Towards a Scenario Based Project Management Paradigm

MÁRCIO DE OLIVEIRA BARROS
marcio@cos.ufrj.br

THESIS SUPERVISORS: CLÁUDIA MARIA LIMA WERNER
and **GUILHERME HORTA TRAVASSOS**

COPPE / UFRJ – Computer Science Department
Caixa Postal: 68511 - CEP 21945-970 - Rio de Janeiro – RJ
Fax / Voice: 5521 590-2552

Abstract

Complex software development is a risky job. The number of unsuccessful projects largely surpasses the number of successful developments. Many studies relate this situation to non-technical problems, especially to inadequate project management. This thesis intends to develop a scenario based project management paradigm, which is based on separately describing a software project expected behavior and potential problems that can affect this behavior. Risk management, formal process modeling, and simulation support the proposed project management paradigm.

The scenario based project management approach conveys a risk management process, which organizes risk-related information as scenarios and describes a technique to apply this information upon a development project. System dynamic models of software development projects are used to support risk analysis and evaluation. Each development project defines and maintains a project model, which is described by its project elements, such as activities, developers, and artifacts. The project model can be integrated to separately defined scenario models, which affect the software project original behavior. The integration of scenario combinations within the project model allows project behavior estimation over several different situations.

Keywords: Risk management, project management, software process modeling

1 Motivation

The state of practice and contemporary literature report the persistence of a recurrent problem in software development: most projects use more resources than planned, take more time to be concluded, provide less functionality and less quality than expected. Software unsuccessful stories, especially for large development projects, can be found in several case studies and experiments documented over the last decades (Neumann, 1995) (Charette, 1996).

Two different philosophical paradigms relate factors that are responsible for the continuing software faults in industry. The first one relates software development problems to technological limitations, augmented by the increasing complexity of current software projects. The second philosophical paradigm transfers the responsibility to management problems, bad communication, and difficulties in handling the uncertainties that are present in innovative, complex or large software projects (Royce, 1998).

The techniques currently applied to software project management, such as work breakdown structures, PERT/CPM charts, and Gantt diagrams, are based on a set of assumptions, necessary for their accuracy (Charette, 1996). These assumptions require a project behavior to be known from the beginning. The development of projects for innovative application domains, the need for domain integration to achieve several objectives, ambiguity, complexity, discontinuity, diseconomy of scale, nonlinearities, and complex feedback loops are characteristics of large projects. These characteristics undermine the assumptions of traditional techniques. Invalidation of these assumptions, and consequently of these techniques, creates demand for new management paradigms and technologies.

Scenario based project management intends to define a new paradigm for project management. It assumes that a project manager defines and documents an expected behavior for a project process. Since this behavior can be affected by unexpected events during project development, the manager can test its sensibility to several combinations of such events, getting feedback, by simulation, about those activities that can represent a possible risk in the context of the process. Events are described as scenarios, being associated to the project application domain, artifacts created or used, resources consumed, and developers roles. These different project features are generically called **project elements**. Scenarios can be reused by several projects. Scenario based project management is strongly supported by risk management, formal process modeling and simulation techniques.

This paper is organized in five sections. The first section comprises this motivation. Section 2 presents the scenario based software project management paradigm. Section 3 compares the proposed approach to related works. Section 4 presents the expected contributions and research methodology. Finally, section 5 presents on going work.

2 The Scenario Based Software Project Management Paradigm

The following statement briefly describes the traditional view of software project management: plan the flight and fly the plan. This guideline is deeply rooted in the assumptions taken by the currently applied software management techniques. It presumes a single route from early system requirements to the final deployable software product. We advocate that uncertainties, presented along the development process, must be considered in the development plan as alternative routes. We define the scenario based software project management paradigm as a set of techniques that allow a project manager to define a model and several different scenarios for a software development project. These scenarios can be plugged to the project model in order to reveal their impact upon its behavior.

The proposed approach to software project management is based on risk management, project modeling, simulation, and quantitative project tracking. Risk management is applied to identify events that may cause prejudice to the project under development. Project modeling is used to formally describe project known facts, potential problems, and problem resolution strategies. Simulation allows the evaluation of problems and their resolution strategy impact upon project expected behavior. Finally, quantitative project tracking allows periodic project behavior reevaluation, as new information about its status is available.

Scenario based project management paradigm is centered on two artifacts: the project model and scenario models. The project model defines the project expected behavior, while scenarios describe alternative routes that the project may follow, due to unexpected events. Such models are represented using system dynamics, which is a modeling technique and language to describe complex systems focusing on their structural aspects (Forrester, 1991). This technique identifies and models cause-effect relationships and feedback loops. These relationships are described by using flow diagrams composed by four basic elements: stocks, rates, flows and processes. Stocks represent elements that can be accumulated or consumed over time. Its level, that is, the number of elements in the stock, describes a stock at a given moment. Rates describe stock variations, mathematically formulating their raises and depletions in an equation. Processes and flows complement complex rate equations and calculate intermediate model variables.

The project model is based on a modified version of the Abdel-Hamid and Madnick's (1991) system dynamic project model. It is intended for operational management, that is, tracking the project and updating its expected behavior, as new information about its status is available. The enhancements made to the original model include uncertainty representation, providing a high-level project representation, capturing the differences among elements of the same category, and separating facts from policies.

Uncertainty representation allows model parameters to express the uncertainties of their values. Project model parameters can be represented as probability mass functions (Vose, 1996), instead of strictly defined numbers. As the parameters are uncertain, so will be the model results. Those are also described as probability distributions, which are estimated through Monte Carlo simulations (Vose, 1996).

The original Abdel-Hamid and Madnick's model is composed of hundreds of equations, which are very hard to understand and adapt. This representation level makes modeling a new software project a very difficult task, since the modeler must carefully examine each equation for its validity in the new development effort. Scenario based project management defines a higher-level project representation, based on descriptive process models, which can be translated to a system dynamics model. The later is used for simulations, while the first is used to describe a new project without having to handle directly with flows and rates. The high-level representation is based on a software project four vertices spatial figure (Figure 1), a geometric analogy that highlights the model's four major features: its activities, developers, artifacts, and resources.

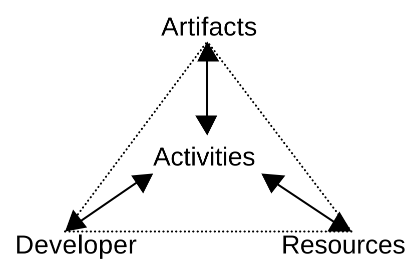

Figure 1 – The software project spatial representation

The original model treats uniformly all project elements of the same category. For instance, all experienced developers are supposed to have the same productivity and all activities are supposed to take the same time to develop. We assume that this model simplification is due to system dynamics inherent incapability to describe element attributes. However, we believe that, if such a model is to be used for operational management, it must capture the differences among its developers, activities, and artifacts. The presented high level representation handles these differences, representing them as project element attributes and generating proper system dynamics equation for them.

Finally, the information within a project model can be classified into two categories: facts and policies. Facts represent information about real world elements, such as developers, and artifacts, and their relationships. Policies are theories and management actions, which change the real world elements' behavior. Most models do not clearly define the borderline between facts and policies. Policies are embedded within facts, their behavior and structure being distributed along hundreds of equations. The project model attempts to separate facts from policies, regarding the latter to scenarios that can be integrated to the model. This separation yields several benefits, such as less complex and more readable models, and general reusable policies that can be applied to a wider range of models.

Scenarios represent the generic and reusable management actions, theories, strategies, policies, and procedures, which can be enacted upon a software project. They can be plugged to the project model, modifying it to represent their consequences and respective expected behavior deviation. The project model describes the scenario integration interface, which is based on the project elements presented in Figure 1, their attributes and relationships. Scenarios can add new attributes to project elements, analyze the values of their attributes, and modify these values, changing their dynamics.

Characterizing the risks faced by a project as scenarios, we define a risk management process based on documenting risks through scenarios, reusing these scenarios along several projects, and simulating scenario combinations for risk evaluation within a particular project. These activities imply two different concerns in the risk management process. First, risks are identified and documented using scenarios. Next, risks are reused along several projects, allowing the simulation of their impact upon project behavior. These different concerns resemble a generic reuse process framework, which is based on two sub-processes: one to develop reusable artifacts and the other for application development based on reusing previously crafted artifacts. Applying the generic process framework to the proposed risk management process, we divided it into two sub-processes: one for risk identification, the other for risk reuse during application development. The reusable artifact shared by both processes, called risk archetype, conveys information about a software risk, including scenarios for its impact evaluation and for resolution strategies.

3 Related Works

The scenario based project management approach can be compared to operational project management models. The proposed project model conveys more information than the work breakdown structure or the PERT/CPM chart, which focus on a project activity network, ignoring other project elements, such as developers, resources, and artifacts. The proposed project model accounts for those elements, allowing scenarios to describe their dynamic characteristics, such as developers' motivation, team exhaustion, and error propagation among artifacts. More sophisticated operational models, such as DesignNet (Liu and Horowitz, 1989) and ProNet (Christie, 1995), also characterize those project elements. However, their static representation and fixed attributes limit the analysis to predetermined dynamic characteristics. Scenarios let the manager define new attributes or modify previously defined attributes, adding new dynamic characteristics to the project model as scenarios are integrated to it.

When compared to other system dynamic software project models, four features differentiate the proposed project model. The first one is the high-level representation, which allows a more suitable project description than system dynamics equations. Secondly, due to high-level representation, the model can handle attributes for each project element, differentiating elements of the same category. Previous system dynamics models, such as SLICS (Lin and Levary, 1989) and SEPS (Lin et al., 1997), generally assume that all elements of the same category are equal. Thirdly, uncertainty representation allows the description of model parameter values as probability distributions, instead of crisp numbers. Finally, the project model defines a scenario integration interface, which allows the model to be extensible. Tvedt (1996) presents a system dynamics model, which allows an organization to evaluate the implementation of process improvement alternatives over its development process. Although the model was developed to be extensible, it has to be modified each time a new process improvement model is to be evaluated. In the proposed approach, a scenario would define the improvement model, being integrated to the project model without modifications to the later.

4 Expected Contributions and Research Methodology

The expected contributions of this work are:

- a risk management process definition that uses dynamic models to evaluate the impact of software project risks and their resolution strategies;
- a high-level project model description that can be translated to system dynamics, and;
- the integration of such project model to separate scenario models.

We describe these as contributions because our assumption is that they allow the construction of reusable models of software engineering theories, management actions, policies, and procedures, which can be integrated to an application specific project model in order to evaluate software project behavior under several different hypotheses combinations.

Besides, the proposed high-level representation can leverage future work on software project modeling, acting as a complexity reduction technique, since we assume that it frees the modeler for dealing with several low-level equations.

The current stage of the project is as follows: a language to represent system dynamics models has been developed. This language allows the definition of stocks, rates, and processes. The equations representing such elements can express their uncertainties using gaussian and uniform probability distribution functions. Other distributions, such as Poisson and beta, will be incorporated to the model, since these techniques seem to be useful to project management. Moreover, a sort of simulation mechanisms, including time-based, Monte Carlo, and parameter variation simulation has also been developed, which can be used to evaluate system dynamics models. Results taken from this tool, its language, and simulation mechanisms are presented in (Barros et al, 2000b) and (Barros et al, 2000c).

A system dynamic metamodel, which is based on entities and their relationships, is under development. Entities convey parameters, state variables, and auxiliary behavior. The metamodel can be instantiated to a model, where parameter values are defined, implicit behavior from state variables and auxiliary behavior is replicated for each entity, and entity relationships are resolved. The instantiated model is translated to system dynamics, in order to be simulated. The metamodel will allow the proposed high-level representation for software projects, being instantiated for each new application to be developed. A language to support the metamodel representation and a translator to system dynamic equations will be available. We intend to evaluate the language by first developing simple models, such as stochastic PERT nets, and progressively develop more complex ones, culminating with the proposed project model. To support these activities some modifications were proposed to the Abdel-Hamid and Madnick's model, which are detailed in (Barros et al, 2000a).

We have also defined a first proposal for the scenario based project risk management process, describing in some detail the activities of its sub-processes, and the artifacts they produce and consume. The risk archetype information structure was defined and some initial examples were provided (Barros et al, 1999).

Finally, we plan to run some case studies in order to evaluate the feasibility of such techniques. Those case studies are intended to apply the techniques in real projects. A first case study is scheduled to occur by the end of August/2000, where the techniques will be applied to a software project, which aims to specialize a financial risk management software package for a large Brazilian enterprise. We expect this case study to provide some concrete examples of risk archetypes and illustrate their relationships with specific project elements, as for instance, the application domain. We also expect to observe during the case study useful information that allow us to refine the techniques and, when the system dynamic metamodel is finished, to develop some scenario models. Moreover, we hope to be able to raise some points that allow us to identify a set of criteria that can be used to evaluate the proposed approach. By doing so, we plan to run, if possible, a second study to analyze the effects of the refinements resulted from the first case study.

5 On Going Work

This paper describes a Sc.D. research project that is under development at the Computer Science and Systems Engineering Department at COPPE/UFRJ, aiming at the definition of a software project management paradigm based on risk analysis and system dynamic models. Separate models represent known facts about the software project and exceptional events, management policies, procedures, and theories that can affect the project. We have presented the current status of the research, the methodology applied within its development, and some case studies that we intend to execute for its validation.

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. (1999) “Risk Analysis: a Key Success Factor for Complex System Development”, IN: *XII International Conference Software & System Engineering and its Applications*, Paris, FR (December)

Barros, M.O., Werner, C.M.L., Travassos, G.H. (2000a) “Applying System Dynamics to Scenario Based Software Project Management”, IN: *The Proceedings of the 2000 International System Dynamics Conference*, Berghen, NW (August)

Barros, M.O., Werner, C.M.L., Travassos, G.H. (2000b) “ILLIUM: Uma Ferramenta de Simulação de Modelos Dinâmicos de Projetos de Software”, IN: *The Proceedings of the Tools Section of the XIV Brazilian Symposium on Software Engineering*, João Pessoa, BR (October), in portuguese

Barros, M.O., Werner, C.M.L., Travassos, G.H. (2000c) “Using Process Modeling and Dynamic Simulation to Support Software Process Quality Management”, Submitted to: *The Proceedings of the WQS’2000 – Workshop de Qualidade de Software*, João Pessoa, BR (October)

Charette, R.N. (1996) “Large-scale Project Management is Risk Management”, *IEEE Software*, Vol. 13, No. 4, pp 110-117, (July)

Christie, A. (1995) *Software Process Automation: The Technology and Its Adoption*, Springer-Verlag Publishing, Berlin

Forrester, J.W. (1991) “System Dynamics and the Lessons of 35 Years”, *Technical Report D-4224-4*, Sloan School of Management, Massachusetts Institute of Technology

Lin, C.Y., Levary, R.R. (1989) “Computer-Aided Software Development Process Design”, *IEEE Transactions on Software Engineering*, Vol. 15, No. 9, pp 1025-1037 (September)

Lin, C.Y., Abdel-Hamid, T., Sherif, J.S., (1997) “Software-Engineering Process Simulation Model (SEPS)”, *Journal of Systems and Software*, Vol. 37, pp. 263-277

Liu, L., Horowitz, E. (1989) “A Formal Model for Software Project Management”, *IEEE Transactions on Software Engineering*, Vol. 15, No. 10, pp. 1280-1293 (October)

Neumann, P.G. (1995) *Computer Related Risks*, Addison-Wesley Publishing Company, Reading, Massachusets

Royce, W. (1998) *Software Project Management: A Unified Framework*, Addison-Wesley Publishing Company, Reading, Massachusets

Tvedt, J.D., 1996, “An Extensible Model for Evaluating the Impact of Process Improvements on Software Development Cycle Time”, *Tese de D.Sc., Arizona State University*, Tempe, AZ

Vose, D. (1996) *Quantitative Risk Analysis: A Guide to Monte Carlo Simulation Modeling*, John Wiley & Sons, Inc, New York