

Modelling Process Dynamics in Software Evolution Processes - Some Issues

Submitted to the Workshop on Software Change and Evolution (SCE'99)

Los Angeles, May 17, 1999

JF Ramil

Dept. of Computing
Imperial College

180 Queen's Gate, London SW7 2BZ

tel +44 171 594 8216 fax +44 171 594 8215

jcf1@doc.ic.ac.uk

<http://www-dse.doc.ic.ac.uk/~jcf1/>

MM Lehman

Dept. of Computing
Imperial College

180 Queen's Gate, London SW7 2BZ

tel +44 171 594 8214 fax +44 171 594 8215

mml@doc.ic.ac.uk

<http://www-dse.doc.ic.ac.uk/~mml/>

Motivation

The continuing changes in the *requirements* that define the properties that software is required to have arise from many sources. Users have new or changed needs, development organisations seek to increase revenue or enlarge the customer base, developers wish to incorporate new concepts or ideas and so on. This position paper is not the place to elaborate on the diverse sources of change. It is a truism to observe that the list of *possible* changes is very large, potentially unbounded. Implementation of even *desirable* changes is likely to exceed the resources or capacity of the software organisation and its process in any given interval. Organisations would hope to complete essential work before clients become dissatisfied, before business opportunities are lost. The process of continuing software evolution is, in fact, essentially, dynamic, sensitive at each moment to the needs of the various stakeholders. Appropriate planning and management strategies must be applied.

Software process dynamics and simulation have been used by a number of investigators to address various aspects of the resultant dynamics and to support software evolution management. A model based on some specific technology¹ is built and calibrated. To the extent that the calibration has been validated its behaviour in execution can then be examined. For example, the effect of and sensitivity to changes in model inputs and parameters can be observed and analysed and provide an impression of the effect of changes on the process itself.

Modelling of software process dynamics has, however, not attracted widespread interest. In the first place the impact of the dynamics on product and process evolution has not been widely recognised. The level of expertise, data² and resources required to build and calibrate such models did not appear justified in a perception of limited benefit. It must also be admitted, their use raises difficult issues, for example:

- does the model (and the mechanisms depicted therein) relate significantly to real mechanisms present in the software process, and if yes, how?
- assuming that the model was, in some sense, representative of the process at a some time, is it still?
- can the model be systematically updated to remain a faithful representation of the process dynamics?

Potential sources of modelling uncertainty are many and diverse. One may identify many, probably hundreds, of drivers, forces and constraints at play in a software process. Some of them tend to improve performance, others tend to degrade it. In general, the software, exposed to change upon change upon change, is in a state of flux and tends to become more complex unless work is done to control growing complexity (second law of software evolution [leh74]). The process itself also evolves. Changes in resources, people, tools and technology, for example, are inevitable. Humans introduce uncertainty

¹ For example, system dynamics [abd91], Petri nets [kus97], combinations of techniques (eg system dynamics and fuzzy logic [lev90]) and, most recently, fuzzy dynamics [ram98].

² Here and in the remaining of this paper *data* refers in general to the data required to build and calibrate a dynamic model of the software process.

[leh77]. Exogenous forces may play also an important role. Thus it appears that, *prima facie*, one should not assume a degree of long term disciplined dynamics. Empirically based observations and related analysis formulated in the seventies and eighties, together with more recent evidence, suggest, however, that evolution activity displays a degree of disciplined behaviour. Such behaviour can be exploited in the context of software evolution management. After a brief account of the results of a recently completed investigation, this position paper discusses in outline some issues related to process dynamics modelling.

FEAST/1 Results and FEAST/2

The recently completed FEAST/1 project [leh96] investigated a hypothesis which stated that *software processes are multi-level, multi-loop, multi-agent feedback systems and have to be treated as such to achieve significant and sustained process improvement*. FEAST/1 collected metrics and indicators from five different software systems from four different industrial organisations³. Software processes were modelled using both, black box [leh97,98c,d,e] and white box (system dynamics) [cha99,leh98a,wer98] techniques. The focus of the project was the *global* software process that encompasses the activities of all involved in the process, developers, managers, marketeers, support staff, users and so on. The approach adopted was different to general process modelling practice in which the object modelled has been the *technical* process, that is the process that takes a software product from concept and requirements to system installation and operation.

The FEAST/1 investigation yielded observations that were consistent with the above stated hypothesis and supported the laws of software evolution [leh74,85], with only minor refinements of the latter [leh98c,e]. The project has identified regularities and patterns over the growth and change data of several systems evolving for periods of up to 15 years. For example, an inverse square model has proven a remarkably good growth model for the systems studied [tur96,leh97,98c]. The FEAST hypothesis suggests a mechanism that appears to be the source of the regularities observed. They are a manifestation of a global dynamics that results from the embedding of the technical software process in the various domains within which it is contained [leh98e].

FEAST/1 results indicate that software process dynamics is a *real* phenomenon with significant process impact. The question remains whether and how this phenomenon can be exploited in the context of software evolution management. One begins by *identifying* the phenomenon. In FEAST/1 this was achieved by data modelling based on historical data sets. It is now apparent that it can also be achieved, for example, *on line* by *updating* process behavioural models as product and processes evolve. This approach can, *inter alia*, yield realistic models of dynamic behaviour, providing the basis to achieve a degree of control of the software evolution process. A degree of *uncertainty* will, however, always remain [leh77]. Investigation of this and related topics will be pursued in the FEAST/2 study, now under way [leh98b].

Process Dynamic Modelling

Since first attempts by Riordon [rio77] to simulate the dynamics of software processes, the underlying modelling *procedure* has remained more or less constant. On the basis of questions about process properties, a model is conceived. The investigating team of one or more persons, having selected a particular modelling technique, observes a process in operation (or a process in conceptualisation stage). This may, for example, be done, by direct observation, participation in process activities and/or interaction with those involved. A model, which abstracts the mechanisms and interactions believed to be dominant is then developed and calibrated using data from various sources. Direct measurement and expert estimation will play a part, both in the make-up of the model and its calibration. The model

³ ICL, Logica and Matra-BAe are the official collaborators. Lucent Technologies became a *de facto* collaborator through the good offices of Dr. D E Perry who was associated with FEAST/1 as a Senior Visiting Fellow.

becomes an artefact for reasoning about the process, its product and its environment. Answers to relevant questions are sought by running the model. The questions may relate to process behaviour, design or improvement, for example. Model building becomes a learning process for all involved. It may well be the most valuable part of the exercise.

Use of the model for *prediction*, makes the establishment of confidence in its validity particularly important. Ways of achieving confidence include, for example, that:

- the model and its outputs are considered reasonable by appropriate experts
- with inputs of past states the model replicates past behaviour
- there is a definable (possibly formal or algorithmic) relationship between the model and data (historical, current, emerging)

In general, only the first two, however, appear to have been used in software process modelling. We suggest here that demonstration of a definable relationship between the model and the data will be facilitated by attaching an *on line parameter estimation mechanism* [lju87,wel91] to the model to update its parameters as new observations become available. A sequence of change requests or maintenance activities (always present in one way or the other in operational *E*-type evolution processes) represent a sequence of inputs in the system-identification sense [lju87wel91]. Their presence offers an opportunity to explore this approach. There are reasons to believe that Bayesian parameter estimation will also be useful in this context, since it enables explicit representation of *a priori* beliefs about the parameters to be identified, providing methods to update the parameter pdfs as new data becomes available. Moreover the Bayesian approach seems to outperform sampling approaches when considering small sequences of observation [zel87].

Definable relationships between data and models, and on line updating of the latter, will also facilitate software process performance monitoring. It is likely to be an improvement on direct inspection of process measurements and indicators as in statistical process control [wet91]. *Condition monitoring* techniques may help to identify whether an observed deviation corresponds to an actual change in software process dynamic characteristics (and its performance) or is due to exogenous forces [gus98].

Many questions need to be addressed before this proposal can be fully implemented. Since the parameter estimation mechanism and the model have to be in some sense compatible with each other it raises an additional consideration whose impact has to be evaluated. How, for example, are on line parameter estimation mechanisms to be attached to models such as those described in the software process modelling literature [eg abd91,cha99,kus97,leh98a,pro98,wer98]? Moreover, the attempt to attach a parameter estimation technique to large models raises issues of *identifiability*⁴ [wel91] and *scalability*⁵. If and when a parameter estimation mechanism cannot be attached to a process dynamic model one must ask whether this is a weakness in the model or the absence of an appropriate technology. On the other hand the modeller(s) may be prepared to do without and consider this issue irrelevant.

Software Process Maturity and Process Monitoring

The above discussion relates to a category of model-based global level monitoring and measurement techniques that could be investigated, refined and deployed to address various aspects of software process dynamics. This suggests a complement to process improvement through process maturation. The current focus in software process improvement, whatever the method, has been to adjust the actual process, so that process output for a particular set of inputs, is more consistent, predictable and timely, less variable and of higher quality, however defined. Achieving a high level of maturity generally requires a sustained

⁴ As when, for example, it is not possible to obtain a unique set of model parameters from the available data.

⁵ For instance, numerical issues could arise when attempting to estimate large number of parameters.

effort of those involved. It transforms, in one way or the another, the internals of the related organisation. Assessment of the level of maturity is conducted through interviews, questionnaires and checklists, for example. The improvement process is *inside out*. Global level monitoring and measurement have the potential to reveal the effectiveness of software improvement efforts, transforming the improvement process to be *outside in* or *closed loop*.

In general, the above consideration of process dynamics monitoring suggests that maturity of the process monitoring scheme must be seen as different (and complementary) to maturity of the process itself. In current process improvement methods, a reduction in uncertainty of process behaviour is attempted by transforming (the inside of) the software process, to achieve decreasing process output variability. We suggest here that a *mature* process dynamics monitoring scheme will be one that helps to characterise and bound process uncertainty. For example, parameter estimation will yield confidence intervals for the parameters being estimated, and hence an estimate of process outcome variability.

Final Remark

This paper has identified a number of issues relating to software process dynamic models. It reflects the view that software process dynamics is a real and significant phenomenon that can be exploited for process management and improvement. Related issues have been outlined. The FEAST/2 [leh98a] project recently started plans to examine these and other issues [ram99]. These efforts could well contribute to achievement of the long held goal of integrated support environments for software evolution.

Acknowledgements

Many thanks are due to Blaise Egan (BT Labs) for his clarifications about Bayesian approaches to one of the authors (JFR). The authors acknowledge financial support from the UK EPSRC, grant numbers GR/K86008(FEAST/1) and GR/M44101 (FEAST/2). One of the authors acknowledges support from BT Labs (Short Term Research Fellowship).

References

- [abd91] Abdel-Hamid T and Madnick S E, *Software Project Dynamics - An Integrated Approach*, Prentice-Hall Softw. Series, Englewood Cliffs, NJ, 1991
- [boe81] Boehm BW, *Software Engineering Economics*, Prentice Hall, NJ, 1981
- [cha99] Chatters BW, Lehman MM, Ramil JF, Wernick P, *Modelling a Software Evolution Process*, submitted for publication, Jan. 1999
- [gus98] Gustafsson F and Graebe SF, *Closed-Loop Performance Monitoring in the Presence of System Changes and Disturbances*, Automatica, vol. 34, no. 11, 1998, 1311 - 1326
- [kus97] Kusumoto S *et al*, *A New Software Project Simulator Based on Generalized Stochastic Petri-net*, Proc. ICSE'97, May 17 - 23, 1997, Boston, MA, 293 - 302
- [leh74] Lehman MM, *Programs, Cities, Students, Limits to Growth?*, Inaugural Lecture, May 1974. Publ. in Imp. Col of Sc. Tech. Inaug. Lect. Ser., vol 9, 1970, 1974, 211 - 229. Also in Programming Methodology, (D Gries ed.), Springer Verlag, 1978, 42 - 62. Also reprinted in [leh85]
- [leh77] *ie*, *Human Thought and Action as an Ingredient of System Behaviour*, in Encyclopaedia of Ignorance, (R. Duncan and MW Smith eds.), Pergamon Press, Oxford, 1977. Also reprinted in [leh85]
- [leh85] Lehman MM and Belady LA, *Program Evolution, - Processes of Software Change*, Academic Press, London, 1985
- [leh96] Lehman MM and Stenning V, *FEAST/1: Case for Support*, Department of Computing, Imperial College, March 1996, <http://www-dse.doc.ic.ac.uk/~mml/feast>
- [leh97] Lehman MM, Perry DE, Ramil JF, Turski WM and Wernick PD, *Metrics and Laws of Software Evolution - The Nineties View*, Proc. Metrics '97, Albuquerque, NM, 5 - 7 Nov., 1997, 20 - 32. Also as: *Process Improvement - The Way Forward*, in Elements of Software Process Assessment and Improvement, K El Eman and N H Madhavji (eds.), IEEE CS Press, 1999

- [leh98a] Lehman MM and Wernick, *System Dynamics Models of Software Evolution Processes*, Proc. Int. Wrksh. on the Principles of Software Evolution, ICSE'98, Kyoto, Japan, April 20 - 21, 1998, 6 - 10
- [leh98b] Lehman MM, *FEAST/2: Case for Support*, Dept. of Comp, Imp. Col., Jul. 1998, available from FEAST web page, see ref. [leh96]
- [leh98c] Lehman MM, Perry DE and Ramil JF, *On Evidence Supporting the FEAST Hypothesis and the Laws of Software Evolution*, Proc. Metrics'98, Bethesda, Maryland, Nov. 20-21, 1998, 84 - 88
- [leh98d] *ie, Implications of Evolution Metrics on Software Maintenance*, Proc. International Conf. on Software Maintenance (ICSM'98), Bethesda, MD, Nov. 16-20, 1998, 208 - 217
- [leh98e] Lehman MM and Ramil JF, *Feedback, Evolution and Software Technology - Some Results from the FEAST/1 Project*, Invited Keynote Lecture, 11th Int. Conf. Softw. Eng. And its Applic., Preprints, Vol. 1, Paris, December 8-10, 1998, 1 - 11
- [lev90] Levary RR, *System Dynamics with Fuzzy Logic*, Int. J. Systems Sci., Vol. 21, 1990, 1701 - 1707
- [lju87] Ljung L, *System Identification: Theory for the User*, Prentice-Hall, Englewood Cliffs, NJ
- [pro98] *Pre-prints of Pro-Sim 98, Software Process, Simulation and Modelling Workshop*, Silver Falls, Oregon 1998 (selected papers to appear in Journal of Systems and Software, 1999).
- [ram98] Ramil JF and Lehman MM, *Fuzzy Dynamics in Software Project Simulation and Support*, Proc. EWSPT'6, Weybridge, UK, 16 - 18 Sep. 1998, LNCS 1487, Springer Verlag 1998, 122 - 126
- [ram99] *ie, Challenges facing Data Collection for Support and Study of Software Evolution Processes*, Submitted to ICSE 99 Workshop on Empirical Studies of Software Development and Evolution, ESSDE'99, Los Angeles, May 18, 1999
- [rio77] Riordan JS, *An Evolution Dynamics Model of Software Systems Development*, in Software Phenomenology - Working Papers of the (First) SLCM Workshop, Airlie, Virginia, Aug. 77. Pub. ISRAD/AIRMICS, Comp. Sys. Comm. US Army, Fort Belvoir, VI, Dec. 1977, 339 - 360
- [tur96] Turski WM, *Reference Model for Smooth Growth of Software Systems*, IEEE Trans. on Soft. Eng. v.22, n. 8, August, 1996, 599 - 600
- [wel91] Wellstead PE and Zarrop MB, *Self-Tuning Systems - Control and Signal Processing*, Wiley, Chichester, UK, 1991
- [wer98] Wernick P and Lehman MM, *Software Process White Box Modelling for FEAST/1*, ProSim'98, Proc. Int. Wrkshp on Softw. Proc. Simulation Modelling, June 22 - 24, 1998, Silver Falls, Oregon, to appear in Journal of Systems and Software, 1999
- [wet91] Wetherill GB and Brown BW, *Statistical Process Control: Theory and Practice*, Chapman & Hall, London, 1991.
- [zel87] Zellner, A, *An Introduction to Bayesian Inference in Econometrics*, Robert E. Krieger Pub. Co., Malabar, FL, Reprint Edition, 1987