The Software Process: A Perspective on Improvement

HELEN E. THOMSON AND PAM MAYHEW

University of East Anglia, Norwich NR4 7TJ, UK

This paper describes a new software improvement approach currently under development at the University of East Anglia. The two best known software approaches are summarized and used to explain why the new approach has been developed. Six organizations have been evaluated. Observations from these evaluations are included and a preliminary examination of the new approach is presented.

Received October 14, 1993, revised June 10, 1994

1. INTRODUCTION

It is estimated that cancelled software projects account for 15% of all software development efforts each year. It is also estimated that 50% of all large software development projects (those larger than 1000 function points) run over budget and behind schedule (Whiting, 1992), while almost all small projects run 20% over budget and 20% behind schedule (KPMG, 1993).

From these figures it can be concluded that most organizations are adversely affected either by software being delivered late by their suppliers, or by the late delivery of software to their customers (Bentley and Mayall, 1991). This has a damaging effect, not only for any organization supplying or receiving software, but also for the professionals who are producing the software. The end result is a poor return for an organization's investment in its software resources and this may result in an organization fighting for survival, rather than fighting for the competitive edge. A secondary, though equally damaging effect is that these delays and failures promote the mistrust of software professionals throughout an organization.

For organizations wishing to overcome their software problems there is a considerable amount of information available about the latest techniques that may be used to prevent project failures. Though these techniques have been well publicised and their success stories well documented, none of them seems to have gained universal acceptance. This apparent inertia may mean that organizations are happy to tolerate project failures as an acceptable overhead, or it may mean that organizations find it difficult to understand their software production problems and consequently cannot assess how a given technique can be applied to resolve them.

Software process improvement approaches have been developed for organizations who wish to understand and resolve their software problems. They define the practices necessary to assess the current software production method and through this expose the problem areas. The infrastructure can then be developed to support the production of reliable software. As improvement is a continuous process there is no easy one-off solution to software production problems, either for an organization or the software professionals within it. Improvement can only occur where there is a focused and sustained effort towards building a process infrastructure of effective software engineering and management practices.

The new approach presented in this paper has been designed to be easily assimilated and implemented by organizations who wish to feel in command of their software developments. The model itself, like most other software process improvement approaches, splits into the two phases of evaluation and improvement. The evaluation phase is less laborious than those approaches currently on offer; however, the improvement phase still requires a considerable commitment of time and effort at all levels within an organization. For the improvement phase the new approach not only includes many aspects in common with other approaches, such as software project planning and management, but identifies additional aspects of importance including how to measure improvement, which support tools to use and when to use them for most impact, how to involve customers and how to quantify the business value of the software department to the organization.

2. SOFTWARE PROCESS IMPROVEMENT APPROACHES

The general approach to software process improvement is shown in Figure 1. From this it can be seen that there are three main considerations common to most software process improvement approaches:

- An organization evaluates its current ability to produce quality software.
- Goals and longer term objectives are then set to improve this ability.
- The process of continuous improvement is initiated.

Though each software improvement approach is built on similar basic principles, each approach differs in its

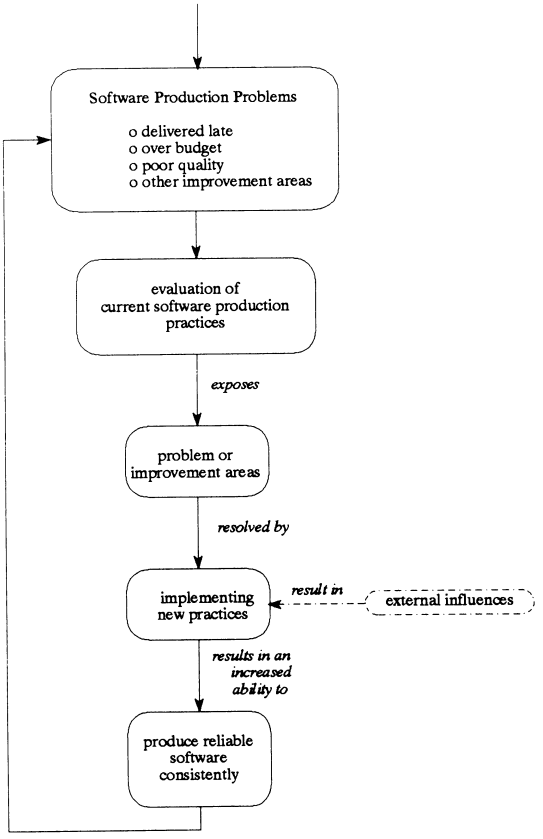

FIGURE 1. The continuous life of software process improvement.

perception of the operations that are necessary to build a good software process. This means that the organization evaluations and goals will vary depending on each approach's definition of the practices that result in a good software process. An additional complexity is that some approaches may be used to perform software audits and software certification against international standards. Additionally an approach may be used for first, second and/or third party assessment.

The two best known and marketed software improvement approaches are the Capability Maturity Model for Software (CMM) developed by the Software Engineering Institute (SEI) in Pittsburgh (Paulk *et al.*, 1993), and TickIT which was originally sponsored by the Department of Trade and Industry (DTI) (TickIT, 1992).

The CMM is a five-level model that has been developed to provide organizations with guidance for establishing software process improvement programs. It describes an evolutionary path from an *ad hoc*, immature software process (level 1) to a mature, disciplined and optimized software process (level 5). It covers practices for planning, engineering, and managing software development and maintenance. By following the practices defined by the CMM an organization should be able consistently and repeatably to produce quality software. This is a thoroughly researched and widely used model that has gained a considerable reputation and is accepted as a *de facto* standard by large organizations in America. Between 1988 and 1989, 80 assessments were made using the Software Process Maturity Model (SPMM) the forerunner of the CMM. A number of organizations, notably Hughes Aircraft (Humphrey *et al.*, 1991), have successfully developed the software process improvements defined by the CMM.

TickIT addresses those aspects of an organization's quality management system (QMS) which are concerned with the specification, design, development, installation and support of computer software. TickIT is seen as a stepping stone in the development of a QMS (within the context of Total Quality Management) based on, but not restricted to, definitive standards. TickIT applies the ISO 9000-3 guidelines for quality software systems to ISO 9001 which details the quality requirements for systems in general. The indications are that TickIT certification is doubling each year. By May 1992, 65 organizations had been certified (*TickIT News*, 1992), by December 1992, 149 organizations (*TickIT News*, 1993) and in 1993 it had been estimated that approximately 300 organizations had been certified internationally.

A number of initiatives have resulted from the CMM and TickIT developments (Thomson, 1993b). For example, the CMM has been tailored by Bell Canada to produce TRILLIUM (Coallier *et al.*, 1993) and by the Institute of Software Engineering in Belfast to produce the Software Quality Improvement Method (Thompson *et al.*, 1992). These are provided for the telecommunications industry and for small organizations of less than 50 people, respectively. TickIT, though developed by UK industry as a uniform national scheme, has spawned a number of similar initiatives in other countries (*TickIT News*, 1993; TickIT 1993a, b). Both TickIT and the CMM are influential in the ISO working group dealing with the development of international standards for software process assessment, improvement and capability determination (SPICE) (Fletcher, 1993). An

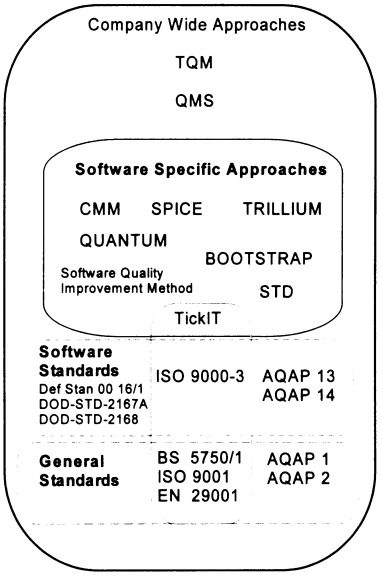

FIGURE 2. An informal guide to standards and approaches.

overview of these standards and approaches is given in Figure 2.

The customer base of the CMM is normally large organizations (with considerably more than 100 software professionals) working on large (over 100 man-years) American military projects. For Hughes Aircraft the cost of the initial assessment was estimated to be around $45 000 with the subsequent 2 year improvement program costing around $400 000. Hughes have also estimated that the annual saving of their improvement program is in the range of $2 million (Humphrey *et al.*, 1991).

TickIT is a less complex approach than the CMM. It can be used by any size of organization producing any type of software. The initial costs are around £10 000 in external fees to attain the TickIT certification standard, approximately 50 man-days of internal organizational effort and with an annual cost of surveillance of about £4500 (TickIT, 1993c). The benefits have been quantified as 25–50% of the failure costs. For example, it is estimated that for an organization with a turnover of £3 million per year that failure costs are £600 000 or 20% of turnover (TickIT, 1992). Using the 25–50% figure the savings would be £150 000–£300 000 per year. These figures are supported by a Price Waterhouse review (PW, 1988).

TickIT certification generally indicates that an organization is between level 2 and level 3 on the maturity scale defined by the CMM ratings. However, as with any certification scheme, it is open to abuse. Some organizations, once gaining certification, do not progress it as part of a Quality Management System (QMS) as recommended in the TickIT guide (TickIT, 1992). Such stagnation does not benefit either the certified organization or the purchasers who use certification as a method of guaranteeing quality software supply. This may mean that the scheme needs to be broadened to encompass an award for organizations that consistently improve their QMS within a defined schedule. The CMM on the other hand is both an evaluation and improvement method. Its size, complexity and expense put it out of the reach of many small and medium sized organizations and organizations with few formal software engineering practices may find it difficult even to contemplate its use.

3. THE NEW SOFTWARE PROCESS IMPROVEMENT APPROACH

The new software improvement approach is described in Appendix A and summarized in Figure 3. This approach has been developed for use by all sizes and types of software producing organizations at all stages of maturity. It should support those organizations lower than level 1 of the maturity scale (Finkelstein, 1992) as well as those, for example, who have gained or are gaining TickIT certification. The new approach views the software development process as being a number of separate processes each of which will have different significance to different organizations. These separate

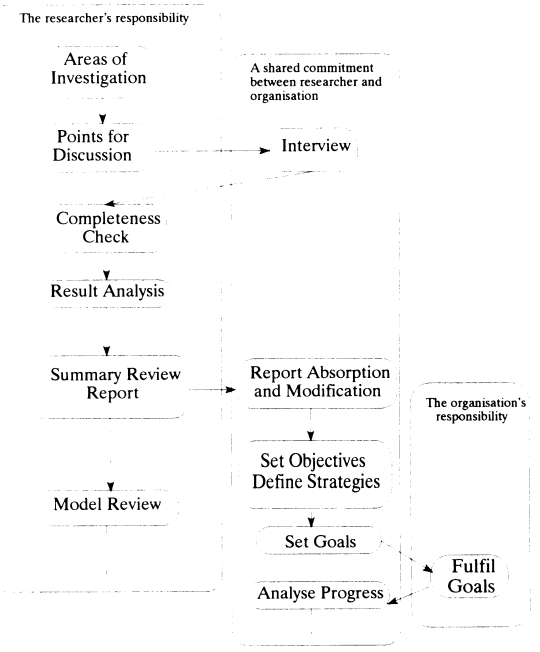

FIGURE 3. A new approach to software process improvement.

processes should complement and form an integral part of an organization's overall business plan. As such the new approach emphasizes where organizations can improve each of these separate processes. Implementing the goals which bring about software process improvement will enable the organization not only to produce better software but to retain its competitive edge.

The new model stresses the importance of providing an organization with a more easily assimilated software improvement approach that is simple to implement, and provides organizations with more achievable intermediate targets, and with objectives more appropriately tailored to their needs. This helps organizations develop their current abilities and augment these with the new skills necessary for survival. The following areas are emphasized by this new approach (Thomson, 1993a):

- Organisation background and culture.
- People and people management.
- Technology, methods and their management.
- The software process and its management (which includes measurement).

With the new approach the evaluation interview takes half a day. To make this possible the evaluator has to prepare thoroughly for the interview. Among other things this preparation will include reading all available documentation on the organization. If an organization wishes a software audit this may be run by the organization to complement the evaluation work.

This new approach analyses and evaluates each process used by an organization to produce software. The evaluation produces a software process profile which can then be used to set objectives for improvement. Where a maturity level is required by an organization a CMM assessment may be a more appropriate model for

use and the new approach does not preclude this. Where certification, e.g. TickIt, is required by an organization this may also be run in conjunction with this method.

4. OVERVIEW OF THE ORGANIZATIONS EVALUATED

A number of organizations were approached and offered an evaluation using the new software improvement approach. These organizations were known to the researchers and were happy to donate their time to help progress the research. Of the nine approached, six were evaluated in two phases with three organizations in each. After the first phase of evaluations some changes were made to the approach before the second phase began. Each organization evaluated was promised total anonymity and each software professional interviewed was promised complete confidentiality.

The organization participating in the research ranged in size from 106 to approximately 200 000 staff in total. Within these organizations the software sections evaluated varied in size from 40 to 1000 staff producing a variety of in-house and externally used software packages. The organizations involved in the first phase were in manufacturing, insurance and communications of which only the communications organization was producing software for external use. The second phase involved a manufacturing organization, a software house, and a computer software and hardware producing organization of which only the manufacturing organization was producing software for internal use. Each organization had varying levels of software quality management and a varying commitment to improving their software processes. Though the organizations were, to a large extent, self-selecting they provided a useful cross-section which was representative of the wider business community.

A detailed analysis of the results, and their effect on each individual organization, will be presented in the paper '*The Application and Evaluation of a Home Grown Software Process Improvement Model*' (Thomson, 1994). However a preliminary synopsis of the most interesting and absorbing findings is given below.

5. ORGANIZATIONS AND SOFTWARE PROCESS EVALUATION

Each organization took the evaluation process very seriously. In most cases the interview responses were honest and open. However, on receiving the evaluation report some initial defensiveness became apparent and an absorption period seemed necessary to avoid any adverse reaction.

With all organizations the plan had been to interview only one senior software professional. In three of the organizations two senior software professionals wished to take part in the interview and in one organization the Managing Director wished to be interviewed separately. This enhanced the evaluations and the evaluator's perspective of the organizations. It was noted by the researchers that the organization evaluation can benefit from having two software professionals participating in an interview or from having two or three separate interviews of selected staff. Since the researchers were trying to develop an evaluation technique which was relatively short this meant that we have to be wary about extending the evaluation interview to encompass multiple interviews.

6. APPRAISAL OF THE NEW SOFTWARE PROCESS EVALUATION APPROACH

Each evaluation provided a wealth of information about the organization and its software processes. However it also provided a wealth of information about the new software improvement approach. This information fell broadly into the two categories of guidelines and lessons learned.

6.1. Guidelines

Before evaluation takes place the evaluator should assess whether the organization's higher management are committed to software process improvement. If the commitment is not there the evaluation process should not proceed. There are many reason supporting this harsh statement. The main one is the demotivation factor. When software professionals participate in evaluation they commit to it and their expectations are raised. If there is no follow through to implementation they will feel let down by the organization. This can lead to all sorts of morale problems.

It should also be made quite clear, before the software process evaluation, to both the organization and software professionals being interviewed, that the objective is to *evaluate the software process*. Hence the software professionals themselves are *not* being evaluated, *no* audit of projects is being made and *no* certification of the software process is being given. This clarity of objective ensures that no defensive barriers need to be broken down during the software process evaluation interview.

Evaluators should take a slightly detached view and above all should listen and provide a sympathetic ear to those participating in the interview.

The evaluation should be processed, and the report produced, immediately—the quicker the better. When left over a number of weeks (or months) the organization can change beyond belief and make the assessment irrelevant.

6.2. Observations and lessons learned

There were a number of observations made and lessons learned during the software process evaluation and subsequent discussions of the resulting report. Many observations were specific to a single organization. For this paper the more organization-wide observations are

highlighted as a more complex evaluation has still to be made.

Some organizations wished the evaluator to take on the rôle of an educator and a barrier remover. This may mean that the organization has committed to improvement. Unless this rôle is formally adopted as part of the plan for software process improvement any proposal should be treated with caution.

For those organizations not previously committed to software process improvement, progressing from evaluation to implementation is a painful process. A reviewer can guide an organization to accept and identify problems with their software process through the software process evaluation report. A reviewer may then be involved in setting objectives and goals but software process improvement can only be effective when controlled from within the organization i.e. for software process improvement to be successful it has to be driven from within the organization itself.

7. CONCLUSION

The advantages of the new software process improvement approach over other methods were originally anticipated to be the inclusion of metrics, the use of tools and technology at the most appropriate time and the inclusion of customer driven aspects. This may be the case at a later stage of implementation but currently, during the software process evaluation phase, the main benefits have been that it is easily assimilated by organizations, it is not threateningly complex and can adapt to any size and quality of organization. The next stage of the research will give a clearer understanding of how organizations progress software process improvement.

The organizations involved in this research (and the researchers to a large extent) initially saw the problem of implementing software process improvement as being a technology driven issue. As organizations approached software process evaluation, and the planning for improvement that results from the evaluation report, a subtle change in the understanding of the problem took place. Technology issues took less priority than people issues, discussions on the implementation and management of change, and how software process improvement would become part of the overall organization's business objectives. There was also an acceptance that software process improvement takes time, commitment, hard work and is a continuous process.

Pursuing a practical line of research has been very rewarding. The researchers have greatly benefitted from seeing life breathed into the theory and method of the new approach—the organizations participating in the research can take all the credit for this. The enthusiasm and humour shown throughout has resulted in all participants looking forward to the next stage of the research.

REFERENCES

Bentley, C. and Mayall, A. (1991) Runaway systems. *Which Computer?*, **January**, 34–44.

CN (1993) Ambulance system failure blamed on management. *Commun. News*, **March**, 1.

Coallier, F., Gammage, N. and Graydon, A. (1993) *Trillium—Telecom Software Product Development Process Capability Assessment*, Draft 2.21. Bell Canada, Quebec.

Crosby, P. B. (1979) *Quality is Free*. McGraw-Hill, New York.

Dorling, A. (1993) SPICE (Software Process Improvement and Capability Evaluation). In *The Programme and Abstract Book for Software Process Modelling in Practice*, Butterworth-Heineman, Oxford.

Finkelstein, A. (1992) A software process immaturity model. *ACM SIGSOFT*, **17**(4), 22–23.

Fletcher, I. (1993) *Quality—The SPICE of Life. A report on the ISO SPICE Project on Software Assessment*. Caledonian University, Glasgow.

Humphrey, W. S. (1990) *Managing the Software Process*. Addison-Wesley, Reading, MA.

Humphrey, W. S., Snyder, T. R. and Willis, R. R. (1991) Software process improvement at Hughes Aircraft. *IEEE Software*, **July**, 11–23.

KPMG (1993) *IS Management Healthcheck. 1993 Survey Results*. KPMG Management Consulting, London.

Paulk, M. C., Curtis, B., Chrissis, M. B. and Weber, C. V. (1993) *Capability Maturity Model for Software, Version 1.1*. Technical Report CMU/SEI-93-TR-24. Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA.

PW (1988) *Software Quality Standards: The Costs and Benefits—A Review for the Department of Trade and Industry*. Price Waterhouse Management Consultants, London.

Radice, R. A., Harding, T., Munnis, P. E. and Phillips, R. W. (1985) A programming process study. *IBM Syst. J.*, **24**(2), 91–101.

Rubin, H. R. (1991) Software process maturity. *Am. Programmer*, **January**, 20–27.

Thompson, K., Ince, D. C., Madden, P. and Angelone, E. (1992) *Practical Quality Improvement through Software Process Maturity*. Institute of Software Engineering, Belfast.

Thomson, H. (1993a) *Components of the Software Process Improvement Model*. Unpublished PhD Thesis chapter, University of East Anglia.

Thomson, H. (1993b) *A Summary of Software Process Improvement Approaches*. Internal Report SYS-C93-11, University of East Anglia.

Thomson, H. (1994) *The Application and Evaluation of a Home Grown Software Process Improvement Model*. Working Paper, University of East Anglia.

TickIT (1992) *TickIT. Guide to Software Quality Management System Construction and Certification using ISO 9001/EN29001/BS 5750 Part 1*, Issue 2.0. Department of Trade and Industry, Department for Enterprise.

TickIT (1993a) *Public Procurement of Software in Europe Unlimited. A TickIT Information Sheet*. Department of Trade and Industry, Department for Enterprise.

TickIT (1993b) *Software Certification in Europe Unlimited. A TickIT Information Sheet*. Department of Trade and Industry, Department for Enterprise.

TickIT (1993c) *The TickIT Scheme. A TickIT Information Sheet*. Department of Trade and Industry, Department for Enterprise.

TickIT News (1992) Issue 2. Department of Trade and Industry, Department for Enterprise.

TickIT News (1993) Issue 3. Department of Trade and Industry, Department for Enterprise.

Waters, R. and Cane, A. (1993) Sudden death of a runaway bull. *Financial Times* (*Technology Section*), March 19, p. 11.
Whiting, R. (1992) Take the black magic out of software development. *Electronic Business*, **June**, 34–42.

APPENDIX A: DESCRIPTION OF THE NEW SOFTWARE IMPROVEMENT APPROACH

Figure 3 outlines the new model and the following paragraphs describe the new software improvement approach. Three areas of responsibility are shown, the researcher's responsibility, a shared commitment between the researcher and the organization, and the organization's responsibility. The areas of responsibility represent a notational division of labour which has assisted in the development and testing of the model. It is anticipated however that as the use of the approach shows benefits in, for example, staff motivation and productivity, and increased revenue, the organization will drive the improvements itself and integrate the software process improvement approach as part of its quality management system (QMS) and subsequently as part of its total quality management system (TQMS). The resulting model will then have no divisions of responsibility. However the model currently under test has these divisions and within each area of responsibility shown in Figure 3 there are a number of activities. The substance of each activity is described below.

A.1. Areas of investigation

The areas of investigation are the components which are considered fundamental when describing a well controlled and successful software producing organization. Many of these elements may exist as part of the organization's QMS or TQMS or may be adopted as part of these by the organization at a future date. Currently the model is used only in the software environment. The areas of investigation are used to assess an organization's present abilities to produce software. They cover the areas of the organization, people and people management, tools and technology and their management, software production and software production management. Each area of investigation is further subdivided into topics. For example, the people and people management area of investigation includes the topics of career progression, recruitment, selection and dismissal, culture, environment, communication, team and group balance, motivation and management (Thomson, 1993a).

The areas of investigation are the foundation for the points for discussion, the completeness check and the improvement goals.

A.2. Points for discussion

The points for discussion (Appendix B) are used in the review interview. They are a concise method of obtaining information about the organization's current ability to produce software.

The points for discussion are sent to organizations in advance of the review date to allow for some preparation time.

A.3. The interview

The participants in the interview are the head of the software department, the researcher and a tape recorder. The points for discussion are addressed. The time committed to the interview is finely balanced. It has to be long enough to allow the researcher to gain insight into the current software practices used within the organization but short enough, in the first instance, to gain the interest and commitment (to both the research and software process improvement) from the organization. For the first round of implementation the researcher proposed that the interview would take about half a day.

A.4. Completeness check and result analysis

After the review interview the completeness check (a tick list of more detailed questions) is completed by the researcher and the results analysed. The result analysis is the input for the review report which is prepared and subsequently presented to the organization.

A.5. Summary review report

A summary review report is produced for the organization and contains a concise list of strengths and weaknesses, two Kiviat (Rubin, 1994) charts of the organization profiles on readiness (the ability of the software organization to cope with new technology and tasks) and business value (the ability of the software organization to establish a link between their work effort and the value this gives to the business) and either a brief list of recommendiations if the organization is immature, or a list of objectives if the organization is already tackling software process improvement.

A.6. Report absorption and modification

The report is presented to the organization and a breathing space of about 2 weeks is then given before further discussion. This gives the organization time to absorb the contents of the report and associate with it. At subsequent discussions this report will be tailored to fit the requirements of the organization.

A.7. Set software process improvement objectives and define strategies

The implementation of the recommendations or objectives is then discussed and flexible schedules are agreed and driven by the organization. The software objectives form the vision of what the software section intends to attain over the long term. This vision can and will change as objectives are met and the improvements generate a positive feedback within the software section.

A.8. Set goals

Goals are units of clearly defined work. An example of one of the goals in the People and People Management Area of Investigation under the Performance Appraisal topic is:

Goal Define job Functions.

To be able to assess a software engineer's performance a job specification is necessary. The achievements of the software engineer may then be assessed against this specification. There may be many facets to each job function and off the shelf specifications may not suit the individual engineer. The specification of the engineer's job function is best when it is defined by the engineer in question but should be approved by the engineer's appraiser.

A.9. Fulfil goals

As each goal is fulfilled it marks a milestone in the organization's improvement of the software process. It may be that some objectives can only be partly completed, it may be that some goals and objectives need to be reprioritized but as long as each change is documented and approved the improvement can still progress in a measurable manner.

A.10. Analyse progress

At an agreed time progress along the path of improvement should be analysed. This tracking and reappraisal means that the organization can celebrate its progress and discuss problems. By doing such an analysis the organization is reaffirming its commitment to software process improvement. A new set of objectives and goals can then be confirmed and the model used recursively so that as the organization attains its objectives new ones can be set.

A.11. Model review

A similar review and analysis of the model itself should be made. Indeed the original model was much more complex than that shown in Figure 3, but with use, the simpler approach evolved and proved much more practical to the organizations participating in the research.

APPENDIX B: POINTS FOR DISCUSSION

B.1. Background

- For my own information I would like some background and history. For example, the age and size of the organization, some understanding of the management structure and the main products of the organizaton, and so on.
- I would like to know about the hierarchy of the organization, and in particular the software section and the tasks that are carried by the identifiable positions within the software organization.
- Have there been any major changes in the structure of the organization over the last few years?
- Has this organization's software process been assessed using, e.g. the Software Engineering Institute's (SEI) or BS5750 methods? If so did you find the assessment useful and have you implemented the resulting recommendations?
- When new projects are undertaken do they follow on from, or are they in addition to, current developments (i.e. is the software organization growing)?

B.2. Methods

- Has this organization adopted a software development method?
- Are all software staff trained in the use of the adopted software development method?
- Are tools used either to support the software development process or to automate repetitive jobs?
- Is there a review mechanism for assessing new technology and its implication on the current and planned work environment?

B.3. People

- Does this organization have a policy on people management, e.g. with regard to staff selection, motivation, appraisals, training, education and career development?
- What is the staff turnover?
- Is it easy to find suitable staff for all your projects?
- Do you feel that your organization has an identifiable and enforced culture?
- Is the amount of uninterrupted time made available to your staff considered important?
- What computer facilities are available to the software staff?

B.4. Quality

- Is historial project information stored and how is it used?
- Is the quality of each individual's work evaluated?
- Is the quality of each product evaluated?
- What standards of professionalism does this organization expect from its staff? And are these formally written down?

B.5. Management of co-operation and projects

- Is there a standard method of interfacing and communicating with customers?
- Is there a formal method of managing and tracking your projects?
- Are interfaces between teams, projects, departments and customers co-ordinated?
- Is the organization committed to improve its competitive edge in its business area?

- Does the software organization understand its role in the organization's overall business plan?

B.6. Conclusion

- If you could change, implement or introduce one element of the software process what would you do, why and how would you do it?
- Could you describe what the organization hopes to gain from this review and if the organization envisages implementing the improvements that will be suggested as part of the review?

GLOSSARY

First party assessment
An organization assesses itself against its own standards.

Second party assessment
An organization is assessed by a purchaser against their standards.

Third party assessment
An organization is assessed, at its own request, by an external body which is not itself a direct purchaser. This is called *Independent Certification* when the assessment is to an approved standard.

Independent certification
An organization is assessed, at its own request, by an external body which is not itself a direct purchaser. The external body has been approved to carry out *certification* to an approved standard, e.g. ISO9001.

ABBREVIATIONS

AQAP	Allied Quality Assurance Publication (NATO)
BSI	British Standards Institute
CMM	Capability Maturity Model
DOD	Department of Defence (USA)
DTI	Department of Trade and Industry
EC	European Commission
EN	Euro Norm
ESPRIT	European Strategic Programme for Research and Development in Information Technology
ESSI	European System and Software Initiative
IEEE	Institute of Electrical and Electronic Engineers
ISE	Institute of Software Engineering (Belfast)
ISO	International Standards Organization
MOD	Ministry of Defence (UK)
QMS	Quality Management System
SEI	Software Engineering Institute (Carnegie Mellon University, Pittsburgh)
SPICE	Software Process Assessment, Improvement and Capability dEtermination
SPMM	Software Process Maturity Model
SQM	Software Quality Management
STD	Software Technology Diagnostic
TQM	Total Quality Management