a voluntary basis. The personal characteristics of the two groups are summarized in Table 2.

2.3. Treatments

The training sessions of both groups, experimental and control, was structured by training scenarios that consisted of a sequence of scenario blocks. The generic training scenario structure is composed of the following four scenario blocks:

1. *Block 1—PM Introduction:* General introduction into the main tasks of software project managers and the typical problems they have to solve with regard to project planning and control. This includes a brief discussion of problems caused by the so-called "magic triangle", i.e., the typical presence of unwanted trade-off effects between project effort (cost), project duration, and product quality (functionality).

2. *Block 2—PM Role Play:* Illustration of common project planning problems on the basis of an interactive case example in which the trainee takes over the role of a fictitious project manager.

3. *Block 3—PM Planning Models:* Presentation of basic models that help a project manager with planning tasks, namely a process map, and a predictive model for effort, schedule and quality.

4. *Block 4—PM Application Examples:* Explanation on how to apply the planning models on the basis of examples that are presented in the form of little exercises.

2.3.1. Treatment of the Experimental Group

The experimental group completed all scenario blocks. The SD model was used as the predictive model in scenario blocks 3 and 4. In addition, the SD model was integrated into the interactive role-play offered by scenario block 2.

The SD model used in the training session consists of five interrelated sub-models (views):

1. *Production View:* This view represents a typical software development lifecycle consisting of the following chain of transitions: set of requirements (planned functionality) $\rightarrow$ design documents $\rightarrow$ code $\rightarrow$ tested code.

2. *Quality View:* This view models the defect co-flow: defect injection (into design or code) $\rightarrow$ defect propagation (from design to code) $\rightarrow$ defect detection (in the code during testing) $\rightarrow$ defect correction (only in the code). Optionally, additional QA activities will result in defect detection and rework already during design and coding.

3. *Effort View:* In this view, the total effort consumption for design development, code development, code testing, optional QA activities, and defect correction (rework) is calculated.

4. *Initial Calculations View:* In this view, the nominal value of the process parameter "productivity" is calculated using the basic COCOMO equations for estimating effort and project duration. The nominal productivity varies with assumptions about the product development mode (organic, semi-detached, embedded) and characteristics of the available project resources (e.g., developer skill).

5. *Productivity, Quality and Manpower Adjustment View:* In this view, project-specific process parameters, like (actual) productivity, defect generation, effectiveness of QA activities, etc., are determined based on (a) planned target values for manpower, project duration, product quality, etc. and (b) time pressure induced by unexpected rework or requirement changes.

A detailed description of the SD model used in the experiment can be found in Pfahl et al. (2001).

Table 3. List of principles dominating project performance.

No.	Principle
1	"Finding and fixing a software problem after delivery is 100 times more expensive than finding and fixing it during the requirements and early design phases."
2	"You can compress a software development schedule up to 25% of nominal, but no more."
3	"Software development and maintenance cost are primarily a function of the number of source lines of code (SLOC) in the product."
4	"Variations between people account for the biggest differences in software productivity."
5	"Software systems and products typically cost three times as much per SLOC as individual software programs. Software-system products (i.e., system of systems) cost nine times as much."
6	"Walkthroughs catch 60% of the errors."

Scenario block 2 (PM Role Play) has been designed to help the trainee understand the complex implications of a set of empirically derived principles that typically dominate software projects conducted according to the waterfall process model. Even though the waterfall process approach is no longer state-of-the art, and in most industrial organizations not even state-of-the-practice, it is expected that it is still an interesting object of study for students having little or no experience with real-world industrial software development. The set of principles used in the block scenario (cf. Table 3) was distilled from the top 10 list of software metric relationships published by Barry Boehm (1987).

In order to make the trainee understand the implications of these principles (and their combinations), a role-play is conducted in which the trainee takes the role of a project manager who has been assigned to a new development project. Several constraints are set, i.e., the size of the product and its quality requirements, the number of software developers available, and the project deadline.

The first thing to do for the project manager (in order to familiarize with the SD simulation model) is to check whether the project deadline is feasible under the resource and quality constraints given. Running a simulation does this check. From the simulation results, the project manager learns that the deadline is much too short. Now, the scenario provides a set of actions that the project manager can take, each action associated with one of the principles and linked to one of the model parameters. Soon the project manager learns that his department head does not accept all of the proposed actions (e.g., reducing the product size or complexity). Depending on the action the project manager has chosen, additional options can be taken. Eventually, the project manager finds a way to meet the planned deadline, e.g., by introducing code and design inspections (cf. Principle 6 in Table 3).

The role-play is arranged in a way that the project manager can only succeed when combining actions that relate to at least two of the principles listed in Table 3. At the end of the role-play, a short discussion of the different possible solutions is provided, explaining the advantages and disadvantages of each.

It should be noted that the complex impact of actions taken during the role-play by the fictitious project manager could not have been be calculated using the COCOMO model. This is mainly due to the fact that the COCOMO model does not (yet) fully cover the impact of effort or size changes on product quality.[1] The SD model used during the role-play has this capability, i.e., constraints on product quality have an impact on project duration and effort consumption, and vice-versa.

2.3.2. Treatment of the Control Group

The control group performed only scenario blocks 1, 3, and 4. The predictive model used in scenario blocks 3 and 4 was the intermediate COCOMO model. A detailed description of the COCOMO model can be found in Boehm (1981).

2.3.3. Differences Between Initial Experiment and Replication.

Since almost all of the participants of the initial study at the University of Kaiserslautern stated that they did not have enough time for working through the materials, more time was reserved for the treatment during the replication at the University of Oulu. Another difference refers to the overall schedule of the experiments. While the initial experiment at the University of Kaiserslautern was conducted on two days with one week of time in between, the replication at the University of Oulu was conducted on one single day.

2.4. Experimental Design

For evaluating the effectiveness of a training session using SD model simulation, a pre-test-post-test control group design was applied. This design involves random assignment of subjects to an experimental group (A) and a control group (B). The subjects of both groups complete a pre-test and a post-test. The pre-test measures the performance of the two groups before the treatment, and the post-test measures the performance of the two groups after the treatment. By studying the differences between the post-test and pre-test scores of the experimental group and the control group, conclusions can be drawn with respect to the effect of the treatment (i.e., the independent variable of the experiment) on the dependent variable(s) under study.

2.5. Experimental Variables

During the experiment, data for three types of variables are collected. Table 4 lists all experimental variables, including one independent variable ($X.1$), four dependent variables ($Y.1, \ldots, Y.4$), and three variables that represent potentially disturbing factors ($Z.1, \ldots, Z.3$).

The conceptual model underlying the proposed statistical analysis assumes that there are two separate effects on the dependent variables. On the one hand the effect

Table 4. Experimental variables.

Independent variable	
$X.1$	Type of treatment
Dependent variables	
$Y.1$	Interest in software project management issues ("Interest")
$Y.2$	Knowledge about typical behavior patterns of software development projects ("Knowledge")
$Y.3$	Understanding of "simple" project dynamics ("Understand simple")
$Y.4$	Understanding of "complex" project dynamics ("Understand complex")
Disturbing factors	
$Z.1$	Personal background (experience)
$Z.2$	Time consumption/time need
$Z.3$	Session evaluation (personal perception)

of the independent variable, and on the other the effect of additional potentially disturbing factors as shown in Figure 2.

2.5.1. Independent Variables

The independent variable $X.1$ (type of treatment) can have two values, either T_A, which is applied to the experimental group A, or T_B, which is applied to the control group B. The difference between T_A and T_B is basically determined by two factors. The first factor is the training scenario according to which the course material is presented. The second factor is the planning model that is used to support software project management decision-making. Table 5 briefly summarizes the differences between the treatment of the experimental group and the treatment of the control group, indicating the duration of the scenario blocks applied, and providing information on the nature of the used planning models. The duration is expressed in minutes, first for the initial experiment, then for the replication.

With regard to the scenario, the main difference consists in the application of scenario block PM Role Play for treatment T_A. As a consequence of performing the

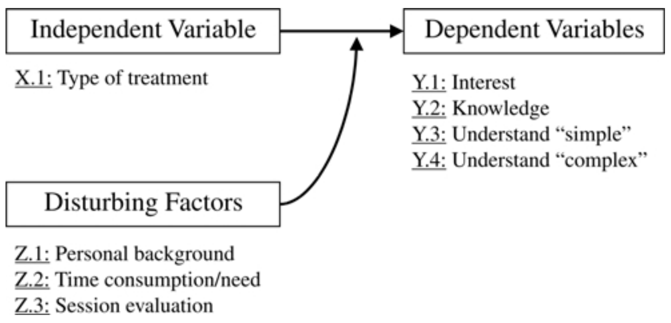


Figure 2. Relations between experimental variables.

Table 5. Differences between treatments.

	Treatment T_A	Treatment T_B
Scenario	Block 1 – (3 min; 5 min)	Block 1 – (3 min; 5 min)
	Block 2 – (15 min; 30 min)	n/a
	Block 3 – (15 min; 30 min)	Block 3 – (30 min; 60 min)
	Block 4 – (12 min; 15 min)	Block 4 – (12 min; 15 min)
PM model	SD model:	COCOMO model:
	• behavioral (white box)	• point estimates (black box)
	• point estimates (black box)	

scenario block PM Role Play, interaction of the trainee with the training module will be high whereas treatment T_B will only trigger low interaction of the trainee with the training module. With regard to the model that is used during the training session, treatment T_B exclusively relies on a black-box model providing point estimates, such as COCOMO. In contrast to this, by using a SD simulation model, treatment T_A is based on a white-box model that in addition to point estimates facilitates insights into behavioral aspects of software projects.

2.5.2. Dependent Variables

The dependent variables *Y*.1, *Y*.2, *Y*.3, and *Y*.4 are determined by analyzing data collected through questionnaires that all subjects have to fill in, the first time during the pre-test, and the second time during the post-test. Each dependent variable was represented by 5 to 7 questions where answers have to be provided on a uniform scale. The value of each dependent variable will then be equal to the average score derived from the related questionnaire. The contents of the questionnaires are as follows:

- *Y*.1 (“Interest”): Questions about personal interest in learning more about software project management.
- *Y*.2 (“Knowledge”): Questions about typical performance patterns of software projects. These questions are based on some of the empirical findings and lessons learned summarized in Barry Boehm’s top 10 list of software metric relations (Boehm, 1987).
- *Y*.3 (“Understand simple”): Questions on project planning problems that require simple application of the provided PM models, addressing trade-off effects between no more than two model variables.
- *Y*.4 (“Understand complex”): Questions on project planning problems addressing trade-off effects between more than two variables, and questions on planning problems that may require re-planning due to alterations of project constraints

(e.g., reduced manpower availability, shortened schedule, or changed requirements) during project performance.

2.5.3. Disturbing Factors

The values of the three potentially disturbing factors *Z*.1, *Z*.2, and *Z*.3 are also derived from questionnaires that all subjects have to fill in. The questionnaire for *Z*.1 must be filled in before the pre-test, the questionnaires for *Z*.2 and *Z*.3 have to be filled in after the post-test. The contents of the questionnaires are as follows:

- *Z*.1: Questions about personal characteristics (age, gender), university education (number of terms, major, minor), practical software development experience, software project management literature background, and preferred learning style.
- *Z*.2: Questions on actual time consumption per scenario block, and on perceived time need.
- *Z*.3: Questions on personal judgment of the training session (subjective session evaluation).

2.6. Experimental Procedure

The initial experiment and its first replication were conducted following the plan presented in Table 6. After a short introduction during which the purpose of the

Table 6. Schedule of experiment and replication.

	Experiment	Replication
Introduction to experiment	5 min	5 min
Background characteristics	5 min	5 min
Pre-test		
Interest	3 min	5 min
Knowledge about empirical patterns	5 min	5 min
Understanding of simple project dynamics	10 min	10 min
Understanding of complex project dynamics	12 min	15 min
Introduction to treatments	5 min	5 min
Random assignment of subjects to groups	5 min	5 min
Treatment	45 min	80 min
Post-test		
Interest	3 min	5 min
Knowledge about empirical patterns	5 min	5 min
Understanding of simple project dynamics	10 min	10 min
Understanding of complex project dynamics	12 min	15 min
Time need and subjective session evaluation	5 min	10 min
Total	130 min	180 min

experiment and general organizational issues were explained, data on the background characteristics (variable $Z.1$) was collected with the help of a questionnaire. Then the pre-test was conducted and data on all dependent variables ($Y.1$ through $Y.4$) was collected, again using questionnaires. Following the pre-test, a brief introduction into organizational issues related to the treatments was given. After that, the subjects were randomly assigned to either the experimental or control group. Then each group underwent its specific treatment. After having concluded their treatments, both groups completed the post-test using the same set of questionnaires as during the pre-test, thus providing data on the dependent variables for the second time. Finally, the subjects got the chance to evaluate the training session by filling in another questionnaire, providing data on variables $Z.2$ and $Z.3$. The time frames reserved for completing a certain step of the schedule was identical for the experimental and control groups. However, more time was reserved during the replication as compared to the initial experiment (cf. columns Experiment and Replication of Table 6).

The initial experiment was conducted in two days. On the first day, the steps "Introduction to experiment", "Background characteristics", and "Pre-test" were conducted, consuming a total of 40 min. On the second day, the steps "Introduction to treatments", "Random assignment of students to groups", "Treatment", "Post-test", and "Time need and subjective session evaluation" were conducted, consuming a total of 90 min.

The replication was conducted in two parts on one day. The first part, including the steps "Introduction to experiment", "Background characteristics", and "Pre-test" were conducted, consumed a total of 45 min. The second part, including the steps "Introduction to treatments", "Random assignment of students to groups", "Treatment", "Post-test", and "Time need and subjective session evaluation", consumed a total of 135 min. There was a break of 30 min between the first and the second part.

Of the 12 students that agreed to participate in the initial experiment, nine students participated in both pre-test and post-test. Five students were assigned randomly to the experimental group (A), and four students to the control group (B). Of the twelve students participating in the first replication, six were assigned randomly to the experimental group (A), and six to the control group (B).

2.7. *Data Collection Procedure*

The raw data for dependent variables $Y.1$ to $Y.4$ were collected during pre-test and post-test with the help of questionnaires (the full set of questionnaires used in the experiments can be found in Pfahl (2001). The values for variable $Y.1$ ("Interest") are average scores derived from five questions on the student's opinion about the importance of software project management issues (i) during university education and (ii) during performance of industrial software development projects, applying a five-point Likert-type scale (Likert, 1932). Each answer in the questionnaire is mapped to the value range $R = [0, 1]$ assuming equidistant distances between possible answers,[2] i.e., "fully disagree" is encoded as 0, "disagree" as 0.25, "undecided" as 0.5, "agree" as 0.75, and "fully agree" as 1.

The values for variables $Y.2$ ("Knowledge"), $Y.3$ ("Understand simple"), and $Y.4$ ("Understand complex") are average scores derived from five (for $Y.2$), seven (for $Y.3$), and six (for $Y.4$) questions in multiple-choice style. The answers to these questions were evaluated according to their correctness, thus having a binary scale with correct answers encoded as 1, and incorrect answers encoded as 0. Missing answers were encoded like incorrect answers.

The raw data for disturbing factors were collected before pre-test ($Z.1$) and after post-test ($Z.2$ and $Z.3$). In order to determine the values of factor $Z.1$ ("Personal background") information on gender, age, number of terms studied, subjects studied (major and minor), personal experience with software development, and number of books read about software project management was collected. In order to simplify the analysis, the actual values for factor $Z.1$ eventually used for the statistical analysis were exclusively based on the normalized average scores derived from the questions on the student's personal experience with software development. Each of these questions could be answered with "yes" (encoded as 1) or "no" (encoded as 0). "Yes" indicated that a certain type of experience was present, and "no", that it was not present. Therefore, simple adding of the scores per answer gives a measure of experience with a maximal score of 6 and a minimal score of 0. Dividing by the number of questions provides a normalized value range, i.e., range $R = [0, 1]$.

The values for factor $Z.2$ are normalized average scores reflecting the "time need" for reading and understanding of the scenario blocks 1, 3, and 4, for familiarization with the supporting tools, and for filling in the post-test questionnaire. For group A, the variable $Z.2'$ includes also scores related to scenario block 2. If a subject wants to express that more than the available time was needed related to a certain task, then "yes" (encoded as 1) should be marked, otherwise "no" (encoded as 0). Adding the scores and dividing them by the number of tasks once again provides a normalized value range $R = [0, 1]$, with 1 indicating time need for all tasks and 0 indicating the absence of time need.

The values for factor $Z.3$ ("Session evaluation") are based on subjective measures reflecting the quality of the treatment. Again, for group A, the variable $Z.3'$ includes scores related to scenario block 2. The subjective perception of the treatment quality was evaluated with regard to four dimensions ("useful" versus "useless", "absorbing" versus "boring", "easy" versus "difficult", and "clear" versus "confusing") using five-point Likert-type scales, e.g. "extremely boring", "boring", "undecided", "absorbing", "extremely absorbing". Similar to variable $Y.1$, possible answers were encoded as 0, 0.25, 0.5, 0.75, and 1 depending on whether the subjective judgment was very negative, negative, undecided, positive, or very positive. By taking the average of the values for all four questions the values for disturbing factors could be mapped to range $R = [0, 1]$.

2.8. Data Analysis Procedure

The conceptual model underlying the proposed statistical analysis is inspired by the work of Jac Vennix who conducted a similar experiment (Vennin, 1990). He assumes that there are two separate effects on the dependent variables: On the one hand the effect of the independent variable, and on the other the effect of additional potentially disturbing factors (cf. Figure 2).

In a first step of the statistical analysis a *t*-test was used to investigate the effect of the independent variable $X.1$ on the dependent variables $Y.1$ to $Y.4$. For testing hypothesis $H_{0,1}$, a one-way paired *t*-test was used, because the data collected for this hypothesis is within-subjects, i.e., post-test scores are compared to pre-test scores of subjects within the same group (Sheskin, 1997). For testing hypotheses $H_{0,2a}$ and $H_{0,2b}$, the appropriate test was a one-sided *t*-test for independent samples (Sheskin, 1997).

In addition, analysis of covariance (ANCOVA) was applied for testing hypotheses $H_{0,2a}$ and $H_{0,2b}$, to improve the precision of the statistical analysis by removing potential bias due to disturbing factors (Wildt and Ahtola, 1978). Since these additional analyzes did not alter any of the results from the *t*-tests, the ANCOVA results are not presented in this paper. The ANCOVA results for the initial experiment can be found in Pfahl et al. (2001).

A prerequisite for applying the *t*-test is the assumption of normal distribution of the variables in the test samples. Therefore, a test to check this assumption was conducted. While the results of the *t*-test are often robust against violation of the normality assumption it is strongly influenced by outliers in the data sets. Hence, an analysis to detect the presence of outliers was performed. Checking for the normality assumption showed that no normal distribution of the variables in the test samples could be assumed. On the other hand, the outlier analysis showed that all data points lie within the range of ± 2 standard deviations around the samples' means, and in most cases even within the range of ± 1.5 standard deviations around the samples' means. Although no outliers were detected, additional non-parametric tests were conducted to re-confirm the findings of the *t*-tests. Since the additional tests (Wilcoxon matched pair test for hypothesis $H_{0,1}$ and Mann-Whitney U test for hypotheses $H_{0,2a}$ and $H_{0,2b}$) did not yield any differences to the results of the *t*-tests, these test results are not presented in the paper.

Ideally, researchers should perform a power analysis (Cohen, 1988) before conducting a study to ensure the experimental design will find a statistically significant effect if one exists. The power of a statistical test is dependent on three different components: significance level α, the size of the effect being investigated, and the number of subjects. Low power will have to be considered when interpreting non-significant results.

Usually, the commonly accepted practice is to set $\alpha = 0.05$. However, controlling a Type I error (α) and Type II error (β) requires either a large effect size or large sample sizes. This represents a dilemma in a software-engineering context since much research in this area involves relatively modest effects sizes, and in general, small sample sizes. As pointed out in Lipsey (1990), if neither effect size nor sample size can be increased to maintain a low risk of error, the only remaining strategy—other than abandoning the research altogether—is to permit higher risk of error. Since sample sizes were rather small in the initial experiment and its first replication, and no sufficiently stable effect sizes from previous empirical studies were known, it was decided to set $\alpha = 0.1$.

Effect size is expressed as the difference between the means of the two samples divided by the root mean square of the variances of the two samples (Sheskin, 1997). For this exploratory study, effects where $\gamma > 0.5$ are considered to be of practical significance. This decision was made on the basis of the effect size indices proposed by Cohen (1988).

3. Experimental Results

In the following sub-sections the descriptive statistics of the dependent variables and disturbing factors are summarized. Then, for each experimental hypothesis the results of the statistical analyzes are presented and briefly discussed.

3.1. Descriptive Statistics

The column "Pre-test scores" of Table 7 shows the calculated values for mean, median, and standard deviation of the raw data collected during the pre-test of the initial experiment (E) and the first replication (R).

Table 7. Scores of dependent variables (E and R).

E: Initial experiment KL												
	Pre-test scores				Post-test scores				Difference scores			
Group A (5 subj.)	*Y*.1	*Y*.2	*Y*.3	*Y*.4	*Y*.1	*Y*.2	*Y*.3	*Y*.4	*Y*.1	*Y*.2	*Y*.3	*Y*.4
Mean	0.69	0.56	0.31	0.37	0.79	0.84	0.66	0.43	0.10	0.28	0.34	0.07
Median	0.75	0.60	0.29	0.33	0.85	0.80	0.71	0.33	0.10	0.40	0.43	*0.00*
Stdev	0.18	0.30	0.26	0.25	0.19	0.17	0.13	0.32	0.09	0.36	0.28	0.19
Group B (4 subj.)	*Y*.1	*Y*.2	*Y*.3	*Y*.4	*Y*.1	*Y*.2	*Y*.3	*Y*.4	*Y*.1	*Y*.2	*Y*.3	*Y*.4
Mean	0.81	0.50	0.43	0.33	0.79	0.60	0.82	0.46	*−0.03*	0.10	0.39	0.13
Median	0.78	0.50	0.36	0.25	0.80	0.60	0.86	0.50	*0.00*	0.10	0.50	0.17
Stdev	0.13	0.26	0.31	0.24	0.19	0.16	0.07	0.37	0.09	0.35	0.38	0.34
R: First replication – Oulu												
	Pre-test scores				Post-test scores				Difference scores			
Group A (6 subj.)	*Y*.1	*Y*.2	*Y*.3	*Y*.4	*Y*.1	*Y*.2	*Y*.3	*Y*.4	*Y*.1	*Y*.2	*Y*.3	*Y*.4
Mean	0.83	0.57	0.41	0.44	0.85	0.97	0.67	0.44	0.03	0.40	0.26	*0.00*
Median	0.88	0.60	0.43	0.42	0.85	1.00	0.57	0.50	0.03	0.40	0.14	*0.00*
Stdev	0.14	0.23	0.11	0.23	0.15	0.08	0.27	0.09	0.07	0.28	0.25	0.24
Group B (6 subj.)	*Y*.1	*Y*.2	*Y*.3	*Y*.4	*Y*.1	*Y*.2	*Y*.3	*Y*.4	*Y*.1	*Y*.2	*Y*.3	*Y*.4
Mean	0.70	0.47	0.33	0.33	0.78	0.43	0.74	0.33	0.08	*−0.03*	0.41	*0.00*
Median	0.73	0.40	0.36	0.25	0.83	0.40	0.79	0.33	0.08	*0.00*	0.43	0.08
Stdev	0.18	0.21	0.17	0.30	0.21	0.15	0.17	0.24	0.13	0.32	0.19	0.45

Table 8. Scores of disturbing factors (*E* and *R*).

E: Initial experiment – KL					
Group A—experimental	$Z.1$	$Z.2$	$Z.2_{B2}$	$Z.3$	$Z.3_{B2}$
Mean	0.48	0.44	0.2	0.41	0.68
Median	0.4	0.4	0	0.38	0.69
Stdev	0.23	0.46	0.45	0.09	0.26
Group B—control	$Z.1$	$Z.2$		$Z.3$	
Mean	0.6	0.35		0.66	
Median	0.6	0.3		0.69	
Stdev	0.23	0.19		0.06	
R: First replication – Oulu					
Group A—experimental	$Z.1$	$Z.2$	$Z.2_{B2}$	$Z.3$	$Z.3_{B2}$
Mean	0.61	0.17	0	0.35	0.82
Median	0.8	0.2	0	0.38	0.82
Stdev	0.43	0.15	0	0.20	0.17
Group B—control	$Z.1$	$Z.2$		$Z.3$	
Mean	0.62	0.25		0.71	
Median	0.57	0.25		0.75	
Stdev	0.28	0.19		0.18	

The column “Post-test scores” of Table 7 shows the calculated values for mean, median, and standard deviation of the raw data collected during the post-test of the initial experiment (E) and the first replication (R).

The column “Difference scores” of Table 7 shows the calculated values for mean, median, and standard deviation of the differences between post-test and pre-test scores of the initial experiment (E) and the first replication (R). The italicized entries in the table indicate that the difference between average post-test scores and average pre-test scores is zero or even negative, i.e., based on average data no (or even negative) relative learning effect was observed. This phenomenon occurred two times during the initial experiment (variables $Y.4$ (A) and $Y.1$ (B)) and three times during the first replication (variables $Y.4$ (A and B) and $Y.2$ (B)). Possible reasons for these unexpected outcomes are discussed in Section 3.6.

Table 8 shows the calculated values for mean, median, and standard deviation of the raw data collected for the disturbing factors during the initial experiment (E) and the first replication (R).

In the initial experiment, there could be observed a difference between students in the experimental group A and the control group B regarding experience with software development ($Z.1$). This difference could not be observed in the first replication.

Table 9. Subjective evaluation of Scenario Block 2 (E and R).

Group A—experimental (E)	Useful	Absorbing	Easy	Clear
$\text{Mean}_{Z.2/\text{B2}}$	0.75	0.6	0.65	0.7
$\text{Median}_{Z.2/\text{B2}}$	0.75	0.5	0.75	0.75
$\text{Stdev}_{Z.2/\text{B2}}$	0.31	0.29	0.22	0.27
Group A—experimental (R)	**Useful**	**Absorbing**	**Easy**	**Clear**
$\text{Mean}_{Z.2/\text{B2}}$	0.88	0.85	0.79	0.75
$\text{Median}_{Z.2/\text{B2}}$	0.88	1.00	0.75	0.75
$\text{Stdev}_{Z.2/\text{B2}}$	0.14	0.22	0.19	0.27

In the initial experiment, students in the control group (B) expressed less need of additional time ($Z.2$) for conducting the treatment and completing the tests than did students in the experimental group (A). In the first replication, both groups (A and B) expressed less need of additional time to conduct scenario blocks 1, 3, and 4 than the students in control group (B) of the initial experiment. This is also true when looking at the time need expressed by students of the experimental group (A) with regards to conducting scenario block 2 (Role Play/$Z.2_{\text{B2}}$).

Finally, in both initial experiment and first replication, students in the control group (B) on average perceived their treatment easier, clearer, more absorbing, and more useful ($Z.3$) than the students in the experimental group (A). This evaluation, however, relates only to those scenario blocks that are conducted by both groups, i.e., blocks 1, 3, 4. When looking at the evaluation of scenario block 2 (Role Play/$Z.3_{\text{B2}}$) separately, high scores can be observed for the initial experiment, and even higher scores for the first replication (cf. Table 9 for detailed evaluation results).

Table 10. Results for "post-test" vs. "pre-test" (E).

Group A—experimental (five subjects)					
Variable	γ	df	t-value	Crit. $t_{0.90}$	p-value
$Y.1$—interest	1.07	4	2.39	1.53	0.04
$Y.2$—knowledge	0.77	4	1.72	1.53	0.08
$Y.3$—understand "simple"	1.23	4	2.75	1.53	0.03
$Y.4$—understand "complex"	0.35	4	0.78	1.53	0.24
Group B—control (four subjects)					
Variable	γ	df	t-value	Crit. $t_{0.90}$	p-value
$Y.1$—interest	**	3	– 0.58	1.64	**
$Y.2$—knowledge	0.29	3	0.58	1.64	0.30
$Y.3$—understand "simple"	1.05	3	2.09	1.64	0.06
$Y.4$—understand "complex"	0.37	3	0.73	1.64	0.26

Table 11. Results for "post-test" vs. "pre-test" (R).

Group A—experimental (six subjects)					
Variable	γ	df	t-value	Crit. $t_{0.90}$	p-value
Y.1—interest	0.36	5	0.89	1.48	0.21
Y.2—knowledge	1.41	5	3.46	1.48	0.01
Y.3—understand "simple"	1.06	5	2.61	1.48	0.02
Y.4—understand "complex"	**	5	− 0.00017	1.48	**
Group B—control (six subjects)					
Variable	γ	df	t-value	Crit. $t_{0.90}$	p-value
Y.1—interest	0.65	5	1.58	1.48	0.09
Y.2—knowledge	**	5	− 0.25	1.48	**
Y.3—understand "simple"	2.13	5	5.22	1.48	0.0017
Y.4—understand "complex"	0.00	5	0.00	1.48	0.50

3.2. *Hypothesis $H_{0,1}$*

Table 10 shows for each group (A and B) separately the initial experiment's (E) results of testing hypothesis $H_{0,1}$ using a one-tailed t-test for dependent samples. Column one represents the dependent variable, column two the size of the effect detected, column three the degrees of freedom, column four the t-value of the study, column five the critical value for $\alpha = 0.10$ (as discussed in Section 2.8) which the t-value has to exceed to be statistically significant, and column six provides the associated p value.

By examining columns four and five of Table 10, one can see that group A achieved significant results for dependent variables Y.1, Y.2, and Y.3, and group B for dependent variable Y.3. It is worth noting though that the values for dependent variable Y.4 support the direction of the expected positive learning effect in both groups, however without showing an effect size of practical significance. In addition, for group B, values for dependent variable Y.2 also support the direction of the expected positive learning effect, but again without showing an effect size of practical significance. The values for variable Y.1 do not even support the direction of the expected learning effect in group B.

Table 11 shows for each group (A and B) separately the first replication's (R) results of testing hypothesis $H_{0,1}$ again using a one-tailed t-test for dependent samples. One can see that that group A achieved significant results for dependent variables Y.2, and Y.3, and group B for dependent variables Y.1 and Y.3. It is worth noting though that the values for dependent variable Y.1 support the direction of the expected positive learning effect in group A, however, without showing an effect size of practical significance. The value for variable Y.2 do not even support the direction of the expected learning effect in group B, and the values for variable Y.4 do not even support the direction of the expected learning effects in both groups. As a consequence, in the replication, no further testing of hypothesis $H_{0,2}$ was done for this variable.

3.3. *Hypothesis $H_{0,2a}$*

Table 12 shows the initial experiment's (E) results of testing hypothesis $H_{0,2a}$ using a one-tailed *t*-test for independent samples.

For significance level $\alpha = 0.1$, the score difference between post-test and pre-test for variable *Y*.1 is significantly larger in group A as compared to group B, and thus hypothesis $H_{0,2a}$ can be rejected. It can also be noted that the values of variable *Y*.2 support the direction of the expected relative learning effect, showing a medium to large effect size. The values for variables *Y*.3 and *Y*.4 do not even support the direction of the expected relative learning effect.

Table 13 shows the first replication's (R) results of testing hypothesis $H_{0,2a}$. As can be seen, only the score difference of variable *Y*.2 is significantly larger in group A as compared to group B, and thus hypothesis $H_{0,2a}$ can be rejected for this variable. For variables *Y*.1 and *Y*.3 the values do not even support the direction of the expected relative learning effect. No useful analysis could be conducted for variable *Y*.4, because no difference between post-test and pre-test groups could be observed neither in group A nor in group B.

3.4. *Hypothesis $H_{0,2b}$*

Table 14 shows the initial experiment's (*E*) results of testing hypothesis $H_{0,2b}$ using a one-tailed *t*-test for independent samples.

For significance level $\alpha = 0.1$, the post-test scores of variable *Y*.2 are significantly larger for the experimental group (A) as compared to the control group (B), and thus hypothesis $H_{0,2b}$ can be rejected. It can also be noted that the values of variable *Y*.1

Table 12. Results for "performance improvement" (E).

Group A (experimental) vs. B (control)					
Variable	γ	df	*t*-value	Crit. $t_{0.90}$	*p*-value
Y.1—interest	1.38	7	2.06	1.42	0.04
Y.2—knowledge	0.51	7	0.75	1.42	0.24
Y.3—understand "simple"	**	7	− 0.23	1.42	**
Y.4—understand "complex"	**	7	− 0.33	1.42	**

Table 13. Results for "performance improvement" (R).

Group A (experimental) vs. B (control)					
Variable	γ	df	*t*-value	Crit. $t_{0.90}$	*p*-value
Y.1—interest	**	10	− 0.98	1.37	**
Y.2—knowledge	1.43	10	2.48	1.37	0.02
Y.3—understand "simple"	**	10	− 1.13	1.37	**

Table 14. Results for "post-test performance" (E).

Group A (experimental) vs. B (control)					
Variable	γ	df	t-value	Crit. $t_{0.90}$	p-value
$Y.1$—interest	0.01	7	0.02	1.42	0.49
$Y.2$—knowledge	1.45	7	2.16	1.42	0.03
$Y.3$—understand "simple"	**	7	− 2.28	1.42	**
$Y.4$—understand "complex"	**	7	− 0.11	1.42	**

Table 15. Results for "post-test performance" (R).

Group A (experimental) vs. B (control)					
Variable	γ	df	t-value	Crit. $t_{0.90}$	p-value
$Y.1$—interest	0.36	10	0.63	1.37	0.27
$Y.2$—knowledge	4.40	10	7.63	1.37	0.000009
$Y.3$—understand "simple"	**	10	− 0.56	1.37	**

support the direction of the expected absolute learning effect, however, only with a very small effect size. The values for variables $Y.3$ and $Y.4$ do not even support the direction of the expected absolute learning effect.

Table 15 shows the first replication's (R) results of testing hypothesis $H_{0,2b}$ using a one-tailed t-test for independent samples.

As in the initial experiment, the post-test scores of variable $Y.2$ are significantly larger for the experimental group (A) as compared to the control group (B), and thus alternative hypothesis $H_{0,2b}$ can be rejected. It can also be noted that the values of variable $Y.1$ support the direction of the expected absolute learning effect, again without showing an effect size of practical significance. The values for variable $Y.3$ do not even support the direction of the expected absolute learning effect. No useful analysis could be conducted for variable $Y.4$, because no difference between post-test and pre-test groups could be observed neither in group A nor in group B, and thus differences in post-test scores are irrelevant for evaluating the absolute learning effect.

3.5. Qualitative Results

In addition to filling in the pre-test and post-test questionnaires and the questionnaires about potential disturbing factors, the participants of the case studies had the chance make comments or improvement suggestions, and could raise issues or problems that they encountered during the treatments. Apart from some improvement suggestions related to technical aspects of the role-play and tool usage, comments and problem statements mainly supported the findings of the quantitative analyzes. Positive comments mainly correlated with the high scores for scenario block 2 in the subjective evaluation of the experimental group. In addition, positive statements were made about the clarity of the presentation of the COCOMO model and its usefulness. Negative comments or problem statements mainly addressed the difficulty of understanding the structure of the SD model, and the lack of time for getting acquainted with the tools, for working through the treatments, and for answering the questions in the pre-test and post-test questionnaires. The time issue, however, was less prominent during the replication.

3.6. Analysis Summary and Discussion

Table 16 shows the main results of the initial experiment and its first replication. Only the results for comparing relative and absolute learning effects between the experimental and control groups are shown for each dependent variable (testing of null hypotheses $H_{0,2a}$ and $H_{0,2b}$, respectively). Since there was no positive difference between post-test and pre-test scores within both groups in the replication, no results of the comparison between groups are shown for variable $Y.4$.

In behavioral research, meta-analysis techniques are used for comparing and combining results from different studies. The benefit of meta-analytic procedures is that by combining the results of a number of studies, one can increase the power of the statistical analysis. This enables one to identify effects that could escape the scrutiny in a single study with much lower statistical power. Meta-analytic techniques are based either on p-values or effect sizes. To make a step in this direction and include both, p-values as well as effect sizes, in the discussion, the hypothesis testing results of each study were classified as follows:

- Statistical significance (sta. sig.): null hypothesis could be rejected at significance level $\alpha = 0.1$.
- Practical significance (pract. sig.): null hypothesis could not be rejected but effect size $\gamma \geq 0.5$.
- Positive effect (+): no practical significance could be observed but effect size $\gamma > 0$. The number in parentheses indicates how many subjects would have been needed to achieve statistical significance with the given effect size.
- No effect or negative effect (—): t-value ≤ 0.

Table 16. Results for $H_{0,2}$.

Variable	Experiment		Replication	
	$H_{0,2a}$	$H_{0,2b}$	$H_{0,2a}$	$H_{0,2b}$
$Y.1$—interest	*stat. sig.*	+ (1000)	—	+ *(56)*
$Y.2$—knowledge	pract. sig.	*stat. sig.*	stat. sig.	*stat. sig.*
$Y.3$—understand "simple"	—	—	—	—
$Y.4$—understand "complex"	—	—		

The italicized entries indicate for each study which learning effect was stronger, the relative learning effect (hypothesis $H_{0,2a}$) or the absolute learning effect (hypothesis $H_{0,2b}$). When discussing the results of both studies in combination, two main trends can be observed. First, there is no indication that the experimental group performs better than the control group with regard to understanding of simple and complex project dynamics (variables *Y*.3 and *Y*.4). Second, there is strong support for the expectation that subjects in the experimental group perform better than subjects in the control group with regard to knowledge about empirical patterns in software projects (variable *Y*.2), and there is weak support for the expectation that subjects in the experimental group show higher interest in software project management than subjects in the control group (variable *Y*.1).

The findings of the analyses showed several interesting trends. First, both treatments involving the SD model (i.e., experimental group A) had a positive impact on the change of scores from pre-test to post-test for dependent variables *Y*.1 to *Y*.3. The effect was statistically significant for *Y*.2 and *Y*.3 (cf. Tables 10 and 11). For *Y*.1, in the replication, the power of the test seemed to be too low to be able to detect the effect at the selected significance level $\alpha = 0.1$.

Second, the outcomes of hypothesis testing of learning effectiveness indicate that treatment involving the SD model is significantly better regarding variable *Y*.2. Regarding variable *Y*.1 the positive results of the initial treatment could not be observed in the replication.

Though positive, the second result might be related to the inclusion of the role-play (scenario block 2) exclusively for the experimental group A. Inclusion of the role-play, on the other hand, imposed additional time pressure on the subjects in the experimental group, which might have resulted in low scores for questions related to dependent variables *Y*.3 and—particularly—*Y*.4. Although more time was allowed during replication, and the values for time need were lower than in the initial experiment, qualitative results from the replication still support this subjective feeling of time pressure. Hence, a further replication should allow even more time for executing scenario blocks 2 (PM Role Play) and 3 (PM Planning Models), and for the familiarization with the simulation tool. Moreover, the experimental treatment, as it is now, does not yet fully exploit all potentially available features that learning through SD model usage and SD model building could offer. This relates to the fact that SD models not only make causal relationships explicit and allow for variation of the strength of the relationships, but also offer means to change the structure of these relationships and make the effects of such changes on project performance visible through simulation.

A closer look at the nature of the applied treatments also proposes an improved experimental design for future replications. The inclusion of a role-play (scenario block 2) in the experimental treatment had two consequences. As mentioned before, the role-play in scenario block 2 provided information on typical patterns of software project behavior to subjects in group A, which was not given in such an explicit form to group B subjects. This might explain why group A subjects had clearly better scores for variable *Y*.2 than group B subjects. On the other hand, group B subjects had more time than group A subjects to read and understand the information provided in scenario block 3 (PM Planning Models) because they did not have to perform a role-play. This might explain why group A subjects did not have better scores for variables *Y*.3 and *Y*.4. Another reason for difficulties of the experimental group with variables *Y*.3 and *Y*.4 might be that the presentation of the SD model in scenario block 3 was too hard to grasp, due to the high complexity of the model structure. Some subjects mentioned this issue in the debriefing questionnaires.

4. Threats to Validity

This section discusses the various threats to validity of the study.

4.1. Construct Validity

Construct validity is the degree to which the variables used in the study accurately measure the concepts they purport to measure. The following issues associated with construct validity have been identified:

1. The mere application of a SD model might not adequately capture the specific advantages of SD models over conventional planning models, since it has often been claimed that model building—and not the application of an existing model—is the main benefit of SD simulation modeling (Lane, 1995).

2. Interest in a topic and evaluation of a training session are difficult concepts that have to be captured with subjective measurement instruments. To counteract this threat to validity in the studies, the instruments for measuring variables *Y*.1 and *Z*.3 were derived from measurement instruments that had been successfully applied in a similar study (Vennix, 1990).

3. There are indications that the distinction between "simple dynamics" and "complex dynamics", as it was made for measuring variables *Y*.3 and *Y*.4 (cf. Section 2.5.2.), was too simplistic.

4. It is difficult to avoid "unfair" comparison between SD models and COCOMO, because SD models offer features that per definition are not available for COCOMO (e.g., simulation of parameter changes over time/on-the-fly modification of model assumptions, etc.). In addition, only scenario block 2 explicitly provides information about typical behavior patterns of software projects, because this is an important prerequisite for conducting the role-play. Since exclusively subjects of the experimental group perform scenario block 2, subjects of the control group might be disadvantaged.

4.2. Internal Validity

Internal validity is the degree to which conclusions can be drawn about the causal effect of the independent variable on the dependent variables. Potential threats include selection effects, non-random subject loss, instrumentation effect, and maturation effect.

1. A selection effect was avoided by random assignment of subjects. In addition, existing differences in ability between groups were captured by collecting pre-test scores and by measuring the level of experience of subjects through variable $Z.1$. Any potential bias induced by differences in pre-test scores and experience were tried to be neutralized by using ANCOVA.

2. Non-random drop-out of subjects has been avoided by the experimental design, i.e., assignment of groups only on the second day of the experiment, i.e., directly before the treatment, and not before the pre-test already on the first day of the experiment.

3. The fact that the treatments of group A and B were different in the number of scenario blocks involved and, as a consequence, in the time available to perform each scenario block, may have induced an instrumentation effect. The post-mortem evaluation of the experiment with regard to time requirements indicated in the initial experiment that most subjects of group A did not have enough time to execute both scenario blocks 2 and 3. Due to more relaxed schedules, this effect could be reduced. In addition,—even though this was addressed by careful design—the planning models used in both treatments might slightly differ in scope and handling.

4. A maturation effect could have been caused if subjects had been informed before or during pre-test that at the end of the experiment they will complete a post-test with exactly the same questions. Since this information was not given to the subjects, and all materials were re-collected after the pre-test, it can be assumed that a maturation effect did not occur.

4.3. External Validity

External validity is the degree to which the results of the research can be generalized to the population under study and other research settings. Two possible threats have been identified: subject representativeness and materials:

1. The subjects participating in the experiment were all students in computer science or related fields at an advanced level. It can be expected that the results of the study are to some degree representative for this class of subjects. Any

generalization of the results with regard to education of novice students, or even with regard to training of software professionals should be done with caution.

2. Even when the training sessions are applied to students, adequate size and complexity of the applied materials might vary depending on previous knowledge about SD modeling and COCOMO.

In any case, the point should be emphasized that the presented research at its current stage is exploratory of nature and just the first step of a series of experiments, which—after modification of the treatments and stepwise inclusion of subjects with different backgrounds—might yield more generalisable results in the future.

5. Conclusion

The empirical studies presented in this paper investigated the effect of using a SD simulation model to assist software project management education of computer science students. The treatment focused on problems of project planning and control. The performance of the students was analyzed with regard to four dimensions, i.e., interest in the topic of software project management ($Y.1$), knowledge of typical project behavior patterns ($Y.2$), understanding of simple project dynamics ($Y.3$), and understanding of complex project dynamics ($Y.4$). This was done by comparing the test results of students who performed a training session using the SD model (with web-based role-play) to the test results of students who performed a training session using the COCOMO model (without web-based role-play). Although the two studies seem to be heterogeneous, the findings of the initial experiment were corroborated by the first replication. SD models increase the interest of the subject in software project management and also improve a subject's knowledge of typical behavior patterns. Hence, SD models represent a viable path for learning multi-causal thinking in software project management. This was supported by the subjective evaluation of the role-play scenario involving simulation with the SD model, which received very high scores.

Although the results of the two studies are promising, further replication is required for two reasons. First, a single study even if replicated only provides a starting point for investigation. In this case, the studies were exploratory in nature. Based on the presented results, a further replication should consider the examination of cause/effect relationships. And second, each empirical study exhibits specific threats to validity, which can only be ruled out by replication. Additional replications of this study are currently planned.

Acknowledgment

Part of this work was funded by the European Union under grant IST-1999-11634.

Notes

1. This shortcoming might be resolved soon. In Boehm et al. (2000) Ray Madachy and Barry Boehm announce the integration of System Dynamics into COCOMO.
2. This is a common assumption in experimental software engineering and social science.

References

Abdel-Hamid, T. K., and Madnick, S. E. 1991. *Software Project Dynamics—an Integrated Approach*. Prentice-Hall.

Boehm, B. W. 1981. *Software Engineering Economics*. Englewood Cliffs: Prentice-Hall.

Boehm, B. W. September 1987. Industrial software metrics top 10 list. *IEEE Software*: 84–85.

Boehm, B. W., Abts, C., Brown, W. A., Chulani, S., Clark, B. K., Horowitz, E., Madachy, R., Reifer, D. J., and Steece, B. 2000. *Software Cost Estimation with COCOMO II*. Upper Saddle River: Prentice Hall PTR.

Briand, L. C., Bunse, C., Daly, J. W., and Differding, C. 1997. An experimental comparison of the maintainability of object-oriented and structured design documents. *Empirical Software Engineering* 2(3): 291–312.

Cohen, J. 1988. *Statistical Power Analysis for the Behavioral Sciences*. Academic Press.

Drappa, A., and Ludewig, J. 1999. Quantitative modeling for the interactive simulation of software projects. *Journal of Systems and Software* 46: 113–122.

Forrester, J. W. 1971. *Principles of Systems*. Cambridge: Productivity Press.

Graham, A. K., Morecroft, J. D. W., Senge, P. M., and Sterman, J. D. 1992. Model-supported case studies for management education. *European Journal of Operational Research*. 59: 151–166.

Kellner, M. I., and Hansen, G. A. January 1989. Software process modeling: A case study. *Proc. 22nd Annual Hawaii Int'l Conf. System Sciences*.

Lane, D. C. 1995. On a resurgence of management simulation games. *Journal of the Operational Research Society*. 46: 604–625.

Likert, R. 1932. A technique for the measurement of attitude. *Archives of Psychology* 22(140).

Lin, C. Y. May 1993. Walking on battlefields: Tools for strategic software management. *American programmer*: 33–40.

Lin, C. Y., Abdel-Hamid, T., and Sherif, J. S. 1997. Software-engineering process simulation model (SEPS). *Journal of Systems and Software* 38: 263–277.

Lipsey, M. 1990. *Design Sensitivity*. Sage Publications.

Madachy, R., and Tarbet, D. June 1999. Case studies in software process modeling with system dynamics. *Proc. 2nd Software Process Simulation Modeling Workshop (ProSim'99)*. Silver Falls, Oregon.

Milling, P. March 1995. Managementsimulation im Prozeß des organizationalen Lernens [organizational Learning and its Support by Management Simulators]. *Zeitschrift für Betriebswirtschaft*, Ergänzungsheft (supplement) 3/95: Lernende Unternehmen. 93–112. (Also available at URL http://iswww.bwl.uni-mannheim.de).

Morecroft, J. D. W. 1988. System dynamics and microworlds for policymakers. *European Journal of Operational Research* 35: 301–320.

Pfahl, D. 2001. *An Integrated Approach to Simulation-Based Learning in Support of Strategic and Project Management in Software Organisations*. PhD Theses in Experimental Software Engineering, Vol. 8. Stuttgart: Fraunhofer IRB Press.

Pfahl, D., Klemm, M., and Ruhe, G. 2001. A CBT module with integrated simulation component for software project management education and training. *Journal of Software and Systems* 59: 283–298.

Pfahl, D., Koval, N., and Ruhe, G. April 2001. An experiment for evaluating the effectiveness of using a system dynamics simulation model in software project management education. *Proceedings of 7th International Software Metrics Symposium*. London, United Kingdom, 97–109.

Richardson, G. P., and Pugh, A. L. 1981. *Introduction to System Dynamics Modeling with DYNAMO*. Cambridge: Productivity Press.

Sheskin, D. J. 1997. *Handbook of Parametric and Nonparametric Statistical Procedures*. Boca Raton: CRC Press.

Smith, B. J., Nguyen, N., and Vidale, R. F. May 1993. Death of a software manager: How to avoid career suicide through dynamic software process modeling. *American programmer* 10–17.

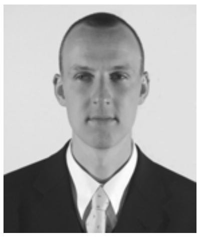
Vennix, J. A. M. 1990. Mental Models and Computer Models – design and evaluation of a computer-based learning environment for policy-making. PhD Thesis, University of Nijmegen.

Wildt, A. R., and Ahtola, O. T. 1978. *Analysis of Covariance*. Newbury Park: Sage Publications. Sage University Paper Series on Quantitative Applications in the Social Sciences, series no. 07–012.

Dietmar Pfahl received his MSc degree in applied mathematics and economics (Diplom-Wirtschafts-mathematik) from Ulm University, Germany, and a PhD in computer science from Kaiserslautern University, Germany. Between 1987 and 1996 he was a research staff member with the German Aerospace Research Establishment (DLR), and a software engineering consultant with Siemens Corporate Research. Since 1996, he has been with the Fraunhofer Institute for Experimental Software Engineering where he manages several national and international research and transfer projects with software industry. His current research interest includes quantitative software project management, software process analysis and improvement, and simulation-based learning. He is a member of the IEEE Computer Society, the ACM, The Institute of Mathematics and its Application (IMA), and the German Computer Society.

Oliver Laitenberger received his MSc and a PhD degree in computer science from Kaiserslautern University, Germany, in 1996 and 2000 respectively. After his affiliation with the Fraunhofer Institute for Experimental Software Engineering, he is currently working for Droege & Comp. GmbH, an international consultancy. His main interest includes project and quality management techniques, methods and tools in large-scale software development organizations. (Oliver_Laitenberger@droege.de; http://www.droege.de)

Günther Ruhe received a doctorate rer. nat degree in mathematics with emphasis on operations research from Freiberg University, Germany, and a doctorate habil. nat. degree from both the Technical University of Leipzig and University of Kaiserslautern, Germany. He had a visiting professorship at University of Bayreuth in 1991/92 and received an Alexander von Humboldt research fellowship at University of Kaiserslautern in 1992. Dr. Ruhe was visiting scientist at the IBM Research Center in Heidelberg in 1993. From 1996 until 2001 he was deputy director of the Fraunhofer Institute for Experimental Software Engineering. In July 2001 he joined the University of Calgary, Canada, where he holds the position of an Industrial Research Chair (iCORE; www.icore.ca), a joint position between the department of computer science and the department of electrical and computer engineering. Dr. Ruhe's laboratory for Software Engineering Decision Support (www.seng-decisionsupport.ucalgary.ca) currently includes 15 graduate students, post-doctoral students and research associates. In addition to his academic merits, Dr. Ruhe has comprehensive experience in collaborations with companies like Allianz life insurance, Bosch, DaimlerChrysler, Schlumberger and Siemens.



Joerg Dorsch received his MSc degree in computer science from Kaiserslautern University, Germany, in 2003, his primary interest being focused on software engineering and computer networks. During his study period, he worked several years as research assistant at Fraunhofer IESE on various projects. In May 2003, he joined Accenture where he now works as a consultant for IT Projects in Europe.