Fraunhofer Institut
Experimentelles
Software Engineering

An Experiment for Evaluating the Effectiveness of Using a System Dynamics Simulation Model in Software Project Management Education

Authors:
Dietmar Pfahl
Nataliya Koval
Günther Ruhe

IESE-Report No. 057.00/E
Version 1.0
August 15, 2000

A publication by Fraunhofer IESE

Fraunhofer IESE is an institute of the Fraunhofer Gesellschaft.
The institute transfers innovative software development techniques, methods and tools into industrial practice, assists companies in building software competencies customized to their needs, and helps them to establish a competetive market position.

Fraunhofer IESE is directed by
Prof. Dr. Dieter Rombach
Sauerwiesen 6
D-67661 Kaiserslautern

Abstract

Due to increasing demand for software project managers in industry, efforts are needed to develop the management-related knowledge and skills of the current and future software workforce. In particular, university education needs to provide to their computer science students not only technology-related skills but, in addition, a basic understanding of typical phenomena occurring in industrial (and academic) software projects.

This paper presents a controlled experiment that evaluates the effectiveness of using a process simulation model for university education in software project management. The experiment uses a pre-test-post-test control group design with random assignment of computer science students. The treatment of the experimental group involves a System Dynamics simulation model. The treatment of the control group involves a conventional predictive model for project planning, i.e. the well-known COCOMO model.

In addition to the presentation of the results of the empirical study, the paper discusses limitations and threats to validity. Proposals for modifications of the experimental design and the treatments are made for future replications.

Table of Contents

1 Introduction

Software development is a dynamic and complex process as there are many interacting factors throughout the lifecycle that impact cost and schedule of the development project, and quality of the developed software product. In addition, software industry constantly faces increasing demands for quality, productivity, and time-to-market, thus making the management of software development projects one of the most difficult and challenging tasks in any software organisation. Therefore, it is not surprising, that project management is one of the focus areas to which process simulation techniques have been applied in the domain of software engineering during the last decade, starting with the pioneering work of Kellner et al. [9][10] and Abdel-Hamid and Madnick [1].

Considering that the need for software is constantly growing world-wide, and hence the need for experienced and well-trained project managers, it is surprising that experience with using process simulation as a means for software project management education and training has rarely been published (examples are [6] and [15]).

The potential of simulation models for the training of managers has long been recognised: flight-simulator-type environments (or microworlds) confront managers with realistic situations that they may encounter in practice, and allow them to develop experience without the risks incurred in the real world [11]. Two detailed reports about training workshops for managers based on real industrial cases using simulation can be found in [8] (other examples are mentioned in [16] and [17], to give some pointers).

With regard to the specific topic of software project management, only few experimental studies have been conducted that involve the use of simulation models representing typical behaviour of software projects.

Experiments carried out at the Jet Propulsion Laboratory aimed at studying the decision-making process of software managers [13]. Twenty managers were asked to conduct a project simulated with the aid of the Software-Engineering Process Simulation Model (SEPS) [14]; some were provided with cause-effect feedback of their actions, while the others were not. It was observed that the second group (without feedback information) tended to act in a more "fire fighting" mode than the first one; and the feedback information was most beneficial to the less experienced managers.

At Draper Laboratory, a simulation model served as the basis for an experiment involving a group of 50 experienced software managers [21]. The scenario of the experiment involved a 15 percent requirement change in the course of the simulated project. Few of the managers were able to adapt properly to this situation: most of them reacted by hiring new staff late on the project (a typical "fire fighting" policy), and experienced budget and schedule overruns. Worse, the managers tended to reproduce exactly the same errors when running the scenario for the fourth or fifth time. The authors of the study conclude that the participating managers were limited by their mental model of the process, and that they were reluctant to change it.

The two experiments mentioned above show that natural one-way causal thinking can be detrimental to the success of software managers. Therefore, the aim of the training should go beyond that of facing people to realistic problems; the concern is also to make managers adopt systems thinking, and perceive the existence of (unexpected) feedback to management decisions.

As a first preparatory step towards supporting the formation of well-trained software project managers in industry, university education needs to provide to their computer science and software engineering students not only technology-related skills but, in addition, a basic understanding of typical phenomena occurring in industrial (and academic) software projects. Usually, there exist practical limitations that prohibit the exposure of students to realistic, large-scale industrial software development projects during education. This could be partially compensated by making students use software process simulation models that reproduce the behaviour of realistic (i.e. complex) software development projects.

This paper presents a controlled experiment that was conducted to investigate the effectiveness of computer-based training in the field of software project management using a System Dynamics (SD) simulation model. This study is viewed as exploratory, i.e. the intention is to identify and refine important hypotheses and to investigate them further. Section 2 presents the experimental details of the study. Section 3 summarises the results of the data analysis and presents important details that help to explain them. 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 to 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 has been 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 arrangement and presentation of the course material is defined by the single-learner training scenario. 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 System Dynamics (SD) simulation modelling method [7] [19]. This model simulates typical behaviour 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 pass 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 [2] 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 organisations.

The following dimensions were used to characterise "effectiveness" of the training session:

1. Interest in software project management issues.
2. Knowledge about typical behaviour patterns of software development projects.
3. Understanding of "simple" project dynamics.
4. Understanding of "complex" project dynamics.

In the study, these dimensions were represented by dependent variables (Dep.1 to Dep.4). How these variables were measured is described in more detail in Section 2.5.

2.1 Hypotheses

Standard significance testing was used to analyse the effectiveness of the training session. Two null hypotheses together with their associated alternative hypotheses were stated.

The first null hypothesis was stated as:

$H_{0,1}$: There is no difference between scores before (pre-test) and after (post-test) the training session.

The second null hypothesis was stated as:

$H_{0,2}$: There is no difference in effectiveness between the experimental group (using the SD model) and the control group (using the COCOMO model).

The alternative hypotheses, i.e., what was expected to occur, were then stated as:

1. H1 (relates to H0,1) – "Post-test versus pre-test scores": The average performance of all subjects (experimental group and control group) during post-test is not worse than during pre-test.
2. H2 (relates to H0,2) – "Performance improvement": The average performance improvement of the experimental group is not worse than the average performance improvement of the control group.
3. H3 (relates to H0,2) – "Post-test performance": The average post-test scores of the experimental group are better than the average post-test scores of the control group.

H_1 and H_2 apply to all dependent variables (Dep.1 to Dep.4). H_3 only applies to dependent variables Dep.1, Dep.2, and Dep.4, since it was not expected that using a SD model would significantly increase the understanding of "simple" project dynamics (Dep.3) if compared to using conventional project planning models.

2.2 Subjects

The participants of the study were computer science students at the University of Kaiserslautern, Germany, who were enrolled in the advanced software engineering class lasting one semester. During the lectures the students were

taught among other things fundamentals of an engineering-style planning and execution of software development projects.

While the course was running, subjects were asked if they would be interested in participating in an experiment related to software project management issues that would involve a simulation model. The subjects knew that they would have to pass a self-learning training session, that they would have to pass a test, and that the test scores would be analysed to evaluate the training session. Twelve students expressed their interest in participation.

As the German system allows students to take different classes at different times during their studies, information on their personal background with regard to experience in software development and software project management was captured before passing the pre-test.

2.3 Treatments

The training sessions of both groups, experimental and control, was structured by training scenarios, consisting of a sequence of scenario blocks. The generic 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 passed 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 submodels (views):

1. Production View: This view represents a typical software development life-cycle consisting of the following chain of transitions: set of requirements (planned functionality) → design documents → code → tested code. In order to limit model complexity, detection of defects during testing only causes reworking of the code and not of previous design documents. Similarly, optional quality assurance (QA) activities conducted during design and coding will only trigger reworking of documents that are developed during the respective phase.
2. Quality View: This view models the defect co-flow: defect injection (into design or code) → defect propagation (from design to code) → defect detection (in the code during testing) → 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 central 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). The average needed manpower is derived from the effort and duration estimates.
5. Productivity, Quality & 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 [18].

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 organisations 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 1) was distilled from the top 10 list of software metric relationships published by Barry Boehm in 1987 [3].

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 percent 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 3 times as much per SLOC as individual software programs. Software-system products (i.e. system of systems) cost 9 times as much."
6	"Walkthroughs catch 60% of the errors."

Table 1: List of principles dominating project performance

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 familiarise 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 (associated with Principle 6 in Table 1).

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 1. At the end of the role-play, a short discussion of the different possible solutions is provided, explaining the advantages and disadvantages of each.

2.3.2 Treatment of the control group

The control group passed 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 [2].

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 (cf. Table 2).

Random selection of subjects (R)	
Observation O_1 (pre-test)	
Treatment T_A of experimental group (A)	Treatment T_B of control group (B)
Observation O_2 (post-test)	

Table 2: Pre-test-post-test control group design

This design involves random assignment (R) of subjects to an experimental group and a control group. Both groups have to pass a pre-test (O_1) and a post-test (O_2). The pre-test and post-test concern the subjects' performance on the dependent variable(s) of the experiment. 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 3 lists all experimental variables, including one independent variable, four dependent variables, and three variables that represent potentially disturbing factors.

Independent Variable	
Ind.1	Type of treatment
Dependent Variables	
Dep.1	Interest in software project management issues ("Interest")
Dep.2	Knowledge about typical behaviour patterns of software development projects ("Knowledge")
Dep.3	Understanding of "simple" project dynamics ("Understand simple")
Dep.4	Understanding of "complex" project dynamics ("Understand complex")
Disturbing Factors	
DiF.1	Personal background
DiF.2	Time consumption / time need
DiF.3	Session evaluation (personal perception)

Table 3: Experimental variables

2.5.1 Independent variables

The independent variable Ind.1 (type of treatment) can have two values, either T_A, which is applied to the experimental group, or T_B, which is applied to the control group. 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 4 briefly summarises 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.

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 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. The application of scenario block 2 also has implications on the medium through which the course material is presented. Treatment T_B may choose between paper-based or web-based, whereas treatment T_A at least requires web-based presentation of scenario block PM Role Play. It should also be noted that for treatment T_B the presentation style of the training materials can always be similar to a textbook presentation even if the materials are provided web-based. 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 behavioural aspects of software projects.

	Treatment T_A	Treatment T_B
Scenario	Block 1 – 3' Block 2 – 15' Block 3 – 15' Block 4 – 12'	Block 1 – 3' n/a Block 3 – 30' Block 4 – 12'
PM model	SD model: - behavioural (white box) - point estimates (black box)	COCOMO model: - point estimates (black box)

Table 4: Differences between treatments

2.5.2 Dependent variables

The dependent variables Dep.1, Dep.2, Dep.3, and Dep.4 are determined by analysing 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 questionnaire consists of 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:

- Dep.1 ("Interest"): Questions about personal interest in learning more about software project management.
- Dep.2 ("Knowledge"): Questions about typical performance patterns of software projects. These questions are based on some of the empirical findings and lessons learned summarised in Barry Boehm's top 10 list of software metric relations [3].
- Dep.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.
- Dep.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 DiF.1, DiF.2, and DiF.3 are also derived from questionnaires that all subjects have to fill in. The questionnaire for DiF.1 will be filled in before the pre-test, the questionnaires for DiF.2 and DiF.3 will be filled in after the post-test.

The contents of the questionnaires are as follows:

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

2.6 Experimental procedure

The experiment was conducted following the plan presented in Table 5.

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

Table 5: Schedule of experiment

After a short introduction during which the purpose of the experiment and general organisational issues were explained, data on the background characteristics (variable DiF.1) was collected with the help of a questionnaire. Then the pre-test was conducted and data on all dependent variables (Dep.1 through Dep.4) was collected, again using questionnaires. Following the pre-test, a brief introduction into organisational 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 passed 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 DiF.2 and DiF.3. The time frames reserved for passing a certain step of the schedule was identical for the experimental and control groups.

The experiment was performed on two days following the schedule presented in Table 5. On the first day, the steps “Introduction to experiment”, “Background characteristics”, and “Pre-test” were conducted, consuming a total of 40 minutes. On the second day, the steps “Introduction to treatments”, “Random assignment of students to groups”, “Treatment”, “Post-test”, and “Time need & subjective session evaluation” were conducted, consuming a total of

90 minutes. Of the 12 students that agreed to participate in the experiment, 9 students participated in both pre-test and post-test. 5 subjects were assigned randomly to the experimental group (A), and 4 students to the control group (B).

2.7 Data collection procedure

The raw data for dependent variables Dep.1 to Dep.4 were collected during pre-test and post-test with the help of questionnaires.

The values for variable Dep.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 [12]. Each answer in the questionnaire is mapped to the value range R = [0, 1] assuming equidistant distances between possible answers, 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". All questions were formulated in a way that positive attitude towards project management education and application of project management techniques in projects must be expressed by ticking the fields "agree" or "fully agree".

The values for variables Dep.2 ("Knowledge"), Dep.3 ("Understand simple"), and Dep.4 ("Understand complex") are average scores derived from five (for Dep.2), seven (for Dep.3), and six (for Dep.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".

The raw data for disturbing factors DiF.1 to DiF.3 were collected before pre-test (DiF.1) and after post-test (DiF.2 and DiF.3).

In order to determine the values of factor DiF.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 DiF.1 eventually used for the statistical analysis were exclusively based on the normalised average scores derived from six 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 normalised value range, i.e. range R = [0, 1].

The values for factor DiF.2 are normalised average scores reflecting the "time need" for reading and understanding of the scenario blocks 1, 3, and 4, for familiarisation with the supporting tools, and for filling in the post-test questionnaire. For group A, the variable DiF.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 normalised 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 DiF.3 ("Session evaluation") are based on subjective measures reflecting the quality of the treatment. Again, for group A, the variable DiF.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 Dep.1, possible answers were encoded as "0", "0.25", "0.5", "0.75", and "1" depending on whether the subjective judgement 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 summarised in Figure 1. It is inspired by the work of Jac Vennix who conducted a similar experiment [22], assuming 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.

In a first step, the statistical analysis applies a t-test to investigate the effect of the independent variable on the dependent variables. For testing hypothesis H_1, a one-way paired t-test can be 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 [20]. For testing hypotheses H_2 and H_3, repeated measures analysis could not be applied, thus the appropriate test was a one-sided t-test for independent samples, or, equivalently, a single factor, one-way ANOVA [20].

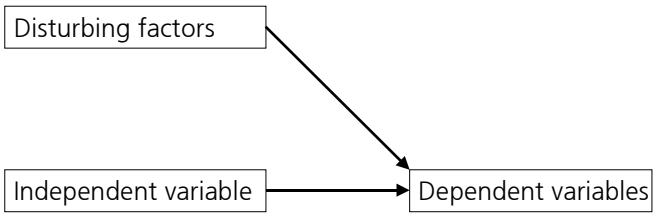

Figure 1: Relation between experimental variables

In addition to that, for testing hypotheses H_2 and H_3, analysis of covariance (ANCOVA) could be applied to improve the precision of the statistical analysis by removing potential bias due to disturbing factors [23].

To conduct the analysis, a level of significance, i.e., the α-level, has to be specified. Several factors have to be considered when setting α. First, the implications of committing a Type I error, i.e., incorrectly rejecting the true null hypothesis, have to be determined. In the context of this study, it would mean the cost of building and using simulation models in student education without achieving any beneficial effect as compared to using conventional planning models. Second, the goals of the study have to be taken into account. This can be discussed from two perspectives [4].

– From a *practical perspective*, which is asking for a judgement on whether it is likely that using SD simulation models for training has a better learning effect than using traditional planning models.
– From a *scientific perspective*, which is trying to identify cause-effect relationships between the type of the used planning model (with associated training scenarios) and the learning effect, with a high level of confidence.

As previously stated, the empirical work presented in this paper should be considered as exploratory research whose goal is twofold: First, potentially interesting and practically significant trends shall be identified in order to focus future studies. Second, initial insights into what might be the consequences of using SD simulation models for student education shall be gained. Therefore, not a too stringent α level should be adopted, since this might result in overlooking potential areas of further investigation. In the study presented, $\alpha = 0.1$ was used. This can be seen as a compromise between a more practical perspective, and a strictly scientific perspective.

Another factor affecting the analysis procedure is the small sample sizes, which are likely to have an adverse effect on the power of the applied statistical methods, i.e. the chance that if an effect exists it will be found. 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. Given that effect size and number of subjects are constant, increasing α is the only option for increasing the power of the test applied [20]. This provides further

justification for the decision to set α to 0.1 instead of the 0.05 level. Low power will have to be considered when interpreting non-significant results. In these cases practical significance needs to be considered. Practical significance occurs when the effect being investigated impacts upon the dependent variables in a manner that can be considered practically meaningful. To determine if this is the case, the observed effect size (γ) detected for each dependent variable and for each hypothesis has to be calculated. 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 [20]. For this exploratory study, effects where $\gamma \geq 0.5$ are considered to be of practical significance. This decision was made on the basis of the effect size indices proposed by Cohen [5].

3 Experimental results

Data was collected for nine subjects during pre-test and post-test. Therefore, 18 data points were available for each dependent variable and each disturbing factor – ten data points provided by the experimental group (A), and eight data points provided by the control group (B).

Tables 6 and 7 show the raw data collected during pre-test and post-test together with the calculated values for mean, median, and standard deviation.

Dependent variables: pre-test scores				
Group A	**Dep.1**	**Dep.2**	**Dep.3**	**Dep.4**
Student #1	0.75	1	0.29	0.33
Student #2	0.9	0.4	0	0
Student #3	0.5	0.6	0.71	0.5
Student #4	0.8	0.2	0.29	0.67
Student #5	0.5	0.6	0.29	0.33
$Mean_{pre\text{-}test}$	**0.69**	**0.56**	**0.31**	**0.37**
$Median_{pre\text{-}test}$	**0.75**	**0.6**	**0.29**	**0.33**
$Stdev_{pre\text{-}test}$	**0.18**	**0.30**	**0.26**	**0.25**
Group B	**Dep.1**	**Dep.2**	**Dep.3**	**Dep.4**
Student #6	1	0.2	0.14	0.67
Student #7	0.8	0.8	0.29	0.17
Student #8	0.75	0.6	0.43	0.33
Student #9	0.7	0.4	0.86	0.17
$Mean_{pre\text{-}test}$	**0.81**	**0.5**	**0.43**	**0.33**
$Median_{pre\text{-}test}$	**0.78**	**0.5**	**0.36**	**0.25**
$Stdev_{pre\text{-}test}$	**0.13**	**0.26**	**0.31**	**0.24**

Table 6: Pre-test scores

Dependent variables: post-test scores				
Group A	**Dep.1**	**Dep.2**	**Dep.3**	**Dep.4**
Student #1	0.85	0.8	0.43	0.33
Student #2	1	1	0.71	0
Student #3	0.5	1	0.71	0.33
Student #4	0.85	0.8	0.71	0.83
Student #5	0.75	0.6	0.71	0.67
$Mean_{post\text{-}test}$	**0.79**	**0.84**	**0.66**	**0.43**
$Median_{post\text{-}test}$	**0.85**	**0.80**	**0.71**	**0.33**
$Stdev_{pre\text{-}test}$	**0.19**	**0.17**	**0.13**	**0.32**
Group B	**Dep.1**	**Dep.2**	**Dep.3**	**Dep.4**
Student #6	1	0.6	0.86	0.83
Student #7	0.85	0.6	0.86	0.33
Student #8	0.75	0.4	0.86	0
Student #9	0.55	0.8	0.71	0.67
$Mean_{post\text{-}test}$	**0.79**	**0.6**	**0.82**	**0.46**
$Median_{post\text{-}test}$	**0.80**	**0.60**	**0.86**	**0.50**
$Stdev_{post\text{-}test}$	**0.19**	**0.16**	**0.07**	**0.37**

Table 7: Post-test scores

Table 8 shows the differences between post-test and pre-test scores together with the calculated values for mean, median, and standard deviation.

Dependent variables: difference scores				
Group A	**Dep.1**	**Dep.2**	**Dep.3**	**Dep.4**
Student #1	0.10	-0.20	0.14	0
Student #2	0.10	0.60	0.71	0
Student #3	0	0.40	0	-0.17
Student #4	0.05	0.60	0.43	0.17
Student #5	0.25	0	0.43	0.33
$Mean_{difference}$	**0.10**	**0.28**	**0.34**	**0.07**
$Median_{difference}$	**0.10**	**0.40**	**0.43**	**0.00**
$Stdev_{difference}$	**0.09**	**0.36**	**0.28**	**0.19**
Group B	**Dep.1**	**Dep.2**	**Dep.3**	**Dep.4**
Student #6	0	0.40	0.71	0.17
Student #7	0.05	-0.20	0.57	0.17
Student #8	0	-0.20	0.43	-0.33
Student #9	-0.15	0.40	-0.14	0.50
$Mean_{difference}$	**-0.03**	**0.10**	**0.39**	**0.13**
$Median_{difference}$	**0.00**	**0.10**	**0.50**	**0.17**
$Stdev_{difference}$	**0.09**	**0.35**	**0.38**	**0.34**

Table 8: Difference scores

Table 9 shows the data collected for the disturbing factors together with the calculated values for mean, median, and standard deviation. As can be seen, students in the control group (B) on average had more experience with software development (DiF.1) than students in the experimental group (A). In addition, students in the control group expressed less need of additional time (DiF.2) for conducting the treatment and passing the tests than students in the experimental group. Finally, students in the control group on average perceived their treatment easier, clearer, more absorbing, and more useful (DiF.3) than the students in the experimental group.

Disturbing factors					
Group A	**DiF.1**	**DiF.2**	**DiF.2'**	**DiF.3**	**DiF.3'**
Student #1	0.4	0.8	*0.67*	0.44	*0.52*
Student #2	0.6	0.4	*0.33*	0.56	*0.46*
Student #3	0.4	0	*0*	0.38	*0.54*
Student #4	0.8	0	*0*	0.38	*0.48*
Student #5	0.2	1	*1*	0.31	*0.5*
$Mean_{df}$	**0.48**	**0.44**	***0.4***	**0.41**	***0.5***
$Median_{df}$	**0.4**	**0.4**	***0.33***	**0.38**	***0.5***
$Stdev_{df}$	**0.23**	**0.46**	***0.43***	**0.09**	***0.03***
Group B	**DiF.1**	**DiF.2**		**DiF.3**	
Student #6	0.8	0.2		0.69	
Student #7	0.4	0.2		0.56	
Student #8	0.4	0.6		0.69	
Student #9	0.8	0.4		0.69	
$Mean_{df}$	**0.6**	**0.35**		**0.66**	
$Median_{df}$	**0.6**	**0.3**		**0.69**	
$Stdev_{df}$	**0.23**	**0.19**		**0.06**	

Table 9: Disturbing factors

3.1 Anomalies in the data set

Even though, assignment of students to groups A and B was done purely random, the average values for pre-test scores and disturbing factors were not evenly distributed. For example, the subjects in the control group (B) were more experienced with regard to actively developing software than the subjects in the experimental group (A). As will be discussed in sections 3.3 and 3.4 it was not possible in all cases to neutralise this potential bias by applying ANCOVA.

Another anomaly in the data set is the fact that five out of nine subjects performed worse during post-test than during pre-test with regard to at least one variable. This is particularly interesting for variables Dep.2, Dep.3, and Dep.4, because they relate questions for which the correct answers were provided through the training materials.

Particularly for variable Dep.4 ("Understanding of complex dynamics") the large variance in the post-test scores of both groups is surprising. No real clue has yet been found to this phenomenon, neither from analysing the comments made by subjects in the debriefing questionnaire, nor from talking to some of the subjects after the experiment.

3.2 Hypothesis H_1

Tables 10 and 11 show for each group (A and B) separately the results of testing the directional alternative hypothesis H_1: $Mean_{post\text{-}test} > Mean_{pre\text{-}test}$ 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, column five the critical value for $\alpha = 0.10$ which the t-value has to exceed to be significant, and column six provides the associated p value.

Group A					
Variable	γ	df	t-value	Crit. $t_{0.90}$	p-value
Dep.1	1.07	4	2.39	1.53	0.04
Dep.2	0.77	4	1.72	1.53	0.08
Dep.3	1.23	4	2.75	1.53	0.03
Dep.4	0.35	4	0.78	1.53	0.24

Table 10: Group A results for "post-test" vs. "pre-test"

By examining columns four and five of Table 10 one can see that significant results were achieved for dependent variables Dep.1, Dep.2, and Dep.3. Therefore, for group A, H_1 can be accepted for variables Dep.1 to Dep.3 but not for variable Dep.4. It is worth noting though that the value for dependent variable

Dep.4 supports the direction of the hypothesis, however without showing an effect size of practical significance.

For replication purpose, using the same experimental design, a minimum number of 56 subjects is required to have a reasonable chance of achieving statistical significance with regard to variable Dep.4, i.e. one where the power of the test is approximately 0.8 (for $\alpha = 0.1$, and the calculated effect size γ). Recalling the fact that several subjects in group A obviously did not have enough time to read and understand those training materials in scenario block 3 that are particularly related to "complex project dynamics", improvement of the experimental procedure is required before repeating the experiment.

Group B					
Variable	γ	df	t-value	Crit. $t_{0,90}$	p-value
Dep.1	**	3	-0.58	1.64	**
Dep.2	0.29	3	0.58	1.64	0.30
Dep.3	1.05	3	2.09	1.64	0.06
Dep.4	0.37	3	0.73	1.64	0.26

Table 11: Group B results for "post-test" vs. "pre-test"

Table 11 shows that for group B (control group) H_1 can only be accepted for variable Dep.3 ("Understanding of simple project dynamics"). The post-test scores for Dep.3 are significantly larger than the pre-test scores. In contrast, the value for variable Dep.1 not even supports the direction of the alternative hypothesis. For replication purpose, given the medium effect size of variables Dep.2 and Dep.4, a minimum number of approximately 100, respectively 56 subjects will be required to provide the test with a power of 0.8 (with $\alpha = 0.1$).

3.3 Hypothesis H_2

Table 12 shows the results of testing the directional alternative hypothesis H2: Meandifference(A) > Meandifference(B) using a one-tailed t-test for independent samples.

Group A versus B					
Variable	γ	df	t-value	Crit. $t_{0,90}$	p-value
Dep.1	1.38	7	2.06	1.42	0.04
Dep.2	0.51	7	0.75	1.42	0.24
Dep.3	0.16	7	-0.23	1.42	**
Dep.4	0.23	7	-0.33	1.42	**

Table 12: Results for "performance improvement"

As can be seen, for significance level □ = 0.1, the performance improvement, i.e. the score difference between post-test and pre-test, of variable Dep.1 ("Interest") is significantly larger for the experimental group (A) as compared to the control group (B), and thus alternative hypothesis H2 can be accepted. It

can also be noted that the value of variable Dep.2 ("Knowledge of empirical patterns") supports the direction of the hypothesis showing a medium to large effect size. Again, for future replication of the experiment with a power of approximately 0.8, at least 36 subjects per group are needed for the given effect size and □ = 0.1.

Note that the values of variables Dep.3 and Dep.4 not even support the direction of the alternative hypothesis.

Group A versus B							
Var.	df	F	Crit. $F_{0,90}$	p-value	Covariate	Corr. means A ------ B	
Dep.1	7	4.23	3.59	0.08	-	*0.10*	*-0.03*
Dep.1	6	3.16	3.78	0.13	DiF.1	0.09	-0.01
Dep.2	7	0.57	3.59	0.48	-	*0.28*	*0.10*
Dep.2	6	5.86	3.78	0.05	DiF.1	0.35	0.01
Dep.2	6	1.44	3.78	0.28	DiF.2	0.31	0.07

Table 13: ANCOVA results for "performance improvement"

In order to check whether the test results are robust with respect to disturbing factors, an analysis of covariance (ANCOVA) was conducted. Table 13 shows only results for which the implicit assumptions underlying ANCOVA were fulfilled, i.e. homogeneity of regression coefficients, and difference from zero of the coefficient of the disturbing variable. The assumptions were fulfilled only for three cases. In the first case, neutralising the impact of disturbing variable DiF.1 ("Personal background") on variable Dep.1 slightly decreased the strength of the treatment effect (with reduced effect size $\gamma = 0.67$). In the second case, however, neutralisation of the impact of DiF.1 on variable Dep.2 strongly increased the effect of the treatment, i.e. the null hypotheses can be rejected at a significant level (with $p = 0.05$) and thus the alternative hypotheses can be accepted (with $\gamma = 0.91$). When neutralising the impact of the disturbing factor DiF.2 ("Time need") on variable Dep.2, again an improvement of the p-level can be observed, however to a smaller extent (with $\gamma = 0.44$).

3.4 Hypothesis H_3

Table 14 shows the results of testing the directional alternative hypothesis H_3: $Mean_{post\text{-}test(A)} > Mean_{post\text{-}test(B)}$ using a one-tailed t-test for independent samples.

Group A versus B					
Variable	γ	df	t-value	Crit. $t_{0,90}$	p-value
Dep.1	0.01	7	0.02	1.42	0.49
Dep.2	1.45	7	2.16	1.42	0.03
Dep.4	0.07	7	-0.11	1.42	**

Table 14: Results for "post-test performance"

For significance level $\alpha = 0.1$, the post-test scores of variable Dep.2 ("Knowledge of empirical patterns") is significantly larger for the experimental group (A) as compared to the control group (B), and thus alternative hypothesis H_3 can be accepted. It can also be noted that the value of variable Dep.1 ("Interest") supports the direction of the hypothesis, however, only with a very small effect size. In this case, in a future replication of the experiment, for the given effect size and significance level $\alpha = 0.1$, a power of approximately 0.8 could only be reached with more than 1000 subjects per group.

The values for variable Dep.4 ("Understanding of complex project behaviour") not even support the direction of the hypothesis.

Group A versus B							
Var.	df	F	Crit. $F_{0,90}$	p-value	Covariate	Corr. means A ------ B	
Dep.1	7	0.00	3.59	0.98	-	*0.79*	*0.79*
Dep.1	6	3.11	3.78	0.13	Dep.1 (pre)	0.84	0.72
Dep.4	7	0.01	3.59	0.92	-	*0.43*	*0.46*
Dep.4	6	0.08	3.78	0.78	Dep.4 (pre)	0.42	0.48

Table 15: ANCOVA results for "post-test performance"

As with hypothesis H_2, in order to check whether the test results are robust with respect to disturbing factors, an analysis of covariance (ANCOVA) was conducted. Again, Table 15 shows only results for which the implicit assumptions underlying ANCOVA were fulfilled. This happened only in two cases. In the first case, neutralising the impact of pre-test scores on variable Dep.1, considerably increased the strength of the treatment effect ($p = 0.13$), with an effect size of $\gamma = 0.79$ now reaching the level of practical significance. In the second case, however, neutralisation of the impact of pre-test scores on variable Dep.4 only marginally increased the effect of the treatment ($\gamma = 0.10$). Note that attempts to neutralise potential bias induced by disturbing factors DiF.1 to DiF.3 was unfeasible in all cases, since there was no significant difference from zero of the respective coefficient in the ANCOVA model.

3.5 Analysis summary

The results of the statistical analysis can be grouped according to the degree of evidence in supporting the hypotheses. Three categories have been defined: strong support, i.e. the data shows statistical significance (at α level 0.10), weak support, i.e. the data shows practical significance ($\gamma \geq 0.5$), and no support, i.e. the data has neither statistical nor practical significance.

3.5.1 Strong support

Statistical and practical significance was obtained for variables Dep.1 (only group A), Dep.2 (only group A), and Dep.3 (groups A and B) in support of H_1 – "post-test scores versus pre-test scores". After elimination of disturbing factors, statistical and practical significance in support of hypotheses H_2 – "performance improvement" – and H_3 – "post-test performance" – was only obtained for variable Dep.2 ("Knowledge of empirical patterns").

3.5.2 Weak support

Practical significance was obtained for variable Dep.1 ("Interest") in support of hypotheses H_2 – "performance improvement" – and H_3 – "post-test performance". For hypothesis H_2, impact on variable Dep.1 is even statistically significant, if only the results of the one-tailed t-test (cf. Table 12) are considered.

3.5.3 No support

No practical significance was found for variable Dep.4 ("Understand complex project dynamics") in all three hypotheses H_1, H_2, and H_3, and for variable Dep.3 ("Understand simple project dynamics") in hypotheses H_2 and H_3.

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 modelling [11].
2. Interest in a topic and evaluation of a training session are difficult concepts that have to be captured with subjective measurement instruments. In order to keep this threat to validity as small as possible, in the study, the instruments for measuring variables Dep.1 and DiF.3 were derived from measurement instruments that had been successfully applied in a similar study [22].
3. There are indications that the distinction between "simple dynamics" and "complex dynamics", as it was made for measuring variables Dep.3 and Dep.4 (cf. Section 2.5.2.), was too simplistic.
4. It is difficult to avoid "unfair" comparison between SD models and COCOMO, because there are obviously features coming with SD models that per definition cannot be offered by COCOMO (e.g. simulation of parameter changes over time / on-the-fly modification of model assumptions, etc.).

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 tried to be 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 DiF.1. Any potential bias induced by differences

in pre-test scores and experience were tried to be neutralised 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 need indicated that most subjects of group A did not have enough time to execute both scenario blocks 2 and 3. In addition, – even though this was tried to be avoided 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 on the first day of the experiment, i.e. when passing the pre-test, that at the end of the experiment they will pass a post-test with exactly the same questions. Since this information was not given to the subjects, all materials were re-collected after the pre-test, and the treatment with post-test took place 48 hours 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 generalised 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 computer science students 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 generalisation 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 modelling and COCOMO.

In any case, the point should be emphasised 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

This study investigated the effect of using a System Dynamics (SD) simulation model in software project management education of computer science students. The treatment focused on problems of project planning and control. The performance of the students was analysed with regard to four dimensions, i.e., interest in the topic of project management (Dep.1), knowledge of typical project behaviour patterns (Dep.2), understanding of simple project dynamics (Dep.3), and understanding of complex project dynamics (dep.4). This was done by comparing the test results of students who passed a training session using the SD model to the test results of students who passed a training session using the COCOMO model. Even though the statistical results must be interpreted with caution, due to small number of subjects involved and several threats to construct and internal validity, the findings of the analyses showed several interesting trends.

First, the treatment involving the SD model had a positive impact on the change of scores from pre-test to post-test for all four dependent variables. The effect was statistically significant for Dep.1 to Dep.3. For Dep.4 the power of the test seemed to be too low to be able to detect the effect at the set significance level $\alpha = 0.1$.

Second, the treatment involving the SD model achieved practical significance on performance improvement and post-test performance for variable Dep.1, and even statistical significance for variable Dep.2.

Though positive, the second result might be related to problems with internal validity of the experiment, i.e. 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 Dep.3 and – particularly – Dep.4.

In order to avoid speculations about the positive or negative effects of the threats to internal validity, the experimental design, and the treatments involved in the experiment need improvement.

With regard to enhancing the treatments, two issues are of relevance. First, more time has to be allowed particularly for executing scenario blocks 2 (PM Role Play) and 3 (PM Planning Models), and for the familiarisation with the simulation tool. Second, the experimental treatment, as it is now, does not yet fully exploit all potentially available features of a learning tool that SD model

usage and model building can 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 (T_A) had two consequences. As mentioned before, the role-play provided explicit information on observed empirical patterns in software development projects to the subjects in group A, which were not given in such an explicit form to subjects in group B. This might explain why subjects in group A had clearly better scores for variable Dep.2 than subjects in group B. On the other hand, because they did not have to pass a role-play, subjects in group B had more time than subjects in group A (30' instead of 15') to read and understand the information provided in scenario block 3 (PM Planning Models). This might explain why subjects in group A did not have better scores for variables Dep.3 and Dep.4. A 2x2 factorial design (either crossed or nested) could help to better distinguish between the effects of having or not having a role-play and the effects induced by the nature of the PM model.

A main direction of future research will pursue possibilities to extend the SD model and the associated single-learner scenario block "PM Role Play" towards collaborative learning/training.

6 References

[1] Abdel-Hamid TK, Madnick SE, *Software Project Dynamics – an Integrated Approach*, Prentice-Hall, 1991.

[2] Boehm B, *Software Engineering Economics*, Prentice-Hall, Englewood Cliffs, N.J, 1981.

[3] Boehm BW, "Industrial software metrics top 10 list", *IEEE Software*, September 1987, pp. 84-85.

[4] Briand LC, Bunse C, Daly JW, Differding C, "An Experimental Comparison of the Maintainability of Object-Oriented and Structured Design Documents", Empirical Software Engineering, 2(3), 1997, pp. 291-312

[5] Cohen J, *Statistical Power Analysis for the Behavioral Sciences*, Academic Press, second edition, 1988.

[6] Drappa A, Ludewig J, "Quantitative modeling for the interactive simulation of software projects", *Journal of Systems and Software*, 46, 1999, pp. 113-122.

[7] Forrester JW, *Principles of Systems*, Productivity Press, Cambridge, 1971.

[8] Graham AK, Morecroft JDW, Senge PM, Sterman JD, "Model-supported case studies for management education", *European Journal of Operational Research*, 59, 1992, pp. 151-166.

[9] Humphrey WS, Kellner MI, "Software Process Modeling: Principles of Entity Process Models", Proc. 11th Int'l Conf. Software Engineering (ICSE), IEEE Computer Soc., 1989, pp. 331-342.

[10] Kellner MI, Hansen GA, "Software Process Modeling: A Case Study", *Proc. 22nd Annual Hawaii Int'l Conf. System Sciences*, Jan. 1989.

[11] Lane DC, "On a Resurgence of Management Simulation Games", *Journal of the Operational Research Society*, 46, 1995, pp. 604-625.

[12] Likert R, "A technique for the measurment of attitude", Archives of Psychology, 22(140), 1932.

[13] Lin CY, "Walking on Battlefields: Tools for Strategic Software Management", *American Programmer*, May 1993, pp. 33-40.

[14] Lin CY, Abdel-Hamid T, Sherif JS, "Software-Engineering Process Simulation Model (SEPS)", Journal of Systems and Software, 38, 1997, pp. 263-277.

[15] Madachy R, Tarbet D, "Case Studies in Software Process Modeling with System Dynamics", *Proc. 2nd Software Process Simulation Modeling Workshop (ProSim'99)*, Silver Falls, Oregon, June 1999.

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

[17] Morecroft JDW, "System dynamics and microworlds for policymakers", *European Journal of Operational Research*, 35, 1988, pp. 301-320.

[18] Pfahl D, Klemm M, Ruhe G, "Using System Dynamics Simulation Models for Software Project anagement Education and Training", Proc. 3rd Process Simulation Modeling Workshop (ProSim'2000), London, United Kingdom, July 12-14, 2000. (Also available as Fraunhofer IESE Technical Report no. 035.00/E)

[19] Richardson GP, Pugh AL, *Introduction to System Dynamics Modeling with DYNAMO*, Productivity Press, Cambridge, 1981.

[20] Sheskin DJ, *Handbook of Parametric and Nonparametric Statistical Procedures*, CRC Press, Boca Raton, 1997.

[21] Smith BJ, Nguyen N, Vidale RF, "Death of a Software Manager: How to Avoid Career Suicide through Dynamic Software Process Modeling", *American Programmer*, May 1993, pp. 10-17.

[22] Vennix JAM, "Mental Models and Computer Models – design and evaluation of a computer-based learning environment for policy-making", *PhD Thesis*, University of Nijmegen, 1990.

[23] Wildt AR, Ahtola OT, *Analysis of Covariance*, Sage University Paper Series on Quantitative Applications in the Social Sciences, series no. 07-012, Sage Publications, Newbury Park, 1978.

List of Tables

List of Figures

Document Information

Title:	An Experiment for Evaluating the Effectiveness of Using a System Dynamics Simulation Model in Software Project Management Education
Date:	August 15, 2000
Report:	IESE-057.00/E
Status:	Final
Distribution:	Public