IMPROVING SOFTWARE PROCESS MATURITY THROUGH DYNAMIC MODELING AND SIMULATION

Mercedes Ruiz[1p], Isabel Ramos[2], Miguel Toro[2]
Department of Computer Languages and Systems
[1] Escuela Superior de Ingeniería. University of Cádiz (Spain)
[2] Escuela Técnica Superior de Ingeniería Informática. University of Seville (Spain)

Resumen

Los modelos de procesos actuales como CMM, SPICE y otros recomiendan la aplicación de control estadístico y de guías de métricas para la definición, implementación y posterior evaluación de diferentes mejoras del proceso. Sin embargo, precisamente en este contexto no se ha considerado lo suficiente el modelado cuantitativo, reconocido en otras áreas como un elemento esencial para la adquisición de conocimiento. En este trabajo se describe la base conceptual y fundamental utilizada para el desarrollo de un marco enfocado a la mejora de procesos software que combina las técnicas de estimación tradicionales con la utilización extensiva de modelos dinámicos de simulación como herramienta para asesorar en el proceso de evolución entre los diferentes niveles de madurez propuestos por el modelo de referencia CMM. Tras la necesaria introducción a los conceptos fundamentales del modelado y simulación del proceso software y la justificación para la creación de dicho marco, se abordan las cuestiones fundamentales para su desarrollo, tales como el enfoque conceptual y su estructura, prestando especial atención al paradigma de desarrollo de los modelos dinámicos de simulación que le dan soporte.

Abstract

Current software process models (CMM, SPICE, etc.) strongly recommend the application of statistical control and measure guides to define, implement and evaluate the effects of different process improvements. However, whilst quantitative modelling has been widely used in other fields, it has not been considered enough in

the field of software process improvement. During the last decade software process simulation has been used to address a wide diversity of management problems. Some of these problems are related to strategic management, technology adoption, understanding, training and learning, and risk management, among others. In this work a dynamic integrated framework for software process improvement is presented. This framework combines traditional estimation models with an intensive utilisation of dynamic simulation models of software process. The aim of this framework is to support a qualitative and quantitative assessment for software process improvement and decision making to achieve a higher software development process capability according to the Capability Maturity Model. The concepts underlying this framework have been implemented in a software process improvement tool that has been used in a local software organisation. The results obtained and the lessons learned are also presented in this paper.

1 INTRODUCTION

Dynamic modelling and simulation as process improvement tools have been intensively used in the manufacturing area. Currently, software process modelling and simulation are gaining an increasing interest among researchers and practitioners as an approach to analyse complex business and solve policy questions.

In this paper an approach is proposed that combines traditional estimation techniques with System Dynamics modelling. The aim of this combination is to build a framework to support a qualitative and quantitative assessment for software process improvement and decision making. The purpose of DIFSPI (*Dynamic Integrated Framework for Software Process Improvement*) is to help organisations to achieve a higher software development process capability according to the Capability Maturity Model (Paulk et al. 1993). The dynamic models built inside this framework provide the capability of gaining insight over the whole life cycle at different levels of abstraction. The level of abstraction used in a certain organisation will depend on its maturity level. For instance, in a level 1 organisation the simulator can establish a baseline according to traditional estimation models from an initial estimate of the size of the project. With this baseline, the software manager can analyse the results obtained with the simulation of different process improvements and study the outcomes of over or underestimate of cost or schedule. During the simulation metric

data are saved. These data conform to SEI core measures recommendation (Carleton et al. 1992) and are mainly related to cost, schedule and quality.

The structure of the paper is as follows. Section 2 provides a brief overview of the work conducted in the field of software process simulation. In Section 3 the fundamental basis and structure of this framework are described. The implementation and results obtained when applying it inside a local organisation are discussed in Section 4. Finally, Section 5 summarises the paper and draws the conclusions and lessons learnt.

2 SOFTWARE PROCESS SIMULATION

Simulation can be applied in many critical areas in support of software engineering. It enables one to address issues before these issues become problems. Simulation is more than just a technology, as it forces one to think in global terms about system behaviour, and about the fact that systems are more than the sum of their components (Christie 1999). A simulation model is a computational model that represents an abstraction or a simplified representation of a complex dynamic system. Simulation models have as a main advantage the possibility of experimenting different management decisions.

Thus it becomes possible to analyse the effect of those decisions in systems where the cost or risks of experimentation make it unfeasible. Another important factor is that simulation provides insights into complex process behaviour which is not possible to be analysed by means of stochastic models. Like many processes, software processes can contain multiple feedback loops, such as associated with correction of defects. Delays resulting from these defects may range from minutes to years. The resulting complexity makes it almost impossible for mental analysis to predict the consequences.

The common objectives of simulation models consist on supplying mechanisms to experiment, predict, learn and answer questions such as *What if ...*? A software process simulation model can be focussed on certain aspects of the software process or the organisation. It is important to bear in mind that a simulation model constitutes an abstraction of the real system, and so, it represents only the parts of the system that have been intended to be modelled. Furthermore, currently available modelling tools such as STELLA, POWER-SIM and Vensim help to represent the software development process as a system of differential equations. This is a

remarkable characteristic as it makes it possible to formalise and develop a scientific basis for software process modelling and improvement. Some noticeable applications of the dynamic approach to model software process can be found in (Kellner et al. 1999).

3 DIFSPI STRUCTURE

Project management is composed of activities that are intimately interrelated in the sense of that a certain action performed over a determined area will possibly affect other areas. For instance, a time delay will always affect the cost of the project but it may or may not affect the morale of the development team or the quality of the product. The interactions among the different areas of project management are so strong that on some occasions the throughput of one of them can only be achieved by reducing the throughput of another. A clear example of this behaviour can be found in the frequent practice of reducing the quality or the number of requirements to be implemented in a certain version of the product with the aim of accomplishing the time or cost estimates.

Dynamic models help to understand this integrated nature of project management as they describe it through different processes, structures and main interrelationships. In the framework proposed here, project management is considered as a set of dynamic interrelated processes. Projects are composed of processes. Each process is composed of a series of activities designed for the achievement of an objective (Paulk et al. 1993). From a general point of view, it could be said that projects are composed of processes that fall in one of the following categories:

- Management process. This category collects all those processes related to the description, organisation and control of the project.
- Engineering process. All those processes related to the specification and development activities of the software product are collected in this category.

Both categories interact during the time cycle of the project. From an initial plan performed by the project management processes, engineering processes begin to be executed. Using the information gathered about the progress of this second group of processes, project management processes determine the modifications which need to be made to the plan in order to achieve the project objectives. The DIFSPI proposed follows this same classification and it is structured attending to project

management and engineering processes. In both levels, the utilisation of dynamic models to simulate the real processes and to define and develop a historical database will be the main feature

The engineering processes in the DIFSPI the dynamic models simulate the life cycle of the software product. The benefits that simulation provides at this level are the following:

- To build a model it is necessary to improve the knowledge one has about the software development process as it is required to establish the limits and the scope of those real behaviours to be modelled and simulated.
- The parameters required by the model and the tables that determine its time behaviour will constitute the main elements of a metrics collection program to define a historical database.
- The effective application of this metrics program will feed the database. The historical data gathered will assess in the validation and calibration of the model.
- The dynamic model will finally simulate the software processes with the knowledge and the maturity that the organisation has at the moment.
- The utilisation of the dynamic model allows the establishment of a baseline for the project, the investigation of possible improvements, and the development of a historical database which can be fed either by real or simulated data.

The dynamic models of this level at DIFSPI should follow the levels of visibility and knowledge of the engineering processes that organisations have at each maturity level. It is obvious that the complexity of the dynamic model used in level 1 organisations cannot be the same as that one of the models capable of simulating the engineering processes of, for instance, level 4 organisations.

Management processes are divided into two main categories:

- Plan. It groups the processes devoted to the design of the initial plan and the required modifications when the progress reports indicate the appearance of problems. The models of this group integrate traditional estimation and planning techniques together with dynamic ones.
- Control. In this group all the models designed for the monitoring and tracking activities are gathered. These models will also have the responsibility of determining the corrective actions to the project plan. Therefore, the simulation of

process improvements will be of an enormous importance.

4 DIFSPI UTILISATION

The potential applications of the DIFSPI have already been mentioned in the former sections. In this section some of the data obtained when DIFSPI was applied inside a local software development organisation are provided. This local organisation could be placed at level 1. At first the software process capability of this organisation was unpredictable because it was constantly changed or modified as the work progressed. Performance depended on both the capabilities of the project manager and the technical team. Moreover, there were few stable software processes in evidence. According to Level 1 organisations, the software process here was perceived as an amorphous entity, ‘a black box’, and visibility into the project's processes was very limited. Requirements flowed into the software process in an uncontrolled manner, giving a product as a result. The purpose of this application was to insure that the framework could reproduce the behaviour observed in a real project and, therefore, could trigger a metrics collection program, and help in decision making, predicting and cost estimating. Table 1 shows the characteristics of the project that was simulated for this case study together with the data of the baseline reported by the simulation. It should be noted here that the data reported by the simulation conforms the core measures recommended by the Software Engineering Institute (SEI) (Carleton et al. 1992).

Size of the project = 80,000 LOC			
REAL DATA		**SIMULATED DATA**	
Time	250 days	*Time*	263 days
Initial Workforce	8 technician	*Effort*	4,361 technician-day
Effort	4,780 technician-day	*Quality*	80% (tasks revised)
		Workforce	9 technician

Table 1: Real and simulated data for the case study

The scenario shown in Table 2 helps to analyse the impact of the size of the technical staff over the main four variables (time, effort, quality, and overall workforce). Two different cases were simulated. The first one (CASE 1) had a

schedule of 250 days and 16 part-time technicians. The second case (CASE 2) had a schedule of 150 days and 16 full-time technicians.

The expected behaviour for projects with a high level or personnel is that the average productivity per technician achieved will be lower. The average productivity per technician in the baseline was 0.8926 tasks/(technician*day). CASE 1 and 2 both had the double initial workforce than that of the baseline, although schedules and resource allocation were different between them. The average productivity obtained for case 1 and 2 was, respectively, 0.8277 tasks/(technician*day) and 0.8142 tasks/(technician/day).

CASE 1		**CASE 2**	
Time	135 days	*Time*	140 days
Effort	1,396 technician-day	*Effort*	3,596 technician-day
Quality	91%	*Quality*	91%
Workforce	18 technician	*Workforce*	16 technician

Table 2: Simulated data for scenario analysis

5 CONCLUSIONS

Motivated by lessons learnt from another System Dynamics application in an industrial environment, the development of a framework to combine the traditional estimation tools with the dynamic approach has been initiated. The main objective of this dynamic framework is to assess project managers and members of the SEIG to define, evaluate and implement process improvements to achieve higher levels of maturity. The whole process of development of the framework also helps to design a specific metrics collection program which, once implemented, contributes to build and feed a historical database inside an organisation.

With the application of DIFSPI in a level 1 organisation important benefits were obtained. First, it must be mentioned that during the process of model building, the project manager gained much new insight into those aspects of the development process that mostly influence the success of the project (time, cost and quality). Second, having the possibility of gaming with the DIFSPI, it allowed him to better understand the underlying dynamics of the software process. As a consequence, several process improvement suggestions were easily designed and, most

importantly, analysed using simulation of scenarios. Finally, templates and guidelines for a metrics collection program were almost automatically derived from the requirements of the dynamic modules.

Our future work will mainly concentrate on research towards a full development of the dynamic modules that implement the key process areas of the higher maturity levels. Once this development has been accomplished it is intended to validate the complete DIFSPI in real industrial environments.

REFERENCES

1. Paulk, M., Garcia, S.M., Chrissis, M.B., Bush, M., 1993. Key practices of the capability maturity model. Version 1.1 Technical Report CMU/SEI-93-TR-25. Software Engineering Institute, Carnegie Mellon University, Pittsburg, PA.
2. Carleton, A., Park, R.E., Goethert, W.B., Florac, W.A., Bailey, E.K., Pfleeger, S.L., 1992. Software measurement for DoD systems: recommendations for initial core measures. Technical Report CMU/SEI-92-TR-19. Software Engineering Institute, Carnegie Mellon University, Pittsburg, PA.
3. Christie, A.M., 1999. Simulation in support of CMM-based process improvement. *The Journal of Systems and Software*, 46, (1999), 107-112.
4. Kellner MI, Madachy R, Raffo D. Software process simulation modeling: Why? What? How? *The Journal of Systems and Software*, 46, (1999), 91-105.

CORRESPONDENCE

Mercedes Ruiz Carreira
Dpto. de Lenguajes y Sistemas Informáticos
E.S. de Ingeniería
C/ Chile, nº1
11003 - Cádiz (Spain)
Phone: +34 956 015 714 Fax: +34 956 015 139
e-mail: mercedes.ruiz@uca.es

Acknowledgements

The authors wish to thank to Comisión Interministerial de Ciencia y Tecnología, Spain, (under TIC2001-1143-C03-02) for supporting this research effort.