Transactions on Information and Communications Technologies vol 11, © 1995 WIT Press, www.witpress.com, ISSN 1743-3517

A goal-oriented approach for managing software process change

D.S. Hinley,[a] S. Reiblein[b]

[a] *Department of Computer Science, University of Durham, Durham, DH1 3LE, UK*

[b] *Post-Office IT Consultancy, Chesterfield, Derbyshire, S40 1DY, UK*

Abstract

The constant demand for change to existing systems to meet emerging requirements, and a customer-driven interest in software quality management, e.g. TickIT, is necessitating a process-focused viewpoint. However, surveys of process capability, e.g. Koch [1], in Europe, the U.S., and Japan, have shown a low level of maturity, generally. There is a clear need for process improvement and not simply an assessment mechanism.

Software process models have been shown by Kellner [2], to facilitate managerial planning and control of existing processes, and can provide a much needed focus to the management of process change. This paper presents a Case Study on assessing improvements to software processes in relation to meeting organisational need and management objectives.

1 Introduction

System developers face a continual demand to adapt, enhance, improve, and migrate computer systems throughout their operational life. Evolution of existing systems takes place over a period which is often an order of magnitude greater than that of the original development timescale. System evolution is encompassed within the Standard definition of software maintenance, and the maintenance process generally is becoming more appreciated and indeed valued as a necessary form of asset management, and not simply as a corrective task, dealing with the quality failings of the development process.

The importance of software maintenance is now increasingly recognised within the IT industry and by the end-customer, e.g. overall life-cycle costs are now being considered in MoD procurement exercises, resulting in both researcher and practitioner interest in lesser known (or understood) quality attributes such as maintainability.

However, although the general problems of software maintenance have been recognised by Lientz & Swanson [3], from the management perspective, and part solutions have been put forward, there appears to be no generally accepted model of 'best-practice'. The process remains poorly understood and indeed poorly supported in terms of methods and tools.

Traditionally, software maintenance requests, apart from those limited to corrective activities, are managed generally under the auspices of a software project, and are subject to the same types of constraints as development projects. The constraints may be envisaged in terms of project and product risk; risks to project success are perceived principally in terms of time and cost, whereas product risks are perceived in terms of quality constraints. These constraints may be expressed in terms of goals to be achieved during the course of the project.

The emergence of IT as a 'service' industry, described by Rands [4], is now challenging the simplistic engineering project viewpoint, as the dominant management model governing software processes. However, time, cost and quality constraints are relevant also in the service sector and the goals are being formalised increasingly in terms of Service Level Agreements (SLA's). It would appear that IT Managers have a goal-oriented viewpoint, whether they be project managers, service support or delivery managers.

Within software engineering, there is a growing belief, e.g. Tully [5], that the product quality characteristics are determined by the process elements undertaken during their development and evolution. This has been shown to be the case in other branches of engineering e.g. metal manufacturing technology, in which manufacturing process models have been used not only to specify the process pathway in order to meet the product design criteria, but also to predict the quality characteristics of the product.

The development of such models and their validation can take some time and is only viable for the repetitive manufacture of similar components. A three-phased approach is often used, namely:

- **experimental phase** - in which the product concept information is used in order to develop a pilot manufacture process;
- **concurrent engineering phase** - the detailed design information is used to develop a suitable production process and the initial process capability is assessed;
- **verification phase** - the manufactured product is validated against the design criteria, and the production process is defined. The production process capability is assessed and thereby approved for product manufacture.

It is interesting to note that many of the existing software process capability assessment methods, e.g. SEI-CMM, Bootstrap, have an implicit process model, purporting to describe 'best-practice'. However, there is no attempt to match the process capability to the design intent (product characteristics), thus the goals remain elusive and divorced from the process purpose.

As yet software process models have not had the same degree of success, as manufacture process models. This may be due to the lack of formalised understanding concerning human activity systems. However, there is growing interest both from the developer and the customer perspective in assessing the process capability of the production unit, and supporting organisation.

Two important factors have thus emerged from this brief discussion on the semantics of process capability and in particular conceptual differences between manufacture and software engineering. Firstly, irrespective of whether the relationship between process and product becomes firmly established for software engineering, there is a recognised need to improve software processes in order to more adequately meet business goals and management objectives. Secondly, software processes, outside of the 'software factory', described by Humphrey [6], are unlikely to adopt an experimentation and concurrent engineering approach, however, they can benefit directly from the development of models to depict the actual process undertaken.

This paper describes how simple process models can be developed and used, partly to understand and reason about existing processes and their shortcomings, and partly to develop a measurement framework for the management of change to better achieve organisational goals.

Section 2 provides a brief introduction to software process modelling and describes the simple graphical constructs which have been used to represent software processes at both the managerial and technical level, and more recently to represent IT services, e.g. Systems Management.

Section 3 describes the development of a process measurement framework, based on the Goal / Question / Metric (GQM) paradigm, which attempts to overcome some of the management difficulties in applying GQM in a real situation.

The measurement framework and models of actual processes have been used in an extended Case Study of a maintenance process, and in particular to improve the management of maintenance in order to more adequately meet organisational goals. The lessons learnt from the process improvement Case Study were later employed in the adoption of a goal-oriented approach for the management of service provision. Section 4 describes the development of a goal-oriented approach for process management, and Section 5 presents our findings.

2 Software process modelling

Software process modelling is a relatively new technique which has been shown to be advantageous in a number of quality related areas, e.g. process description and definition, e.g. Drew [7], process improvement, e.g. Hinley & Bennett [8], the elicitation of management metrics, e.g. Shepperd [9], and for facilitating the management of processes, projects and services.

Five principle uses have been identified by Kellner [10], for software process models, they are to:

1. facilitate understanding and communication, e.g. managerial agreement and to underpin team working;
2. support process improvement - by facilitating organisational learning and process reuse and to support managed evolution;
3. support process management - by providing a framework for project planning and the basis for process measurement;
4. provide automated guidance in performing a process, e.g. by retaining reusable process representations in a repository;
5. provide automated execution support, e.g. supporting cooperative working.

Over the last decade, a variety of model formalisms have been put forward and there is a corresponding diversity in software process modelling techniques. For our purposes, a business process can be succinctly defined as a set of activities, tasks and actions performed by 'agents' with the intent to satisfy a purpose and achieve a goal. Hence, models of business processes are of direct interest to managers wishing to adopt a goal-oriented management approach.

Process models differ significantly from general management models, used for planning purposes, in that they are more detailed and support management in dealing with typical project and service issues, e.g. they:

- are responsive to 'customer' demands;
- represent organisational requirements;
- represent the necessary roles required to carry out the process;
- are capable of being compared and evolved;
- underpin process improvement initiatives.

Our models of software processes, for individual projects and IS/IT services, meet the above criteria, using five constructs: process element, product, information flow, process constraint and agents. The models have a distinct architecture, depicting the notional sequence of process elements, hierarchy, control mechanisms, and relevant perspectives. For example, a model of the software maintenance process has been constructed by Hinley & Bennett [11], from the management perspective, showing four control mechanisms: organisational, request, change and release. It shows the key management interactions with Senior Management and Customers, Quality Assurance, and technologists, e.g. Design Authority.

The models described above, provide a static description of actual processes, representing functional and structural components, and some behavioural features, i.e. agent communication and control.

The general definition of a process provides little insight into the practical problems associated with modelling. Key concerns are:

- defining the process scope - by our definition it is necessary to ascertain the purpose of the process and the overall goals. Managers do not readily relate to processes or goals as they tend to have functional responsibilities;
- process information gathering- the technique needs to be geared towards obtaining the necessary information for the construction of a model, e.g. it may be necessary to elucidate agents and role interactions;
- process analysis -it is necessary to structure the information gathered by analytical techniques e.g. abstraction and hierarchical structure, process dependencies may also be elucidated;
- model derivation - the model is constructed using the agreed notation;
- model validation - a walkthrough of the process model is undertaken between the analyst and the process owner.

The concerns expressed above, have been tackled in our Case Studies, with respect to goal-oriented approaches to the management of change. The process modelling work was, in accordance with the usage stated earlier, of significant benefit in terms of: facilitating management understanding and communication, supporting process improvement, and supporting process management.

The models provide a visualisation of complex processes and this was thought to contribute significantly to management understanding. The visualisation was facilitated through explicit recognition of process elements and their inter-relationships. Furthermore, models of actual processes highlighted potential problem areas and aspects for possible improvement. Furthermore, they provide a measurement datum, i.e. existing measures and data collection points for qualitative and quantitative assessment, and a foundation for process improvement.

3 Process measurement framework

Although software engineering projects have differing technical objectives, managers often adopt a simple project life-cycle model to help in their planning and to lessen the risks of missed schedules, over budget expenditure and poor product quality.

Software process models have been shown by Humphrey & Kellner [12], to facilitate management planning and control, and Hinley & Bennett [13], have described how models may help managers to achieve objectives in support of organisational goals. To achieve management objectives, whether they be in terms of project or service aims, technical and business strategies are required, which are in accordance with the overall organisational goals.Thus, it is imperative that managers not only have a plan, but a measurement framework to ascertain how well these goals are being achieved.

Measurement is an essential part of a mature process, it is also a useful component of quality management and project management. Measurement is also useful for prediction and can contribute towards building quality products, in defining services, as well as improved processes.

Appropriate metrics are normally determined by measurement goals, and several techniques for deriving metrics from goals have been proposed. Examples of these 'top-down' approaches include the Quality Function Deployment (QFD) approach by Akao, the Software Quality Metrics (SQM) approach by Murine, and the Goal / Question / Metric (GQM) approach by Basili & Weiss [14].

The QFD approach was originally developed in 1966 to serve as a framework for dealing with quality at the planning stage of manufacturing processes; it was later adapted to the software domain. The QFD approach tries to relate the characteristics of the final product to characteristics found in early deliverables, e.g. a design document may be analysed in order to predict maintainability characteristics of the final product.

The GQM paradigm aims at identifying what measures could be usefully employed in order to evaluate goal achievement. It may therefore be employed for a variety of software projects, the management of change, and also the assessment of services delivered by an IS / IT organisation.

The concept of measurement is the basis of the paradigm, and is important in that it addresses a fundamental weakness with current software engineering practice and also provides the basis for service definition and management. It attempts to facilitate the identification of useful metrics from explicit goals in a 'top-down' manner. As such it provides an informal mechanism for addressing management problems such as organisational change and process improvement.

The simplicity of the paradigm is beguiling, however, in practice goal setting is difficult and measurement generally leads to controversy about the appropriateness of metrics and data collection. Rombach & Basili [15], have set some guidelines on the formulation of a comprehensive goal (Goal Definition Template) and this addresses one area in which managers experience problems, i.e. the identification of goals. This problem has also been recognised by Gilb [16], in terms of his principle of fuzzy targets: "Projects without clear goals will not achieve their goals clearly".

The managerial problem in setting clear goals, i.e. to which some value can be attached has been recognised and goal definition templates may help in this task. However, experience and general consensus are also key factors in this process. Furthermore, having agreed high level goals, difficulties have been experienced in identifying sub-goal hierarchies and formulating questions which could possibly be answered within the organisation.

To alleviate these problems to some extent, we initially augmented the GQM paradigm with an analysis phase in which an orientation exercise is conducted with management. This exercise, provided some assessment of the feasibility of applying a measurement framework to the management of process change, within a particular organisational situation.

The second phase is to undertake goal-orientation, i.e. ascertain the goals, often based on the Mission Statement and Strategic Plans of an organisation. During this phase, existing process problems are catalogued and Critical Success Factors (CSF's) are identified. The organisational goals and CSF's are ranked and dependencies noted.

An inventive search, (question formulation concerning goal achievement) is facilitated by systematically exploring the current practice model and establishing the key process areas, i.e. groups of process elements are identified which are thought to have a direct bearing on a catalogued problem or CSF.

Finally, the model is used to derive process changes in order to better achieve organisational goals. The 'improvement model' is studied to identify potential data sources of both process and product data, in accordance with the measurement framework.

4 Change management evaluation

The above measurement approach was applied to an extended Case Study to improve the management of a software evolution process. Specific goals were to reduce the cost of software maintenance, whilst at least sustaining the product quality, measured in reliability terms.

In the first trial, six sub-goals were identified associated with: satisfying customer requirements, meeting time-scales, controlling costs, assuring software reliability, providing a fast-response to problems, and ensuring quality standards were adhered to. The project team evaluated the goals and the CSF's and considered the existing problems, which could be readily related to the graphical model of current practice. This model was used to identify potential process improvements, i.e. a process which was more suited to achieving the existing organisational goals. Furthermore, the model was used to identify the necessary process and product attributes which could be measured to ascertain goal achievement, with respect to both the current and changed process.

Six substantive changes were made to the existing managerial process, which over a period of twelve months, provided valuable management information, enabling the existing testing process to be questioned.

A number of sub-goals related to testing and likely to lead to some form of process improvement were formulated:

- discover failures earlier during the testing process;
- at least maintain the quality of the delivered product;
- decrease the direct labour effort required for testing;
- decrease the time to discover and fix faults.

The above sub-goals, related to the goal of reducing cost had an implied goal, i.e. to increase the productivity of personnel involved in the testing process. A number of process improvements were made, including: the level of test planning and team preparation, enhanced monitoring and control of testing activities, prioritisation of test schedules, in accordance with their ability to find faults, based on impact analysis.

We have briefly discussed two examples in which simple goals were decomposed and CSF's were identified and related to result areas, i.e. process elements which were thought to make a significant contribution towards the attainment of particular CSF's. They demonstrate one approach by which management measures may be elucidated with respect to organisational goals and management objectives.

Our third example is taken from a more general management group situation within a service organisation, in which a management team identified 14 Key Performance Indicators (KPI's) directly from the organisational Mission Statement and published goals. KPI's differ significantly from CSF's in that they are indicators and therefore, by definition have a value associated with them. The KPI's are used in a hierarchical manner, by being decomposed into performance indicators which may subsequently call for one or more measurements.

This Case Study attempted to assess whether a measurement framework could be formulated and applied directly without the need to undertake a process modelling exercise.

Following the goal orientation phase, the KPI's were 'desk-checked' and a variety of concerns were highlighted:

- some of the KPI's lacked longevity and potentially could have created an environment which lacked focus and thereby reducing management efficiency;
- many of the KPI's were directed towards highlighting performance of new products and services, but they neglected elements of establishing control;
- the indicators showed a general lack of consistency, for example, business unit revenue and costs were of interest but profit was ignored;
- some of the indicators lacked definitions, e.g. 'customer service' - this led to confusion or multiple interpretations.

As a direct consequence of these findings, it was decided to create a glossary of terms, and surprisingly, it was difficult to obtain precise definitions in areas of the business such as Finance. The definition of terms was a problem area in our original Case Studies; however, it could be addressed by means of the questions being asked about goals and by reference to the appropriate areas of the process model. In this latter Case Study it was necessary to tackle the semantics directly, in order to provide some confidence that the lack of clarity would not lead to sufficient uncertainty in managing with KPI's as to foster alternative management practices.

A major advantage of establishing KPI's and then explicitly drawing a three-layered measurement framework: KPI's, Performance Indicators, and measurements, was that dysfunctional performance indicators were immediately apparent. These are management measures which are used at a departmental or functional level, which have no relationship to KPI's and the organisational goals. These dysfunctional measures tended to be interweaved as management information systems or control functions within the process models, and were difficult to mask and ignore, in the previous Case Studies, i.e. to separate system from process. Clearly, process improvement and the measurement of goal achievement needs to be divorced from existing management practices, which do not provide valid or useful measures.

5 Findings

The latter Case Study, which involved a Senior Management team defining KPI's directly from a Mission Statement and high level goals, indicated that to use KPI's effectively there needs to be a full understanding of the purpose for which they are to be used. A team effort is required to explicitly state and agree the performance indicators, i.e. similar in approach to the top-down decomposition of the GQM paradigm.

The major drawback of the KPI approach is that it lacked a reference point , to which the team could address questions concerning the purpose of a particular KPI. For example, management usually contains elements of planning and control, although these aspects may have conflicting indicators.This may be overcome by a process model which represents the managerial process elements and thereby defines the purpose of the process.

A further problem area is the definition of terms; this is particularly acute, with immature processes and goal-orientation and a feasibility assessment are crucial in evaluating whether an organisation is ready to effectively manage processes and their evolution. It was noted that although many of the Managers interviewed could understand why KPI's were important, the notion that KPI's could be a vehicle for management metrics and prediction was outside their vision. The general management view was to manage the immediate, because the time horizon and general demand for results was becoming shorter.

In applying the KPI approach, problems were encountered with the lack of ownership, i.e. disassociation from performance measures.A solution to this problem was envisaged in terms of the devolution of KPI's throughout the organisation. However, workshops and communication exercises were found to be both time consuming and suffered from local attempts to undertake independent analysis and problem resolution. This suggests that process modelling may be beneficial in ascertaining process owners.

Both the GQM and KPI approaches need to be sensitive to the domain they are measuring. It was noted that KPI's had different sensitivities, i.e. some were highly reactive whereas others take much longer to display the impact of events. The level of sensitivity was used to dictate how often measurements were taken, but more importantly, they should also dictate management's response to changes. Determining and understanding the sensitivity of response is implied in the effective use of KPI's. Thus managers needed to devote time to understanding the dynamics of a KPI in the context of the organisation.

This finding has led to further insights into goal-oriented approaches, in that it suggests that a different perspective of the organisational process is required, other than that of our descriptive process models. It suggests that a behavioural or dynamic model would provide a useful management tool in order to reason about the affects of specific goals and KPI's. For example, a specific KPI within a service organisation may be 'market-share by business sector'. A simple dynamic model suggests that comparisons with the external market provide a useful basis at the organisation or market area level, but there are difficulties within the organisation in relating to this KPI across organisational boundaries and indeed the overall purpose of the KPI becomes less obvious and ambiguous. Furthermore, KPI's are only valuable if managers have power to influence the outcome, this infers responsibility and sphere of control.

6 Conclusion

The principle criterion for process improvement within our Case Studies has been the degree to which the existing processes support the achievement of organisational goals within project and service environments.

Goal achievement has been assessed using a measurement framework based on a derivative of the GQM paradigm and other hierarchical approaches, e.g. KPI's. The establishment of suitable measures which could be readily appreciated and adopted by management was found to be difficult, in practice, without a process model to graphically depict the existing process and thereby focus attention on the process purpose, i.e. goals.

Furthermore, the model of current practice could be used to develop an improvement model of proposed processes which would better achieve organisational goals. The measurement framework can be applied to the management of process change, if the measures were common to both the current and improved process, and they related directly to goals rather than individual management objectives and perceptions of what should be measured.

References

1. Koch, G.R. The Bootstrap initiative - reported benefits for the industry, in Esprit Bootstrap (ed H. Woda, W. Schynoll), pp.2-1 to 2-35, *Proceedings of IPSS-Europe Int. Conf. on Lean Software Development*, Stuttgart, Germany, 1992, C.E.C.DGXIII.

2. Kellner, M.I. Software process modeling support for management planning and control, pp.8-28, *Proc. 1st Int. Conf. on the Software Process*, Redondo Beach, Calif. IEEE Comp. Soc. 1991.

3. Lientz, B.P., Swanson, E.B. *Software Maintenance Management*, Addison-Wesley, 1980.

4. Rands, T. Information technology as a service operation, *Journal of Information Technology*, 1992, **7**, 189-201.

5. Tully, C. (Ed). *Representing and enacting the software process*, Proc. 4th Int. Software Process Workshop, Moretonhampstead, Devon. ACM Press, 1989.

6. Humphrey, W.S. Software and the factory paradigm, *Software Engineering Journal*, 1991, **6**(5), 370-376.

7. Drew, D.W. Developing formal software process definitions, pp.12-30, *Proc. of Conf. on Software Maintenance*, Montreal, Canada. IEEE Comp. Soc. 1993.

8. Hinley, D.S., Bennett, K.H. A process modelling approach to managing software process improvement, in Software Quality Management '93 (ed. M. Ross, C.A. Brebbia, G. Staples, J. Stapleton), pp.189-204, *Proc. of First Int. Conf. on Software Quality Management*.. Southampton. Computational Mechanics Publications. Elsevier, 1993.

9. Shepperd, M. Early life-cycle metrics and software quality models, *Information and Software Technology*, 1990, **32**(4), 311-316.

10. Kellner, M.I. Software process modeling: Value and experience, *SEI Technical Review*. 1989, 23-54.

11. Hinley, D.S., Bennett, K.H. Developing a model to manage the software maintenance process, pp.174-182, *Proc. of Conf. on Software Maintenance*, Orlando, Florida. IEEE Comp. Soc. 1992.

12. Humphrey, W.S., Kellner, M.I. Software process modeling: principles of entity process models, pp.331-342, *Proc. of the 11th Int. Conf. on Software Engineering*. IEEE Comp. Soc. 1989.

13. Hinley, D.S., Bennett, K.H. Reducing the risks in software improvement through process-orientated management, pp.319-328, *Proc. of Conf. on Software Maintenance*, Montreal, Canada. IEEE Comp. Soc. 1993.

14. Basili, V.R., Weiss, D.M. A methodology for collecting valid software engineering data, *IEEE Trans. Software Engineering*. 1984, SE -**10**(6), 728-738.

15. Rombach, H.D., Basili, V.R. Quantitative assessment of maintenance an industrial Case Study, pp.135-147, *Proc. of Conf. on Software Maintenance*, Austin, Texas. IEEE Comp. Soc. 1987.

16. Gilb, T. *Principles of Software Engineering Project Management*, Addison-Wesley, 1988.