Evaluating the Use of System Dynamics Models in Software Project Management

MÁRCIO DE OLIVEIRA BARROS
CLÁUDIA MARIA LIMA WERNER
GUILHERME HORTA TRAVASSOS

COPPE / UFRJ – Computer Science Department
Caixa Postal: 68511 - CEP 21945-970 - Rio de Janeiro – RJ
Voice: 5521 2562-8675 / Fax: 5521 2562-8676
{marcio, werner, ght}@cos.ufrj.br

Abstract

This paper presents an empirical study aiming to evaluate the application of system dynamics models in software project management. In this study, a project concerning the specification and implementation of part of an academic control system for a graduate department was proposed for several participants. The project was decomposed into an activity network and several developers were available to accomplish the activities.

Each participant was asked to impersonate as the project manager. Participants should make decisions in order to finish the project in the smallest time as possible. Such decisions included assigning the available developers to project activities (according to developer's skills and activity's requirements), deciding how many hours developers should work per day, and how much time to invest in quality assurance activities. Since a real project could not be initiated for each participant, a software project emulator was developed, where participants analyzed the project status and enacted their decisions.

Half of the eighteen participants managed the project based only on their personal knowledge and experience, while the second half was allowed to use system dynamics models to support their decisions. The results from the experimental study analysis show that, for the selected participants, managers using system dynamics models to support their decision perform better than managers who base their decisions only upon personal experience. In this paper, we present detailed results from the experimental study and some directions to improve the application of system dynamics models in project management that where highlighted during the execution of the study.

KEYWORDS: empirical studies, simulation tools, project management

1 Introduction

The development of large software systems is a complex undertaking. High cost and schedule overruns are frequent in the software development industry (Standish Group, 1994). The recurring failures to produce large systems within planned schedule and budget have often been associated to management problems, such as bad communication, malformed teams, and unreliable risk analysis (Brown, 1996).

Project management is a knowledge intensive activity. Managers use their skills and experience to make decisions while a software development process is executed. It is well accepted that experienced managers usually perform better than novice managers due to the

experience they have accumulated by taking part in past projects and the knowledge acquired from this experience. As proposed by the "recognition-primed decision model" (Klein, 1998), managers tend to keep a collection of patterns in their minds and compare these patterns to the current context when making decisions.

Senior managers generally make this assessment mentally. They construct a mental model of the project, create mental models for the problems and opportunities under investigation, and scan their pattern repository for adequate actions to be taken against the problems or to explore the opportunities. However, mental models are limited by the human mind's ability to cope with multiple distinct factors. So, mental models are usually simple (Sterman, 1988), conveying only a few major components and the relationships among them.

In project management, where several related components, complex dynamics, multiple feedback loops, and delays between actions and their effects are present (Sterman, 1992), simple models may incur in erroneous interpretation of a system's behavior. So, there is a need for explicit representations for mental models. Our hypothesis is that system dynamics can help in the development of these models (Barros et al., 2000).

This paper presents the results of an experimental analysis that evaluated if system dynamics models can help software project managers to support the decisions they make during project development. The experimental study was accomplished within the context of a major development effort, which aims to define a management paradigm, known as *scenario based project management,* which is based on using system dynamics models to support management decisions before applying them to a project under development.

The paper is divided in five sections. The first one comprises this introduction. Section 2 briefly presents the scenario based project management paradigm and its major artifacts. Section 3 describes the experimental study that evaluates the application of system dynamics models on project management. Section 4 presents related works. Finally, we draw some concluding remarks and future work directions in Section 5.

2 Scenario Based Project Management

Scenario based project management is a project management approach that uses system dynamics models to describe the knowledge possessed by senior managers in a way that the effects provided by applying this knowledge upon a specific project can be quantitatively evaluated and presented to less experienced managers. The paradigm aims to share senior managers knowledge by providing models that document it. Also, it aims at quantifying the subjective information regarding software project management, allowing the precise evaluation of the expected effects of actions taken while managing a project.

The scenario based project management paradigm is centered on two artifacts: the project and scenario models. The project model defines the expected behavior for a software project, while scenario models represent situations that may arise during its development process.

The project model depicts the activities that must be performed during the project development, the team that must accomplish these activities, the resources to be consumed, and the software artifacts that will be created and transformed along the process.

Scenario models represent uncertain events, management theories, policies, actions, and strategies that can be applied or imposed to the project. Since some of these practices are valid for several projects, scenario models convey reusable project management knowledge. Scenario models are developed by experienced managers and stored in a scenario repository.

While planning a software project, a manager first builds a project model for the application to be developed. Next, the manager searches and retrieves the relevant scenarios in the development context, that is, scenarios that describe knowledge related to the required activities, artifacts to be built, and the personnel that forms the development team. Finally, the manager executes an iterative analysis, where these scenario models are integrated into the

project model. The integration process occurs within a simulation environment, in which the impact provided by each scenario upon the project model behavior is evaluated. As development unfolds, the project model is updated with status information and new scenario analysis can be carried out to test project sensibility to these scenarios.

Both project and scenario models are formal software project models. They are represented using system dynamics. However, system dynamics models are traditionally described through mathematical equations, which are hard to understand and adapt, inhibiting its use to describe large and complex software projects. The proposed project model is described in a high-level language that provides constructors such as activities, developers, resources, and artifacts. These constructors are detailed in a domain model[1], which allows their translation to system dynamics equations. Such language helps model development, without losing representation and simulation capabilities. Further information about domain models and their translation process to system dynamics equations can be found in (Barros et al., 2000; Barros et al., 2001).

Simulation allows a project manager to evaluate the behavior of a project model. The project's high-level description is translated to mathematical equations, which are calculated along several simulation runs. Such equations describe how project results (cost, schedule, quality, and so on) evolve over time. When a scenario is integrated to a project model, its mathematical formulations blend with the original project model equations, usually modifying the equations that affect project results. So, project models integrated with scenarios may present different behavior when simulated. By simulating the project model, annotating the simulation results, integrating the project model with a scenario model, simulating it again, and comparing both simulation results, a manager can observe the impact provided by the scenario upon a project result (sensitivity analysis). Detailed information about scenario models and their integration to project models is presented in (Barros et al., 2002).

3 Experimental Analysis

After developing the theory behind scenario based project management, the tools required to build and integrate scenario and project models, and defining some scenarios based on information taken from the software engineering literature, we planned an empirical feasibility analysis to evaluate the proposed approach. In this section, we summarize the results of this study.

In a feasibility study, data is collected according to some experimental design, but full control over all possible variables is not achieved (Shull et al., 2001). This kind of empirical study usually aims to provide researchers with enough information to justify continued improvement of the techniques under analysis. In the current study, we emphasized functionality and usefulness, setting usability as a secondary goal. Also, we focused on applying scenarios for project management, instead of building new scenarios, because we wanted to provide some evidence of scenario models efficacy before investing effort in better scenario and project model development tools.

The feasibility study reported in this section was accomplished during a graduate software engineering course in the Winter 2001 at the Systems Engineering and Computer Science Department of the Federal University of Rio de Janeiro (COPPE/UFRJ). The participants of the proposed experimental analysis were students of the referred course, students from a graduate software engineering program and students from an undergraduate computer science department. Of the eighteen (18) subjects, thirteen (13) were master degree

---

[1] A domain model describes the relevant elements that compose a modeling domain and the relationships among them. Each element is composed by properties and behavior. Properties are values that parameterize element behavior, which is described by system dynamics equations (Barros et al., 2001).

students, four (4) were doctoral students, and one (1) was an undergraduate student. Eight (8) subjects had been project leaders in industrial projects, while three (3) subjects had been leaders in academic projects, and seven (7) subjects had not been project leaders neither in academy nor in industry. From the later seven subjects, three (3) had participated in industrial projects, three (3) had developed software as part of coursework, and one (1) had developed software only for personnel uses. Table 1 summarizes subject's information.

Table 1 – Detailed information about the experiment participants

ID	Have used models?	Formation	Development Experience	Leadership Experience
1	Yes	MSc	Industry (3 years)	Industry (1 year)
2	Yes	DSc	Personal use	None
3	Yes	MSc	Academy (6 projects)	Academy
4	Yes	MSc	Academy (3 projects)	None
5	Yes	MSc	Industry (0,5 year)	Academy
6	Yes	MSc	Industry (3 years)	Academy
7	Yes	MSc	Industry (2 years)	None
8	Yes	DSc	Industry (2 years)	Industry (1 year)
9	Yes	MSc	Academy (2 projects)	None
10	No	MSc	Industry (3 years)	None
11	No	Undergraduate	Academy (1 project)	None
12	No	DSc	Industry (10 years)	Industry (4 years)
13	No	DSc	Industry (3 years)	Industry (2 years)
14	No	MSc	Industry (4 years)	Industry (2 years)
15	No	MSc	Industry (4 years)	Industry (2 years)
16	No	MSc	Industry (1 year)	None
17	No	MSc	Industry (4 years)	Industry (2 years)
18	No	MSc	Industry (10 years)	Industry (2 years)

The experimental analysis aimed to observe whether managers who used system dynamics models to support their decision would perform better than managers who relied only on their experience and other techniques. The selected performance criterion was time to conclude a project: managers were asked to conclude their project in the less time possible. We acknowledge that time to conclude a project may not be the most important aspect for a software project. However, it was selected as performance criteria because we needed a quantitative metric to compare the participant's performance (others could have been chosen, such as project cost, project quality, and so on). Our null hypothesis[2] states that the average time to conclude the proposed project of subjects using system dynamics models is equal to the average time to conclude the project of subjects not using these models. Our alternative hypothesis[3] states that the average time to conclude the project of subjects using system dynamics models is below the average time to conclude the project of subjects not using these models. Also, we intended to qualitatively evaluate the feasibility and usefulness of using system dynamics models in project management.

The study was planned as a single object, multi-test experiment (Wohlin et al., 2000). In a single object multi-test experiment, two or more groups analyze the same object, each applying a different approach. In the proposed experiment, the subjects were randomly assigned into two groups: one to use system dynamics models and one to serve as a control group, accomplishing the study without the aid provided by the models. Two (2) doctoral students and seven (7) master degree students composed the first group. The remaining subjects were assigned to the second group. Five (5) subjects from the first group had

---

[2] The null hypothesis is the statement that the experimental study aims to refute.

[3] The alternative hypothesis is the statement that negates the null hypothesis.

leadership experience in academic or industrial projects. Six (6) subjects from the second group had such experience. Table 2 summarizes participants' distribution among groups.

Table 2 – Participants' distribution among groups

	Subjects using models	Subjects not using models
DSc Student	2	2
MSc Student	7	6
Undergraduate	0	1

Each subject was asked to manage a software project, without the intervention of other subjects. The project to be managed aimed to develop part of the Systems Engineering and Computer Science Department's academic control system (CtrlPESC). This system manages information about professors, students, research areas, disciplines, and course registrations. It is a small system, comprising about 67 adjusted function points[4] (Pressman, 2000; IFPUG, 1999). Subjects from the first group were trained to use system dynamics models and a simulation environment, while subjects from the second group only received a project debriefing.

Since we intended to run a feasibility analysis, the cost of executing a real project with similar characteristics (team formation, team domain knowledge, activities to be performed, among others) for each subject was prohibitive. So, a project emulator was used. The emulator is a software system that controls and presents to its user a project's behavior, adjusting it according to decisions taken by the users. The emulator dictates project behavior according to a model and a random generation engine, which defines the duration for each project activity (durations are represented stochastically in the model). Due to the engine's stochastic features, each subject observed distinct activity durations.

Decision points where subjects could act included: determining the developer to assign for each project activity, deciding how much time to spend in quality control activities, and defining the number of hours each developer should work per day. The subjects from the first group could test the project sensibility for their decisions in the simulation environment before applying them to the project emulator. Time was not a constraint on decision-making for both groups: the subjects spent as much time as they desired in analysis before applying their decisions to the project emulator. After applying their decisions, the subjects could instruct the emulator to advance project time by several days, observing the effectiveness of their decisions upon the project behavior.

All subjects received the project emulator, conveying a process for the CtrlPESC project and the descriptions of a set of developers that could take part on the software project team. All subjects received a brief description of the project. Subjects using system dynamics models also received the simulation environment, a project model, and a set of scenarios models, built from knowledge presented in the technical literature, mostly from (Jones, 2000) and (Abdel-Hamid and Madnick, 1991).

Questionnaires were used to collect qualitative data that characterized each distinct subject and directly addressed the questions of feasibility and usefulness. The qualitative data included open-ended questions concerned with opinions about the models' effectiveness, measured by confidence on whether subjects could conclude the project in less time without their aid, and a subjective evaluation about the models' usefulness and the difficulties regarding the interpretation of their results. The project emulator gathered quantitative

---

[4] "Function points" is a metric that measures the size of a software project based on its inputs, outputs, internal file structure, and interfaces. For information systems developed in high-level languages, such as Pascal or C, a function point can be approximately compared to 100 lines of code.

information about each subject performance, that is, the time that each subject took to conclude the CtrlPESC project, measured in days.

After collecting quantitative performance data, we have conducted a two-phased outlier elimination procedure. First, a qualitative analysis eliminated one subject that used system dynamics models (46 days) because the subject declared that he committed errors during the study. Then, a T distribution-based cut was used to quantitatively eliminate extreme values (Freund and Perles, 1999). This type of extreme value elimination procedure is useful when we have small samples and cannot assume that the population is normally distributed. Table 3 presents the data points after extreme value eliminations. Eliminated values are crossed and italiced.

Table 3- Time taken by each subject to accomplish the example project (after extreme value elimination)

Subjects using models		
39 days	38 days	25 days
25 days	25 days	27 days
30 days	*46 days*	27 days

Subjects not using models		
18 days	28 days	37 days
42 days	31 days	33 days
35 days	*57 days*	*58 days*

Since we are interested in verifying differences among averages, we selected average evaluation statistical techniques to analyze the results from the experimental study. The data in Table 3, except for the outliers, was submitted to a 95% T-test[5] (Freund and Perles, 1999) that concluded that the average time to conclude the project for subjects who used system dynamics models was less than the average time taken by the ones who did not use such models. A Mann-Whitney (Wohlin et al., 2000) rank-based statistical analysis asserted the T-test results. Table 4 and Table 5 summarize the analysis results.

Table 4 – Analysis results for the experimental study

Project Conclusion Time	Subjects using models	Subjects not using models
Average	28,1 days	34,3 days
Maximum	38,0 days	42,0 days
Minimum	25,0 days	28,0 days
Standard deviation	4,7 days	4,9 days

Table 5 – Intermediate results from the T-test (Freund e Perles, 1999)

T-Test upon Project Conclusion Time (95%)				
Std Deviation	T_0	Freedom	T Distribution	Result ($T_0 < -T$)
4,79	-2,32	11	2,20	$\mu_{\text{w/out models}} > \mu_{\text{w/models}}$

The quantitative data from this study showed that subjects using the system dynamics models performed better than subjects not using them. Also, all subjects from the first group agreed that the models were helpful. Thus, the data and positive results drawn from it show some indications that the system dynamics based project and scenario model integration and simulation techniques are feasible and that they provide help for project managers.

The qualitative data indicates that the research upon the techniques is not concluded. The questionnaires returned by subjects from the first group indicate that three (3) subjects had difficulties interpreting the results produced by the simulator. So, the simulation environment's user interface should be improved. Also, though the experimental study yielded positive results, we found some points where it could be improved, including:

---

5 A T-test is a statistical analysis procedure based on the T distribution that compares if the averages of two groups are the same (to a degree of certainty, such as 90% or 95%) in the lights of their variances.

- Allowing subjects from the first group to use the simulator in a "toy" project during the training session. The current training session included only a brief presentation of the simulator. Four subjects from the first group required more training to effectively use the proposed techniques and the simulator environment;

- Providing training on how to use the project emulator for subjects from both groups. Since support from the study organizers was required by several subjects, a training section that shows subjects how to use the emulator would be beneficial;

- Presenting the dynamics that govern the models used in the experimental study during the training sessions. One subject from the first group was confused with some technical terms used in the study, while two subjects from the same group were concerned about quality assurance activities;

- Perfecting the mechanisms that present simulation results to the user. Three subjects from the first group proposed improvements for the simulation environment, mostly involving enhancements in simulation results visualization.

4 Related Works

Several models were developed to represent the dynamics of managing a software project (Abdel-Hamid and Madnick, 1991; Lin and Levary, 1989; Lin et al., 1997; Pfhal and Lebsanft, 1999), but only few experimental analysis were performed to evaluated if such models help project managers in the decisions they make along a development project.

Maier and Stohhecker (1996) developed a study to investigate if *management flight simulators* could support the decisions made by their users. A management game, named LOBSTER, was used in the study. Four groups, each composed by three students, competed in a dynamic market and had to determine whether to invest in equipments, research, and development. Other decisions included price definitions, marketing investments, and staff hiring policies. Two groups used the management game and a simulator, while the remaining groups, which used only the management game, were treated as control groups. Enterprise profits or losses after a predefined number of simulation steps were taken as the performance criteria to compare distinct groups. Due to the small sample, experimental results could not be statistically validated: profits from the groups that used the simulator were not as different from profits achieved by the control groups as required by statistical tests. However, the quantitative results from the experiment show indications that groups that took decisions based on the simulator performed better than the control groups.

Drappa and Ludewig (2000) planned and executed a case study and a controlled experiment to evaluate if students could improve their project management abilities by using software project models. The subjects were asked to answer a questionnaire and prepare a plan for a proposed software project. Next, subjects were allowed to run several simulations using rule-based models that described example software projects (in the controlled experiment, a control group was not allowed to run these simulations). Finally, subjects were requested to fulfill another questionnaire and prepare a new project plan. By comparing the first and the second project plans and the number of correct answer in both questionnaires, the authors inferred if there was ability improvement due to model simulations. In the controlled experiment, the authors compared the results in both groups to evaluate if improvements were due to the simulations. As it happened in the preceding experiment, the results reported for the case study and the controlled experiment were not statistically conclusive: the group that used simulations and the control group had shown very similar performance.

5 Conclusions

This paper described an experimental study that evaluated if system dynamics models can help project managers by supporting the decisions they make during a software development process execution. Eighteen subjects participated in the experimental analysis, impersonating as the manager for a proposed software project. The subjects where divided into two groups. The first group received system dynamics models that could be used while managing the proposed project. Subjects from the second group were asked to manage the project based on their experiences, without using system dynamics models. In average, participants from the first group performed better than participants from the second group. So, experimental results show indications that system dynamics models can be useful to support the decisions taken by project managers.

Since the experimental study was planned as a feasibility analysis, aimed to justify effort investments in the proposed techniques' research, next steps regard the refinement of the scenario based project management paradigm. Current research focus includes the definition of more complex scenario models and applying them to operational project management situations. As the results of the first experimental study had proposed, there is a need for improvements in the tools that currently support the project and scenario models integration and simulation techniques. Also, we intend to provide better tools to help scenario and project model development.

Finally, more experiments are planned to explore system dynamics model application to software project management. Currently, we are executing the experiment described in this paper in an academic-industrial setting[6] in order to verify if the conclusion from the first analysis can be drawn from a different context. Future papers will show a comparison from both experiment application results.

Acknowledgements

References

Abdel-Hamid, T., Madnick, S.E. (1991) *Software Project Dynamics: an Integrated Approach*, Prentice-Hall Software Series, Englewood Cliffs, New Jersey

Barros, M.O., Werner, C.M.L., Travassos, G.H. (2000) "Applying System Dynamics to Scenario Based Software Project Management", *Proceedings of the 18th Conference of the System Dynamics Society*, Bergen, NW (August)

Barros, M.O., Werner, C.M.L., Travassos, G.H. (2001) "From Models to Metamodels: Organizing and Reusing Domain Knowledge in System Dynamics Model Development", *Proceedings of the 19th Conference of the System Dynamics Society*, Atlanta, USA (July)

Barros, M.O., Werner, C.M.L., Travassos, G.H. (2002) "Enhancing Metamodels with Scenarios: Plug-&-Simulate Extensions for Model Developers", to be published in *Proceedings of the 20h Conference of the System Dynamics Society*, Palermo, Italy (July)

Brown, N. (1996) "Industrial-Strength Management Strategies", IEEE Software, Vol. 13, No. 4, pp 94–103 (July)

Drappa, A.; Ludewig, J. (2000) "Simulation in Software Engineering Training", *Proceedings of the 22th International Conference on Software Engineering*, Limerick, Ireland

---

[6] By academic-industrial settings, we mean a software development laboratory from a large computer manufacturing industry that is installed within a Brazilian university. The laboratory staff is mainly composed by students and controlled by both professors and executive managers.

Freund, J.E., Perles, B.M. (1998) *Statistics: A First Course*, Seventh Edition, Prentice-Hall, Englewood Cliffs, New Jersey

IFPUG, 1999, *Function Point Counting Practices Manual, Release 4.1*, Westerville, OH: International Function Point Users Group

Jones, C. (2000) *Software Assessments, Benchmarks, and Best Practices*, Addison-Wesley Information Technology Series, Addison-Wesley Publishing Company, Reading, Massachusets

Klein, G. (1998) *Sources of Power*, MIT Press, Cambridge, Massachusetts

Lin, C.Y., Levary, R.R., 1989, "Computer-Aided Softwar Development Process Design", *IEEE Transactions on Software Engineering*, Vl. 15, No. 9 (Setembro), pp. 1025 – 1037

Lin, C.Y., Abdel-Hamid, T., Sherif, J.S., 1997, "Software-Engineering Process Simulation Model (SEPS)", *The Journal of Systems and Software*, Vol. 38, pp. 263 – 277

Maier, F.H., Strohhecker, J., 1996, "Do Management Flight Simulators Really Enhance Decision Effectiveness", IN: *Proceedings of the 1996 International System Dynamics Conference*, Vol. 2, Cambridge, MA, pp. 341–344

Pfhal, D., Lebsanft, K., 1999, "Integration of System Dynamics Modelling with Descriptive Process Modelling and Goal-Oriented Measurement", *The Journal of Systems and Software*, Vol. 46, pp. 135-150

Pressman, R., 2000, *Software Engineering: A Practitioner's Approach*, 5th Edition, IN: McGraw-Hill Higher Education International Editions, London, UK: McGraw-Hill Co.

Shull, F., Carver, J., Travassos, G.H. (2001) "An Empirical Methodology for Introducing Software Processes", IN: Proceeding of the Joint 8th European Software Engineering Symposium and 9th ACM SIGSOFT Symposium on the Foundations of Software Engineering, Vienna, Austria

Standish Group, T., 1994, *"The Chaos Report"*, The Standish Group Report, available at the URL *http://www.pm2go.com/sample_research/chaos_1994_1.asp*

Sterman, J.D., 1988, "A Skeptic's Guide to Computer Models", Technical Report D-4101-1, MIT System Dynamics Group, Cambridge, MA

Sterman, J.D., 1992, "System Dynamics Modeling for Project Management", *Technical Report*, MIT System Dynamics Group, Cambridge, MA

Wohlin, C., Runeson, P., Höst, M., Ohlsson, M.C., Regnell, B., Wesslén, A. (2000) *Experimentation in Software Engineering: an Introduction*, Kluver Academic Publishers, Norwell, Massachusets