A tool for evaluation of the Software Development Process

Peter Henderson
(P.Henderson@ecs.soton.ac.uk)
Declarative Systems and Software Engineering Group
Department of Electronics and Computer Science
University of Southampton
Southampton, UK.
SO17 1BJ

Robert Walters
(R.J.Walters@ecs.soton.ac.uk)
Declarative Systems and Software Engineering Group
Department of Electronics and Computer Science
University of Southampton
Southampton, UK.
SO17 1BJ

Introduction

There is continuing pressure on Software Developers to improve both the quality and speed of development of software. One response is to review and to improve the software development *process.* There are accepted schemes [4, 7] which attempt to classify the software development process (not the software produced) according a measure of its "quality" or "maturity". However, few software developers have high ratings on these scales.

One reason is that, despite their modest ratings, software developers have a considerable investment in their existing development process and the infrastructure supporting it. They could not adopt an "ideal" process even if such a process could be identified and defined. Instead, they need to adapt and enhance their existing processes. To achieve the highest ratings, developers need to measure and manage their processes.

We describe RolEnact which attempts to address the requirements for a tool to support the simulation, evaluation and improvement of software development processes.

Motivation

Most schemes and tools directed at helping developers to monitor and improve their processes have a notion of how the process operates which is essentially fixed. To make these systems work the developers may need to collect data which is not readily available and be prepared to accept analysis results which require further interpretation.

For example, consider an experiment carried out by ICL to improve cost estimation of software projects [1]. The experiment looked at a novel method for monitoring and controlling the progress of a software project. The results show two typical features of this type of exercise:

- The figures collected are analysed manually because either the figures themselves or the analysis required cannot be handled by the existing support system.

- The figures show unexpected features. In this case, although it is unlikely to be true, the data suggests that almost half of the work of the project was carried out in the penultimate month.

The first of these problems is often addressed by the construction of a custom support tool which is tailored to the particular situation. However, once built and implemented, the new tool can be expected to cause the similar problems to the one it replaced. "Exceptional" behaviour may be dealt with by adjustments to the data to more accurately reflect "what must have happened" or by explaining away the analysis results. Neither of these approaches is really satisfactory.

We believe that a better approach is to modify the analysis of the model and the model itself in the light of the observed behaviour. To do this, we need analysis and simulation tools which are able to adopt and evolve with (understanding of) the process under consideration.

Requirements for an evaluation tool

The tool must be able to work with an understanding of the existing process. Typically this process will be unique to the developer as it will have evolved over a period of time.

The tool needs to be easy to understand as many of the people who will use it will not be familiar with process modelling.

Once a description of the existing process (or part of the process) has been captured, the tool must then be able run simulations of the process using appropriate data from the real process.

The tool needs to be able to adapt to the process under consideration as it develops and to allow novel views of those processes as well as providing features to enable analysis and evaluation of the process.

Description of RolEnact

RolEnact is a collection of tools which support the development and analysis of "role based" models [2, 3, 6]. The complete set comprises a model building/visualising tool together with a "stepper" and the simulator. RolEnact describes processes as a collection of interacting "Roles". Each of these "Roles" can be thought of as representing the behaviour an individual (or job) within the process. These Roles are able to create new instances of Roles as well as perform internal changes of state and interactions with other Roles.

The RolEnact Model Generator

For a process model to be useful, it needs to be accessible to everyone involved in the process. Our experience suggests even a simple model description language like that used by RolEnact represents a significant barrier to many people. The generator enables a modeller to build a model without writing code. A model is constructed using a collection of dialogue boxes and the modeller works by making selections from lists wherever possible. To assist the inexpert modeller further, the generator is also able to display a graphical representation of the model as a "RAD-like" [5] diagram.

The RolEnact Stepper

A completed model may be simulated manually using the stepper. In a running simulation, each instance of a "Role" has its own window which shows its state and a list of the events it is presently able to perform. The modeller steps through the model by successively selecting an event from the lists of available events in the Roles of the model. No specialised knowledge is required to animate a model using the stepper.

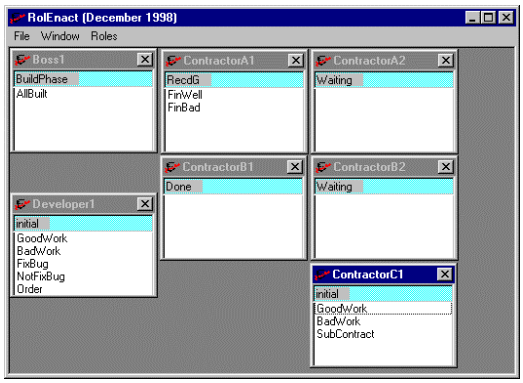

Figure 1: RolEnact running a model of a development project

The RolEnact Simulator

The RolEnact Simulator takes a RolEnact model and runs it automatically, recording the results in an Excel workbook. After each simulation, the results can be examined and analysed using Excel. We have adopted this approach to give the modeller both the power and flexibility of Excel for analysis and an easy route to integrating the results into documents and presentations.

The model shown in the figures has been constructed to examine the effects differing sub-contracting strategies on the quality and delivery time of the final project. The Role "Boss" represents the customer, the "Developer" role represents the prime contractor. The "Developer" can either build (parts of) product itself or place work with one or more contractors (who may also choose to sub-contract work). At each stage the work of each of the Roles can be either "good" or "bad". The relative numbers of these types of event is used in an analysis of each run of the model as a measure of the quality of the product. The Developer also has the ability devote effort to attempting to correct problems.

As an example of the type of analysis that might be performed, if the developer were to be more inclined to pass work to contractors, then it seems reasonable to expect that a poor sub-contractor will have a detrimental effect on the quality final product. At the same time, using more contractors might lead to an improvement because of the extra time and effort the developer is able to devote to correcting problems. The number of interactions between players in the process and the extent of their interdependence makes it hard to understand the effect of such changes. However, a complete understanding is not required. In this context, the information which

is really needed is: if *this developer* were to change their process in this way, what would the effect be on the quality of the product? Using the RolEnact Simulator and figures from the real process, it is possible to predict the effect of a change in the developer's policy regarding sub-contracting.

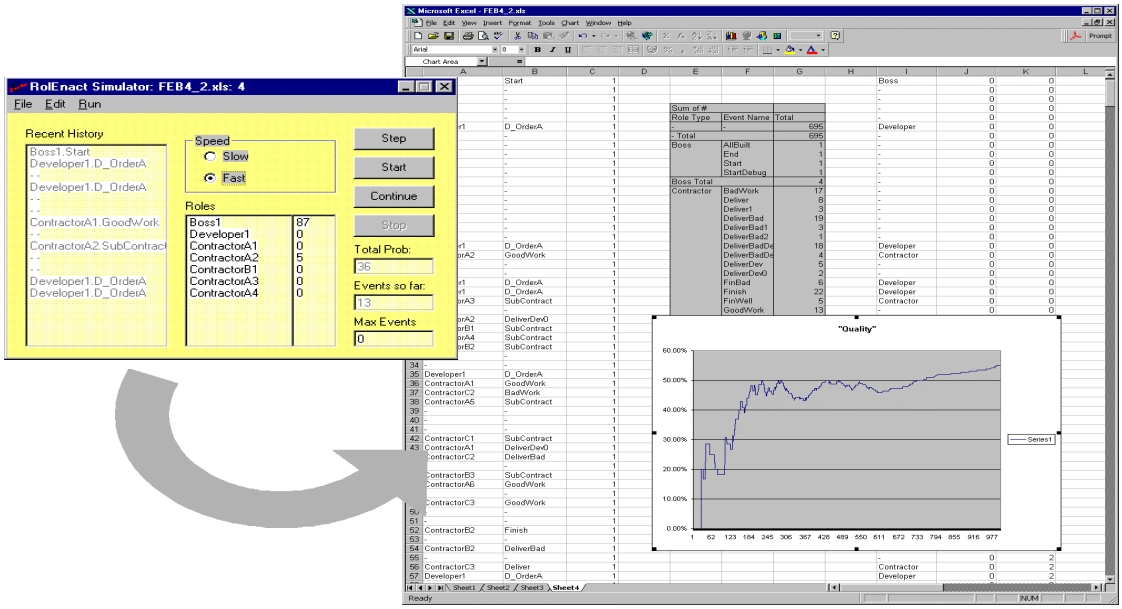

Figure 2: RolEnact Simulator paused whilst running a simulation of the project modelled in Figure 1.

Conclusion

Today we use huge software systems which are both highly featured and reliable and it would be easy, based on the size, reliability and performance of these systems, to assume that the problems of building them have been solved. However, the process for creating software systems is much less mature than the products we see and this is evidenced by the low ratings of most software developers in evaluations like CMM and SPICE [4, 7].

Software developers have major investments in their existing processes and support systems which they cannot afford to abandon so, if they are to achieve a high rating on these scales, they need to evaluate and improve their existing processes. To do this successfully, they need a new breed of flexible process modelling and analysis tools.

References

[1] B. Chatters and P. Henderson, "An Experiment to Improve Cost Estimation and Project Tracking for Software and Systems Integration Projects," *EuroMicro99*, Milan, 1999, pp.

[2] P. Henderson and R.J. Walters, "Component Based systems as an Aid to Design Validation," *14th IEEE International Conference on Automated Software Engineering (ASE99)*, Cocoa Beach, Florida, 1999, pp. 303-306.

[3] P. Henderson and R.J. Walters, "System Design Validation Using Formal Methods," *Tenth IEEE International Workshop on Rapid System Prototyping (RSP99)*, Clearwater, Florida, 1999, pp. 10-14.

[4] J. Herbsleb, D. Zubrow, D. Goldenson, W. Hayes, and M. Paulk, "Software quality and the Capability Maturity Model," *Communications of the Acm*, vol. 40, pp. 30-40, 1997.

[5] M.A. Ould, *Business Processes - Modelling and Analysis for Re-engineering and Improvement*: John Wiley and Sons, 1995.

[6] K.T. Phalp, P. Henderson, G. Abeysinghe, and R.J. Walters, "RolEnact - Role Based Enactable Models of Business Processes," *Information And Software Technology*, vol. 40, pp. 123-133, 1998.

[7] J.M. Simon, "SPICE: Overview for software process improvement," *Journal of Systems Architecture*, vol. 42, pp. 633-641, 1996.